المقدمة

العنصر	الوصف
عنوان البحث / اللغة العربية	project proposal no 1
عنوان البحث / اللغة الانجليزية	project proposal no 1
الاولوية	النفط والغاز
التخصص العام	تقنيات معالجة مياه الشرب
التخصص الدقيق	المعالجة الغشائية
الباحث الرئيسي / اللغة العربية	احمد ابراهيم عبده شرف
الباحث الرئيسي / اللغة الانجليزية	ahmed Ibrahim Abduo Sharaf
تاريخ الارسال	23-10-2015
مرحلة التقديم	30-3-2015
مدة المشروع -بالشهور	12

فريق العمل

#	اسم الباحث	التخصص العام	الوظيفة	الدرجة العلمية
1	احمد ابراهيم عبده شرف	هندسة حاسبات	باحث رئيسي	أستاذ مساعد

CHAPTER

Template Matching

5

5.1 INTRODUCTION

In this chapter, we assume that each class is represented by a single pattern. A set of such *reference* patterns (or *prototypes*) is available and stored in a database. Given an unknown *test* pattern, template matching consists of searching the database for the reference pattern most "similar" to the given test pattern. This is equivalent to defining a matching cost that quantifies similarity between the test pattern and the reference patterns.

Template-matching techniques are very common in string matching, speech recognition, alignment of molecular sequences, image retrieval, and so forth. They often come with different names depending on the application. For example, in speech recognition the term *dynamic time warping* is used, whereas in string matching *Edit* (or *Levenstein*) *distance* is quite common.

This chapter is devoted to a series of examples of increasing complexity, culminating in an example from speech recognition.

5.2 THE EDIT DISTANCE

A *string* pattern is defined as an *ordered sequence of symbols* taken from a discrete and finite set. For example, if the finite set consists of the letters of the alphabet, the strings are words. The *Edit distance* between two string patterns A and B, denoted $D(A,B)$, is defined as the minimum total number of changes (C), insertions (I), and deletions (R) required to change pattern A into pattern B,

$$D(A,B) = \underbrace{\min}_{j}[C(j) + I(j) + R(j)] \tag{5.1}$$

where j runs over all possible combinations of symbol variations in order to obtain B from A. If the two strings are exactly the same, then $D(A,B) = 0$. For every symbol "change," "insertion," or "deletion," the cost increases by one.

The required minimum is computed by means of the dynamic programming methodology [Theo 09, Section 8.2.1]. That is, an optimal path is constructed in the 2-dimensional grid formed by the two sequences in the 2-dimensional space, by locating one sequence across the horizontal axis and the other across the vertical axis. The Edit distance is commonly used in spell-checking systems where the prototypes stem from the vocabulary of words.

DOI: 10.1016/B978-0-12-374486-9.00005-1

Example 5.2.1. Compute the Edit distance between the words "book" and "bokks," taking the former as the reference string. Plot the optimal matching path and comment on the sequence of operations needed to change "bokks" to "book." Repeat for the words "template" (reference) and "teplatte."

Solution. Use function *editDistance* by typing

```
[editCost,Pred]=editDistance('book','bokks');
```

The Edit distance equals 2 and is stored in the variable *editCost*. The value of 2 is the result of a symbol change (*k* to *o*) and a deletion (*s* at the end of the word). To extract the matching path, give matrix *Pred* as input to function *BackTracking* as follows:

```
L_test=length('bokks'); % number of rows of the grid
L_ref=length('book'); % number of columns of the grid
[BestPath]=BackTracking(Pred,L_test,L_ref,1,'book','bokks');
% The fourth input argument indicates that a plot of the best
% path will be generated. The last two arguments serve as labels for the
resulting axes.
```

The resulting best path is stored in the vector variable *BestPath* and is presented in Figure 5.1(a), where the reference pattern has been placed on the horizontal axis. Each element of vector *BestPath* is a complex number and stands for a node in the path. The real part of the element is the node's row index and the imaginary part is its column index. Inspection of the path in Figure 5.1(a) reveals that the first

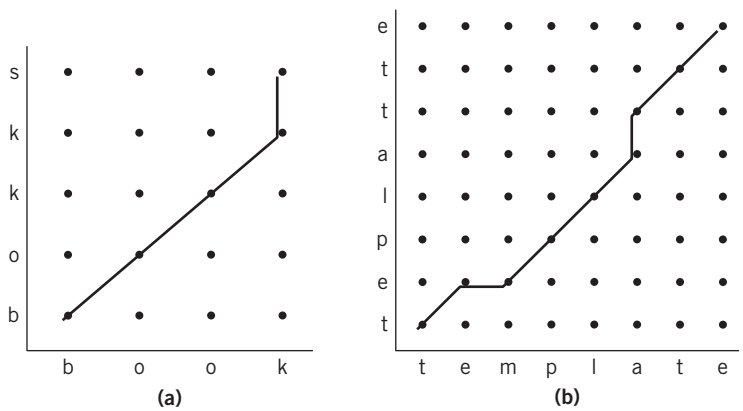

FIGURE 5.1

(a) Optimal matching path between "book" and "bokks." (b) Optimal matching path between "template" and "teplatte." A diagonal transition between two successive nodes amounts either to zero cost (if the corresponding symbols are the same) or to one (if the corresponding symbols are different). Horizontal transitions (symbol insertions) and vertical transitions (deletions) contribute one to the total cost.

occurrence of k in "bokks" has been changed to o. In addition, the vertical segment of the best path is interpreted as the deletion of s in "bokks." A total of one symbol change and one symbol deletion is needed to convert "bokks" to "book" in an optimal way.

Similarly, the Edit cost for matching "teplatte" against "template" equals 2, and the resulting best path can be seen in Figure 5.1(b). In this case, an insertion (horizontal segment) and a deletion (vertical segment) are required to convert "teplatte" to the reference pattern in an optimal way. ■

Exercise 5.2.1

Given the words "impose," "ignore," and "restore" as prototypes, determine which one stands for the most likely correction of the mistyped word "igposre" in terms of the edit distance. Note that, as is the case with most spell checkers, ties result in multiple correction possibilities.

5.3 MATCHING SEQUENCES OF REAL NUMBERS

In this section, we focus on a slightly different task, that of matching sequences of real numbers. In contrast to Section 5.2, where the goal was to change one string pattern to another, the aim here is to measure how similar/dissimilar are two given *ordered* sequences of numbers. For example, if we are given two real numbers, x, y, their similarity can be quantified by the absolute value of their difference. If we are given two vectors (i.e., two strings of real numbers of *equal* length), we can use the respective Euclidean distance. A more interesting case is when two sequences of numbers are of different length. One approach to this problem is to allow local "stretching"/"compressing," known as *warping*, achieved by constructing the optimal (low-cost) path through nodes of the respective grid. The grid is formed by the two sequences in the 2-dimensional space by locating one sequence along the horizontal axis and the other along the vertical axis.

Assuming that the reference sequence is placed on the horizontal axis, the dimensions of the grid are $J \times I$, where I and J are the lengths of the reference and test sequences, respectively. In the simplest case, the cost assigned to each node of the grid is equal to the absolute value of the difference between the respective numbers associated with a specific node. The type and the allowable degree of expansion/compression are determined by the so-called local constraints. Popular choices include the Sakoe-Chiba and Itakura constraints [Theo 09, Section 8.2]. Basically, these are constraints imposed on the allowable jumps among nodes in the grid.

The purpose of matching sequences of numbers is pedagogical and is used as an intermediate step to help the reader acquire a better understanding of the concepts underlying dynamic time warping for speech recognition, which is treated in the next section (see [Theo 09, Section 8.2.3]).

Example 5.3.1. Let $P = \{-1, -2, 0, 2\}$ be the prototype and $T = \{-1, -2, -2, 0, 2\}$ be the unknown pattern.

1. Compute the matching cost and the resulting best path by adopting the Sakoe-Chiba local constraints. Comment on the shape of the resulting best path.
2. Repeat with $T = \{-1, -2, -2, -2, -2, 0, 2\}$.

Solution

Step 1. For $T = \{-1, -2, -2, -2, -2, 0, 2\}$, use function *DTW Sakoe* and type

```
P=[-1,-2,0,2];
T=[-1,-2,-2,0,2];
[MatchingCost,BestPath,D,Pred]=DTWSakoe(P,T,1);
```

where D is the array having as elements the costs associated with optimally reaching each node of the grid. The value 1 is used if a plot is required; if not, 0 is used. Although the two sequences differ by one symbol, the matching cost equals 0. This is due to the fact that a) the absolute difference was employed as the (node) cost and b) the only difference between the two sequences is symbol repetition.

To further interpret the result, observe the respective best path in Figure 5.2(a). It can be seen that the vertical segment of the path corresponds to a local stretching operation; that is, the symbol -2 of the prototype (horizontal axis) is matched against two consecutive occurrences of -2 in sequence T.

Step 2. To repeat the experiment with $T = \{-1, -2, -2, -2, -2, 0, 2\}$ type

```
P=[-1,-2,0,2];
T=[-1,-2,-2,-2,-2,0,2];
[MatchingCost,BestPath,D,Pred]=DTWSakoe(P,T,1);
```

The matching cost remains 0 and the resulting best path is presented in Figure 5.2(b). It can be seen that the vertical segment of the path is now four nodes long. This should not come as a surprise

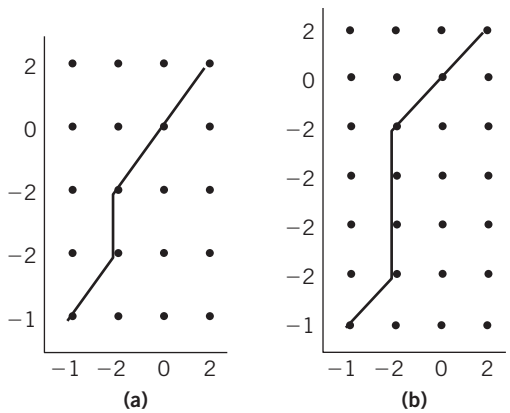

FIGURE 5.2

(a) Best path for $P = \{-1, -2, 0, 2\}$ and $T = \{-1, -2, -2, 0, 2\}$. (b) Best path for $P = \{-1, -2, 0, 2\}$ and $T = \{-1, -2, -2, -2, -2, 0, 2\}$.

because, as before, T differs from P in terms of *symbol repetition*. *The length of repetition does not affect the cost*; it only causes a more intense time-warping effect. To draw an analogy with speech, we may have the same phoneme but one time it is said fast and another time slowly. Long horizontal or vertical segments are very common when the Sakoe-Chiba local constraints are employed. If no global constraints are specified, the horizontal (vertical) segments can become arbitrarily long. This behavior is often undesirable with real-world signals, such as in speech.

Example 5.3.2. Let $P = \{1,0,1\}$ be the prototype, and let $T_1 = \{1,1,0,0,0,1,1,1\}$, $T_2 = \{1,1,0,0,1\}$ be two unknown patterns. Compute the matching cost for the standard Itakura local constraints between P and T_1 and between P and T_2.

Solution. To compute the matching cost for the two unknown patterns using the standard Itakura local constraints, use function *DTWItakura* and type

```
P=[1,0,1];
T1=[1,1,0,0,0,1,1,1];
T2=[1,1,0,0,1];
[MatchCost1,BestPath1,D1,Pred1]=DTWItakura(P,T1,1);
[MatchCost2,BestPath2,D2,Pred2]=DTWItakura(P,T2,1);
```

The returned value of *MatchCost*1 is ∞, whereas the value of *MatchCost*2 is 0. This is because one property of the standard Itakura constraints is that the maximum allowed stretching factor for the prototype is 2. In other words, the length of the unknown pattern has to be, in the worst case, twice the length of the prototype. If this rule is violated, the *DTWItakura* function returns ∞. In the case of P and T_2, this rule is not violated and the best path can be seen in Figure 5.3.

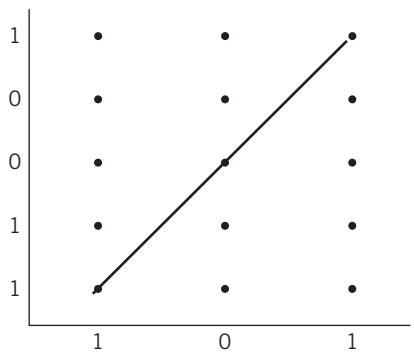

FIGURE 5.3

Best path for $P = \{1,0,1\}$ and $T = \{1,1,0,0,1\}$ using the standard Itakura local constraints.

Example 5.3.3. This example demonstrates the importance of the endpoint constraints [Theo 09, Section 8.2.3]. Let the sequence $P = \{-8, -4, 0, 4, 0, -4\}$ be a prototype. Also let the sequence $T = \{0, -8, -4, 0, 4, 0, -4, 0, 0\}$ be the unknown pattern.

1. Compute the matching cost by adopting the Sakoe-Chiba local constraints and comment on the result.
2. Repeat, allowing for endpoint constraints. Specifically, omit at most two symbols from each endpoint of T.

Solution

Step 1. For the first case, type

```
P=[-8,-4,0,4,0,-4];
T=[0,-8,-4,0,4,0,-4,0,0];
[MatchingCost,BestPath,D,Pred]=DTWSakoe(P,T,1);
```

The matching cost turns out to be 16. In practice, the cost is normalized by dividing it by the length of the best path. In this example, the best path is 9 nodes long, as can be seen in Figure 5.4(a), and so the normalized cost is 1.778. Hence, although P and T can be considered practically the same (they only differ in the trailing zeros), the matching cost is nonzero.

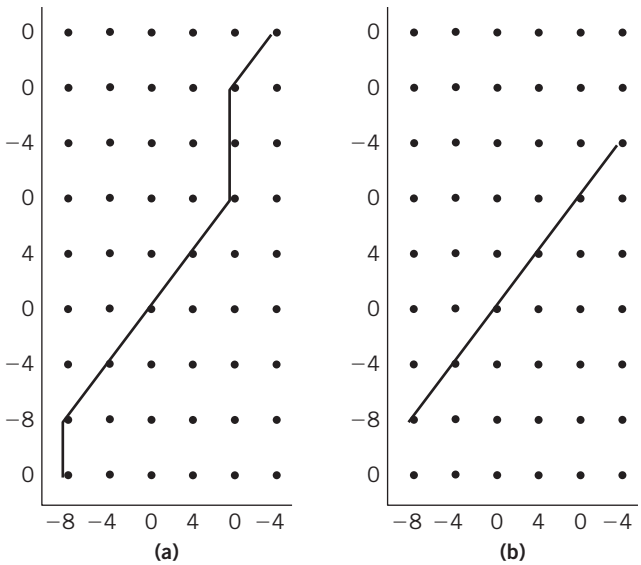

FIGURE 5.4

(a) Best path for $P = \{-8, -4, 0, 4, 0, -4\}$ and $T = \{0, -8, -4, 0, 4, 0, -4, 0, 0\}$, no endpoint constraints.
(b) Previous sequences matched while allowing endpoint constraints.

Inspection of Figure 5.4(a) reveals that time stretching occurs at the beginning and end of the path and is due to the existence of zeros at the endpoints. This is a common situation; that is, the unknown pattern contains "garbage" near the endpoints. As a remedy, we can resort to a variation of the standard matching scheme, where it is possible to omit a number of symbols at the endpoints of the unknown pattern; it suffices to specify the maximum number of symbols to omit. This type of enhancement to the standard matching mechanism can be easily embedded in the dynamic programming methodology.

Step 2. To employ the endpoint constraints in the current example, use function *DTWSakoeEndp* and type

```
P=[-8,-4,0,4,0,-4];
T=[0,-8,-4,0,4,0,-4,0,0];
[MatchingCost,BestPath,D,Pred]=DTWSakoeEndp(P,T,2,2,1);
```

In this function call, the third and fourth arguments stand for the number of symbols that can be omitted at each endpoint (2 in this example). The fifthindicates that a plot of the best path will be generated. The resulting matching cost is zero, as can be verified from Figure 5.4(b), where the first row and the last two rows of the grid have been skipped by the algorithm.

Endpoint constraints are very useful in speech recognition because the unknown pattern usually has silence periods around the endpoints, whereas the prototypes are free of such phenomena. ■

5.4 DYNAMIC TIME WARPING IN SPEECH RECOGNITION

Here we focus on a simple task in speech recognition known as *isolated word recognition (IWR)*. We assume that the spoken utterance consists of discrete words; that is, there exist sufficient periods of silence between successive words (hence the term "isolated"). This is a convenient assumption, since it allows for employing segmentation algorithms capable of locating the boundaries (endpoints) of the spoken words with satisfactory precision. Note that in practice a certain amount of silence/noise is likely to exist close to the endpoints of the detected word after the segmentation stage.

At the heart of any IWR system is an architecture consisting of a set of reference patterns (prototypes) and a distance measure. Recognition of a test (unknown) pattern is achieved by searching for the best match between the test and each one of the reference patterns, on the basis of the adopted measure.

As a first stage, a feature extraction algorithm converts each signal into a sequence of feature vectors—instead of matching sequences of real numbers, the task is computing the matching cost between two sequences of vectors. However, the rationale is the same, and the only difference lies in replacing the absolute value with the Euclidean distance between vectors. In speech recognition, this type of matching is known as *dynamic time warping* (DTW).

We now develop a simple, speaker-dependent IWR system for a 10-word vocabulary consisting of "zero," "one," . . . , "nine" and uttered by a single male speaker. We are actually building an "isolated digit recognition" system. We need a total of 10 utterances to use as prototypes and a number of utterances for testing.

At this point, you may record your own audio files or you may use the files available via this book's website. If you record your own files, name the prototypes as follows: *zero.wav*, *one.wav*, and so on, for the sake of compatibility, and place all 10 in a single folder (this is the naming convention we have adopted for the audio samples on the website). Note that "clean" prototypes are desirable. That is, make sure silence/noise has been removed, at least to the extent possible, from the endpoints of the prototypes before storing. Such care need not be taken with the samples that will be used for testing. In addition, the file names for the test patterns need not follow any naming convention and it suffices to store the unknown patterns in the folder where the prototypes are held.

To build the system, we use short-term energy and short-term zero-crossing rate as features [Theo 09, Section 7.5.4], so that each signal is represented by a sequence of 2-dimensional feature vectors. Note that this is not an optimal feature set in any sense and has only been adopted for simplicity.

The feature extraction stage is accomplished by typing the following code:

```
protoNames={'zero','one','two','three','four','five',...
   'six','seven','eight','nine'};
for i=1:length(protoNames)
   [x,Fs,bits]=wavread(protoNames{i});
   winlength = round(0.02*Fs); % 20 ms moving window length
   winstep = winlength; % moving window step. No overlap
   [E,T]=stEnergy(x,Fs,winlength,winstep);
   [Zcr,T]=stZeroCrossingRate(x,Fs,winlength,winstep);
   protoFSeq{i}=[E;Zcr];
end
```

which performs feature extraction per prototype and uses a single cell array (*protoFSeq*) to hold the feature sequences from all prototypes. To extract the features, a moving windows technique is employed [Theo 09, Section 7.5.1]. The length of the moving windows is equal to 20 milliseconds and there is no overlap between successive windows.

To find the best match for an unknown pattern—for example, a pattern stored in file *upattern1.wav*—type the following code:

```
[test,Fs,bits]=wavread('upattern1');
winlength = round(0.02*Fs); % use the same values as before
winstep = winlength;
[E,T]=stEnergy(test,Fs,winlength,winstep);
[Zcr,T]=stZeroCrossingRate(test,Fs,winlength,winstep);
Ftest=[E;Zcr];

tolerance=0.1;
LeftEndConstr=round(tolerance/winstep); % left endpoint constraint
RightEndConstr = LeftEndConstr;
for i=1:length(protoNames)
   [MatchingCost(i),BestPath{i},D{i},Pred{i}]=DTWSakoeEndp(...
        protoFSeq{i},Ftest,LeftEndConstr,RightEndConstr,0);
end
```

```
[minCost,indexofBest]=min(MatchingCost);
fprintf('The unknown pattern has been identified as %s \n',...
        protoNames{indexofBest});
```

This code uses the standard Sakoe local constraints and allows for endpoint constraints. Specifically, it is assumed that the length of silence/noise at each endpoint may not exceed 0.1 seconds (i.e., at most 5 frames can be skipped from each endpoint of the test utterance). Note that, instead of using the function *DTWSakoeEndp* to compute the matching cost, we could have used *DTWItakuraEndp*. To avoid repeating the code for each unknown pattern, the whole system is available on the website as a single m-file under the name *IsoDigitRec.m*.

Remarks

- Errors may occur in the experiments. This is also expected in practice, and is even more true here since only two features have been used for pedagogic simplicity.
- Moreover, this a speaker-dependent speech recognition example. Hence, if you record your own voice and test the system using the provided prototypes, the performance may not to be a good one, especially if the accent of the speaker is very different from the accent we used to record the prototypes. You can reconstruct the whole example by using your own voice for both the test data and the prototypes.

Attribute reduction for dynamic data sets

Feng Wang[a], Jiye Liang[a,*], Chuangyin Dang[b], Yuhua Qian[a]

[a]*Key Laboratory of Computational Intelligence and Chinese Information Processing of Ministry of Education, School of Computer and Information Technology, Shanxi University, Taiyuan, 030006, Shanxi, China,*
[b]*Department of System Engineering and Engineering Management, City University of Hong Kong, Hong Kong*

Abstract

Many real data sets in databases may vary dynamically. With such data sets, one has to run a knowledge acquisition algorithm repeatedly in order to acquire new knowledge. This is a very time-consuming process. To overcome this deficiency, several approaches have been developed to deal with dynamic databases. They mainly address knowledge updating from three aspects: the expansion of data, the increasing number of attributes and the variation of data values. This paper focuses on attribute reduction for data sets with dynamically varying data values. Information entropy is a common measure of uncertainty and has been widely used to construct attribute reduction algorithms. Based on three representative entropies, this paper develops an attribute reduction algorithm for data sets with dynamically varying data values. When a part of data in a given data set is replaced by some new data, compared with the classic reduction algorithms based on the three entropies, the developed algorithm can find a new reduct in a much shorter time. Experiments on six data sets downloaded from UCI show that the algorithm is effective and efficient.

Keywords: Dynamic data sets, Rough sets, Information entropy, Attribute reduction

1. Introduction

In real world databases, data sets usually vary with time. This phenomenon occurs in many fields such as economic research, social survey and medical research. As data sets change with time, especially at an unprecedented rate, it is very time-consuming or even infeasible to run repeatedly a knowledge acquisition algorithm. To overcome this deficiency, researchers have recently proposed many new analytic techniques. They usually can directly carry out the computation using the existing result from the original data set. These techniques mainly address knowledge updating from three aspects: the expansion of data [1-7], the increasing number of attributes [8-11] and the variation of data values [12-13]. For the first two aspects, a number of incremental techniques have been developed to acquire new knowledge without recomputation. However, little research has been done on the third aspect in knowledge acquisition, which motivates this study. This paper concerns attribute reduction for data sets with dynamically varying data values. For convenience of the following discussion, here are some specific explanations regarding data sets with dynamically varying data values. Generally speaking, this case usually occurs in the following several situations. One situation is that a part of the original data in a database is identified as wrong, thus needing to be corrected. Wrong data obviously lose their value to store further, and will be directly replaced by correct ones. Another familiar situation is that, initially collected data in a database may increase gradually, though it usually is not necessary to acquire knowledge from total data all the time. In other words, the sizes of interested data sets do not change. For example, in a pollution survey of X city, observed data in last few years or even decades may be stored in a database totally. However, analysis of pollution in each week (or each day, each month, etc.) does not require total observed data in the past. In this situation, because data sets of adjacent time intervals are usually similar to each other, an interested data set at one moment can be slightly amended to obtain a data set for the next moment. In addition, with

*Corresponding author. Tel./Fax: +86 0351 7018176.
Email addresses: sxuwangfeng@126.com (Feng Wang), ljy@sxu.edu.cn (Jiye Liang), mecdang@cityu.edu.hk (Chuangyin Dang), jinchengqyh@126.com (Yuhua Qian)

the rapid development of information technology, timeliness of data becomes more and more important. Thus, any out-of-date data in databases are usually useless. To improve the efficiency of knowledge acquisition, useless data should be directly updated by the latest or real-time ones. Furthermore, other similar situations occur rather often in applications such as stock analysis, tests for the disease and annual appraisal of workers.

Feature selection, a common technique for data preprocessing in many areas including pattern recognition, machine learning and data mining, has hold great significance. Among various approaches to select useful features, a special theoretical framework is Pawlak's rough set model [14,15]. One can use rough set theory to select a subset of features that is most suitable for a given recognition problem [16-21]. Rough feature selection is also called attribute reduction, which aims to select those features that keep the discerniblity ability of the original ones[22-26]. The feature subset generated by an attribute reduction algorithm is called a reduct. In the last two decades, researchers have proposed many reduction algorithms [27-32]. However, most of these algorithms can only be applicable to static data sets. Although several algorithms have been proposed for dynamic data sets [1-11], as mentioned above, they are incremental approaches only for the dynamic expansion of data or attributes.

This paper focus on attribute reduction for dynamically varying data values. To tackle this problem, this paper will exploit information entropy. The information entropy from classical thermodynamics is used to measure out-of-order degree of a system. Information entropy is introduced in rough set theory to measure uncertainty of a given data set [33-36], which have been widely applied to devise heuristic attribute reduction algorithms [37-41]. Complementary entropy [33], combination entropy [35] and Shannon's entropy [36] are three representative entropies which have been mainly used to construct reduction algorithms in rough set theory. To fully explore properties in reduct updating caused by the variation of data values, this paper develops an attribute reduction algorithm for dynamic data sets based on the three entropies. In view of that a key step of the development is the computation of entropy, this paper first introduces three updating mechanisms of the three entropies, which determine an entropy by changing one object to a new one in a decision table. When only one object is changed, instead of recomputation on the given decision table, the updating mechanisms derive new entropies by integrating the changes of conditional classes and decision classes into the existing entropies. With these mechanisms, an attribute reduction algorithm is proposed for dynamic decision tables. When a part of data in a given data set is replaced by some new data, compared with the classic reduction algorithms based on the three entropies, the developed algorithm can find a new reduct in a much shorter time. Experiments on six data sets downloaded from UCI show that the algorithm is effective and efficient.

The rest of this paper is organized as follows. Some preliminaries in rough set theory are briefly reviewed in Section 2. Traditional heuristic reduction algorithms based on three representative entropies are introduced in Section 3. Section 4 presents the updating mechanisms of the three entropies for dynamically varying data values. In Section 5, based on the updating mechanisms, a reduction algorithm is proposed to compute reducts for dynamic data sets. In Section 6, six UCI data sets are employed to demonstrate effectiveness and efficiency of the proposed algorithm. Section 7 concludes this paper with some discussions.

2. Preliminary knowledge on rough sets

2.1. Basic concepts

This section reviews several basic concepts in rough set theory. Throughout this paper, the universe U is assumed a finite nonempty set.

An information system, as a basic concept in rough set theory, provides a convenient framework for the representation of objects in terms of their attribute values. An information system is a quadruple $S = (U, A, V, f)$, where U is a finite nonempty set of objects and is called the universe and A is a finite nonempty set of attributes, $V = \bigcup_{a\in A} V_a$ with V_a being the domain of a, and $f : U \times A \to V$ is an information function with $f(x, a) \in V_a$ for each $a \in A$ and $x \in U$. The system S can often be simplified as $S = (U, A)$.

Each nonempty subset $B \subseteq A$ determines an indiscernibility relation in the following way,

$$R_B = \{(x, y) \in U \times U \mid f(x, a) = f(y, a), \forall a \in B\}.$$

The relation R_B partitions U into some equivalence classes given by

$$U/R_B = \{[x]_B \mid x \in U\}, \text{ just } U/B,$$

where $[x]_B$ denotes the equivalence class determined by x with respect to B, i.e.,

$$[x]_B = \{y \in U \mid (x, y) \in R_B\}.$$

Given an equivalence relation R on the universe U and a subset $X \subseteq U$, one can define a lower approximation of X and an upper approximation of X by

$$\underline{R}X = \bigcup\{x \in U \mid [x]_R \subseteq X\}$$

and

$$\overline{R}X = \bigcup\{x \in U \mid [x]_R \cap X \neq \emptyset\},$$

respectively [39]. The order pair $(\underline{R}X, \overline{R}X)$ is called a rough set of X with respect to R. The positive region of X is denoted by $POS_R(X) = \underline{R}X$.

A partial relation $\preceq$ on the family $\{U/B \mid B \subseteq A\}$ is defined as follows [37]: $U/P \preceq U/Q$ (or $U/Q \succeq U/P$) if and only if, for every $P_i \in U/P$, there exists $Q_j \in U/Q$ such that $P_i \subseteq Q_j$, where $U/P = \{P_1, P_2, \cdots, P_m\}$ and $U/Q = \{Q_1, Q_2, \cdots, Q_n\}$ are partitions induced by $P,\ Q \subseteq A$, respectively. In this case, we say that Q is coarser than P, or P is finer than Q. If $U/P \preceq U/Q$ and $U/P \neq U/Q$, we say Q is strictly coarser than P (or P is strictly finer than Q), denoted by $U/P \prec U/Q$ (or $U/Q \succ U/P$).

It is clear that $U/P \prec U/Q$ if and only if, for every $X \in U/P$, there exists $Y \in U/Q$ such that $X \subseteq Y$, and there exist $X_0 \in U/P$ and $Y_0 \in U/Q$ such that $X_0 \subset Y_0$.

A decision table is an information system $S = (U, C \cup D)$ with $C \cap D = \emptyset$, where an element of C is called a condition attribute, C is called a condition attribute set, an element of D is called a decision attribute, and D is called a decision attribute set. Given $P \subseteq C$ and $U/D = \{D_1, D_2, \cdots, D_r\}$, the positive region of D with respect to the condition attribute set P is defined by $POS_P(D) = \bigcup_{k=1}^{r} \underline{P}D_k$.

For a decision table S and $P \subseteq C$, $X \in U/P$ is consistent iff all its objects have the same decision value; otherwise, X is inconsistent. The decision table S is called a consistent decision table iff $\forall X \in U/C$ are consistent; and if $\exists x, y \in U$ are inconsistent, then the table is called an inconsistent decision table. One can extract certain decision rules from a consistent decision table and uncertain decision rules from an inconsistent decision table.

For a decision table S and $P \subseteq C$, when a new object x is added to S, x is indistinguishable on B iff, $\exists y \in U$, $\forall a \in P$, such that $f(x, a) = f(y, a)$; and x is distinguishable on P iff, $\forall y \in U$, $\exists a \in P$ such that $f(x, a) \neq f(y, a)$.

2.2. Attribute reduction in rough set theory

Given an information system, all the attributes are not necessarily in the same importance, and some of them are irrelevant to the learning or recognition tasks. The concept of attribute reduction was first originated by Pawlak in [14, 15], which aimed to delete the irrelevant or redundant attributes on the condition of retaining the discernible ability of original attributes (the whole attributes set). The retained attribute subset got from attribute reduction is called a reduct.

Definition 1. *Let $S = \{U, A\}$ be an information system. Then $B \subseteq A$ is a reduct of S if*
(1) $U/B = U/A$ and
(2) $\forall a \in B, U/(B - \{a\}) \neq U/B$.

There are usually multiple reducts in a given information system, and the intersection of all reducts is called core. Given a decision table, the retained attribute subset got from attribute reduction is called a relative reduct, and the intersection of all relative reducts is called relative core [14, 15].

Definition 2. *Let $S = \{U, C \cup D\}$ be a decision table. Then $B \subseteq C$ is a relative reduct of S if*
(1) $POS_B(D) = POS_C(D)$ and
(2) $\forall a \in B, POS_{B-\{a\}}(D) \neq POS_B(D)$.

For these two definitions, the first condition guarantees that the reduct has the same distinguish power as the whole attribute set, and the second condition guarantees that there is no redundant attributes in the reduct. A reduct is called an exact reduct if it satisfies both of these two constraints, otherwise, is just an approximate reduct. In [40], Skowron proposed a discernibility matrix method to find all exact reducts of an information system without decision attributes.

However, it has been proved that using this algorithm to generate reducts is an NP-hard problem. For decision tables, Kryszkiewicz proposed an approach to computing the minimal set of attributes that functionally determine a decision attribute[41]. This algorithm can find an exact reduct of a given decision table. These two approaches are both very time-consuming.

As is well known, Pawlak's rough set model is applicable for the case that only nominal attributes exist in data sets. However, many real data in applications usually come with a complicated form. To conceptualize and analyze various types of data, researchers have generalized Pawlak's classic rough set model, and attribute reduction based on these generalizations was also redefined. Reducts generated by these reduction algorithms are usually approximate reducts. Ziarko provided the concept of β-reduct based on the introduction of variable precision rough set model (VPRS)[42]. VPRS deals with partial classification by introducing a probability value β. The β value represents a bound on the conditional probability of objects in a condition class which are classified to the same decision class. Yao proposed the decision-theoretic rough set model and also defined attribute reduction based on this generalized model[43, 44]. This model with loss functions aims to obtain optimization decisions by minimizing the overall risk with Bayesian decision procedures. An extensive review of multi-criteria decision analysis based on dominance rough sets was given by Greco et al. [45]. Dominance rough set model has also been applied for ordinal attribute reduction and multi-criteria classification [46, 47]. Dubois and Prade constructed the first fuzzy rough model by extending equivalence relation to fuzzy equivalence relation [48], where fuzzy equivalence relations satisfy reflexivity, symmetry and max-min transitivity. Reduction algorithms based on above generalized rough set models usually generate one or more approximation reducts and have been applied to solve their corresponding issues. It is deserved to point out that each kind of attribute reduction tries to preserve a particular property of a given table.

In addition, to save computational time of finding reduct, researchers have also developed many heuristic attribute reduction algorithms which can generate a single reduct from a given table [33, 35, 37, 38, 49]. Most of them are greedy and forward search algorithms, keeping selecting attributes with high significance until the dependence no longer increases. The reduct generated by a heuristic reduction algorithm is usually considered as an approximation reduct. It will be an exact reduct when deleting its redundant attributes.

However, the above analysis about generating an exact reduct and an approximation reduct is not very rigorous. For example, though reducts generated by heuristic reduction algorithms are considered as approximation reducts, some of them are often exact redcuts. Because so many reduction algorithms have been proposed in the last two decades and it is very difficult to list all of them here, this section just introduces a common distinction between generating an exact reduct and generating an approximation reduct.

3. Attribute reduction based on information entropy

Among various heuristic attribute reduction algorithms, reduction based on information entropy (or its variants) is a kind of common algorithm which has attracted much attention. There are three representative entropies which are used to construct reduction algorithms. They are complementary entropy[33], combination entropy[35] and Shannons information entropy[36]. The heuristic attribute reduction algorithms based on these three entropies are reviewed in this section.

In [33], the complementary entropy was introduced to measure uncertainty in rough set theory. Liang et al. also proposed the conditional complementary entropy to measure uncertainty of a decision table in [34]. By preserving the conditional entropy unchanged, the conditional complementary entropy was applied to construct reduction algorithms and reduce the redundant features in a decision table [35]. The conditional complementary entropy used in this algorithm is defined as follows[33, 34, 35].

Definition 3. *Let $S = (U, C \cup D)$ be a decision table and $B \subseteq C$. Then, one can obtain the condition partitions $U/B = \{X_1, X_2, \cdots, X_m\}$ and $U/D = \{Y_1, Y_2, \cdots, Y_n\}$. Based on these partitions, a conditional entropy of B relative to D is defined as*

$$E(D|B) = \sum_{i=1}^{m} \sum_{j=1}^{n} \frac{|X_i \cap Y_j|}{|U|} \frac{|Y_j^c - X_i^c|}{|U|}, \tag{1}$$

where Y_i^c and X_j^c are complement set of Y_i and X_j respectively.

Another information entropy, called combination entropy, was presented in [35] to measure the uncertainty of data tables. The conditional combination entropy was also introduced and can be used to construct the heuristic reduction algorithms [35]. This reduction algorithm can find a feature subset that possesses the same number of pairs of indistinguishable elements as that of the original decision table. The definition of the conditional combination entropy is defined as follows[35].

Definition 4. *Let $S = (U, C \cup D)$ be a decision table and $B \subseteq C$. Then one can obtain the condition partitions $U/B = \{X_1, X_2, \cdots, X_m\}$ and $U/D = \{Y_1, Y_2, \cdots, Y_n\}$. Based on these partitions, a conditional entropy of B relative to D is defined as*

$$CE(D|B) = \sum_{i=1}^{m}(\frac{|X_i|}{|U|}\frac{C^2_{|X_i|}}{C^2_{|U|}} - \sum_{j=1}^{n}\frac{|X_i \cap Y_j|}{|U|}\frac{C^2_{|X_i \cap Y_j|}}{C^2_{|U|}}). \tag{2}$$

where $C^2_{|X_i|}$ denotes the number of pairs of objects which are not distinguishable from each other in the equivalence class X_i.

Based on the classical rough set model, Shannon's information entropy[36] and its conditional entropy were also introduced to find a reduct in a heuristic algorithm [38, 50]. In [38], the reduction algorithm keeps the conditional entropy of target decision unchanged, and the conditional entropy is defined as follows[38].

Definition 5. *Let $S = (U, C \cup D)$ be a decision table and $B \subseteq C$. Then, one can obtain the condition partitions $U/B = \{X_1, X_2, \cdots, X_m\}$ and $U/D = \{Y_1, Y_2, \cdots, Y_n\}$. Based on these partitions, a conditional entropy of B relative to D is defined as*

$$H(D|B) = -\sum_{i=1}^{m}\frac{|X_i|}{|U|}\sum_{j=1}^{n}\frac{|X_i \cap Y_j|}{|X_i|}\log(\frac{|X_i \cap Y_j|}{|X_i|}). \tag{3}$$

For convenience, a uniform notation $ME(D|B)$ is introduced to denote these three entropies. For example, if one adopts Shannon's conditional entropy to define the attribute significance, then $ME(D|B) = H(D|B)$. Given a decision table $S = (U, C \cup D)$ and $B_1, B_2 \subseteq C$. According to literatures [33, 35, 38], if $U/B_1 \preceq U/B_2$, one can get that $ME(D|B_1) \leq ME(D|B_2)$. This conclusion indicates that, for a given decision table, as its condition classifications become finer, its entropies (the three entropies) are monotone decreasing. In addition, as the condition classifications become finer, the classified quality (see Definition 11) of the given decision table is monotone increasing. Thus, one can get that the three entropies of a given decision table are monotone decreasing with the classified quality increasing. Especially, when the classified quality is one, the entropies are zero [33, 35, 38].

The attribute significances based on entropies in a heuristic reduction algorithm is defined as follows (See Definitions 6-7) [33, 35, 38].

Definition 6. *Let $S = (U, C \cup D)$ be a decision table and $B \subseteq C$. $\forall a \in B$, the significance measure (inner significance) of a in B is defined as*

$$Sig^{inner}(a, B, D) = ME(D|B - \{a\}) - ME(D|B). \tag{4}$$

Definition 7. *Let $S = (U, C \cup D)$ be a decision table and $B \subseteq C$. $\forall a \in C - B$, the significance measure (outer significance) of a in B is defined as*

$$Sig^{outer}(a, B, D) = ME(D|B) - ME(D|B \cup \{a\}). \tag{5}$$

Given a decision table $S = (U, C \cup D)$ and $a \in C$. From the literatures [26-29], one can get that if $Sig^{inner}(a, C, D) > 0$, then the attribute a is indispensable, i.e., a is a core attribute of S. Based on core attributes, a heuristic attribute reduction algorithm can find a reduct by gradually adding selected attributes to the core. The definition of reduct based on information entropy is defined as follows [26-29].

Definition 8. *Let $S = (U, C \cup D)$ be a decision table and $B \subseteq C$. Then the attribute set B is a relative reduct if B satisfies:*

(1) $ME(D|B) = ME(D|C)$;

(2) $\forall a \in B$, $ME(D|B) \neq ME(D|B - \{a\})$.

Formally, the searching strategies in reduction algorithms based on the three entropies are similar to each other. The specific steps are written as follows [33, 35, 38].

Algorithm 1. Classic attribute reduction algorithm based on information entropy for a decision table (CAR)

Input: A decision table $S = (U, C \cup D)$
Output: Reduct *red*

Step 1: $red \leftarrow \emptyset$;
Step 2: $for\ (j = 1; j \leq |C|; j++)$
{ if $Sig^{inner}(a_j, C, D) > 0$, then $red \leftarrow red \cup \{a_j\}$;
}
Step 4: $P \leftarrow red$, *while* $(ME(D|P) \neq ME(D|C))$ *do*
{
Compute and select sequentially $Sig^{outer}(a_0, P, D) = max\{Sig^{outer}(a_i, P, D)\}, a_i \in C - P$;
$P \leftarrow P \cup \{a_0\}$;
}
Step 5: $red \leftarrow P$, return *red* and end.

Based on Definition 6, one can get core attributes according to steps 1-2 in this algorithm. Steps 3-4 add selected attributes to the core gradually, and then one can obtain a reduct of the given table. This algorithm can be considered as the common attribute reduction algorithm based on information entropy. The time complexity of CAR given in [37] is $O(|U||C|^2)$. However, this time complexity does not include the computational time of entropies. For a given decision table, computing entropies is a key step in above reduction algorithm, which is not computationally costless. Thus, to analyze the exact time complexity of above algorithm, the time complexity of computing entropies is given as well.

Given a decision table, according to Definitions 3-5, it first needs to compute the conditional classes and decision classes, respectively, and then computes the value of entropy. Xu et al. in [51] gave a fast algorithm for partition with time complexity being $O(|U||C|)$. So, the time complexity of computing entropy is

$$O(|U||C| + |U| + \sum_{i=1}^{m} |X_i| \cdot \sum_{j=1}^{n} |Y_j|) = O(|U||C| + |U| + |U||U|) = O(|U|^2),$$

where the specific introduction of m, n, X_i and Y_j is shown in Definitions 3-5. Thus, the time complexity of computing core (steps 1-2) is $O(|C||U|^2)$, and the time complexity of computing reduct according to CAR is

$$O(|C||U|^2 + |C|(|U||C| + |U|^2)) = O(|C|^2|U| + |C||U|^2).$$

4. Updating mechanism of information entropy

Given a dynamic decision table, based on the three representative entropies, this section presents the updating mechanisms of the three entropies for dynamically varying data values. As data values in a decision table vary with time, recomputing entropy is obviously time-consuming. To overcome this deficiency, the updating mechanisms derive new entropies by integrating the changes of conditional classes and decision classes into existing entropies. When data values of a single object vary, Theorems 1-4 introduce the updating mechanisms for the three entropies respectively.

For convenience, here are some explanations which will be used in the following theorems. Let $S = (U, C \cup D)$ be a decision table, $B \subseteq C$ and $x \in U$. $U/B = \{X_1, X_2, \cdots, X_m\}$, $U/D = \{Y_1, Y_2, \cdots, Y_n\}$, $x \in X_{p_1}$ and $x \in Y_{q_1}$

($p_1 \in \{1, 2, \cdots, m\}$ and $q_1 \in \{1, 2, \cdots, n\}$). If attribute values of x are varied, and here assumes that x is changed to x'. Let $U_{x'}$ denotes the new universe, one has $x' \in X'_{p_2}$ and $x' \in Y'_{q_2}$ ($X'_{p_2} \in U_{x'}/B$ and $Y'_{q_2} \in U_{x'}/D$). Obviously, one can get that $X'_{p_2} - \{x\} \in U/B$, $Y'_{q_2} - \{x\} \in U/D$, $X_{p_1} - \{x\} \in U_{x'}/B$ and $Y_{q_1} - \{x\} \in U_{x'}/D$. In addition, X'_{p_1}, Y'_{q_1}, X_{p_2} and Y_{q_2} denote $X_{p_1} - \{x\}$, $Y_{q_1} - \{x\}$, $X'_{p_2} - \{x\}$ and $Y'_{q_2} - \{x\}$, respectively.

Theorem 1. *Let $S = (U, C \cup D)$ be a decision table and $B \subseteq C$. The complementary conditional entropy of D with respect to B is $E_U(D|B)$. Then, one can obtain the partitions $U/B = \{X_1, X_2, \cdots, X_m\}$ and $U/D = \{Y_1, Y_2, \cdots, Y_n\}$. $x \in X_{p_1}$ and $x \in Y_{q_1}$. If one and only object $x \in U$ is changed to x', then $x' \in X'_{p_2}$ and $x' \in Y'_{q_2}$ ($X'_{p_2} \in U_{x'}/B$ and $Y'_{q_2} \in U_{x'}/D$). The new conditional complementary entropy becomes*

$$E_{U_{x'}}(D|B) = E_U(D|B) + \frac{2|X'_{p_2} - Y'_{q_2}| - 2|X'_{p_1} - Y'_{q_1}|}{|U|^2},$$

where $X'_{p_1} = X_{p_1} - \{x\}$ and $Y'_{q_1} = Y_{q_1} - \{x\}$.

Proof. For the decision table S, when x is changed to x', there are four situations about the changes of conditional classes and decision classes, which are as follows:

(a) $U_{x'}/B = \{X_1, X_2, \cdots, X'_{p_1}, \cdots, X_m, \{x'\}\}$ and $U_{x'}/D = \{Y_1, Y_2, \cdots, Y'_{q_1}, \cdots, Y_n, \{x'\}\}$;
(b) $U_{x'}/B = \{X_1, X_2, \cdots, X'_{p_1}, \cdots, X'_{p_2}, \cdots, X_m, \}$, $x' \in X'_{p_2}$, and $U_{x'}/D = \{Y_1, Y_2, \cdots, Y'_{q_1}, \cdots, Y_n, \{x'\}\}$;
(c) $U_{x'}/B = \{X_1, X_2, \cdots, X'_{p_1}, \cdots, X_m, \{x'\}\}$, $U_{x'}/D = \{Y_1, Y_2, \cdots, Y'_{q_1}, \cdots, Y'_{q_2}, \cdots, Y_n\}$ and $x' \in Y'_{q_2}$;
(d) $U_{x'}/B = \{X_1, X_2, \cdots, X'_{p_1}, \cdots, X'_{p_2}, \cdots, X_m, \}$, $x' \in X'_{p_2}$ and $U_{x'}/D = \{Y_1, Y_2, \cdots, Y'_{q_1}, \cdots, Y'_{q_2}, \cdots, Y_n\}$, $x' \in Y'_{q_2}$.

For convenience, here introduces a uniform notation about these four situations.

Let $U/B = \{X_1, X_2, \cdots, X_m, X_{m+1}\}$ and $X_{m+1} = \emptyset$. Then, we have $U_{x'}/B = \{X_1, X_2, \cdots, X'_{p_1}, \cdots, X'_{p_2}, \cdots, X_{m+1}, \}$, $X'_{p_1} = X_{p_1} - \{x'\}$ and $X'_{p_2} = \{x'\} \cup X_{p_2}$. Similarly, let $Y_{n+1} = \emptyset$, we can get that $U/D = \{Y_1, Y_2, \cdots, Y_{n+1}\}$, $U_{x'}/D = \{Y_1, Y_2, \cdots, Y'_{q_1}, \cdots, Y'_{q_2}, \cdots, Y_{n+1}\}$ and $Y'_{q_2} = \{x'\} \cup Y_{q_2}$. Obviously, for situation (a), we have $X'_{p_2} = X_{m+1} \cup \{x'\} = \emptyset \cup \{x'\} = \{x'\}$ and $Y'_{q_2} = Y_{n+1} \cup \{x'\} = \{x'\}$. Similarly, for situation (b), we have $Y'_{q_2} = Y_{n+1} \cup \{x'\} = \{x'\}$. And for situation (c), we have $X'_{p_2} = \{x'\}$.

According to the uniform notation, the updating mechanism of complementary conditional entropy is as follows.

$$\begin{aligned}
&E(D|B)\\
&= \sum_{i=1}^{m}\sum_{j=1}^{n} \frac{|X_i \cap Y_j|}{|U|}\frac{|Y_j^c - X_i^c|}{|U|}\\
&= \sum_{i=1}^{m}\sum_{j=1}^{n} \frac{|X_i \cap Y_j|}{|U|}\frac{|X_i - Y_j|}{|U|}.
\end{aligned}$$

And

$$\begin{aligned}
&E_{U_{x'}}(D|B)\\
&= \sum_{i=1}^{m-1}(\sum_{j=1}^{n-1} \frac{|X_i \cap Y_j||X_i - Y_j|}{|U|^2} + \frac{|X_i \cap Y'_{q_1}||X_i - Y'_{q_1}|}{|U|^2} + \frac{|X_i \cap Y'_{q_2}||X_i - Y'_{q_2}|}{|U|^2}) + \sum_{j=1}^{n-1} \frac{|X'_{p_1} \cap Y_j||X'_{p_1} - Y_j|}{|U|^2} + \frac{|X'_{p_1} \cap Y'_{q_1}||X'_{p_1} - Y'_{q_1}|}{|U|^2} + \frac{|X'_{p_1} \cap Y'_{q_2}||X'_{p_1} - Y'_{q_2}|}{|U|^2} +\\
&\quad \sum_{j=1}^{n-1} \frac{|X'_{p_2} \cap Y_j||X'_{p_2} - Y_j|}{|U|^2} + \frac{|X'_{p_2} \cap Y'_{q_1}||X'_{p_2} - Y'_{q_1}|}{|U|^2} + \frac{|X'_{p_2} \cap Y'_{q_2}||X'_{p_2} - Y'_{q_2}|}{|U|^2}\\
&= \sum_{i=1}^{m-1}(\sum_{j=1}^{n-1} \frac{|X_i \cap Y_j||X_i - Y_j|}{|U|^2} + \frac{|X_i \cap Y_{q_1}||X_i - Y_{q_1}|}{|U|^2} + \frac{|X_i \cap Y_{q_2}||X_i - Y_{q_2}|}{|U|^2}) + \sum_{j=1}^{n-1} \frac{|X_{p_1} \cap Y_j| \cdot (|X_{p_1} - Y_j| - 1)}{|U|^2} + \frac{(|X_{p_1} \cap Y_{q_1}| - 1) \cdot |X_{p_1} - Y_{q_1}|}{|U|^2} + \frac{|X_{p_1} \cap Y_{q_2}| \cdot (|X_{p_1} - Y_{q_2}| - 1)}{|U|^2}\\
&\quad + \sum_{j=1}^{n-1} \frac{|X_{p_2} \cap Y_j| \cdot (|X_{p_2} - Y_j| + 1)}{|U|^2} + \frac{|X_{p_2} \cap Y_{q_1}| \cdot (|X_{p_2} - Y_{q_1}| + 1)}{|U|^2} + \frac{(|X_{p_2} \cap Y_{q_2}| + 1) \cdot |X_{p_2} - Y_{q_2}|}{|U|^2}\\
&= \sum_{i=1}^{m+1}\sum_{j=1}^{n+1} \frac{|X_i \cap Y_j||X_i - Y_j|}{|U|^2} + \frac{|X_{p_2} - Y_{q_2}| - |X_{p_1} - Y_{q_1}|}{|U|^2} + \sum_{j=1, j\neq q_2}^{n+1} \frac{|X_{p_2} \cap Y_j|}{|U|^2} - \sum_{j=1, j\neq q_1}^{n+1} \frac{|X_{p_1} \cap Y_j|}{|U|^2}\\
&= E_U(D|B) + \frac{|X'_{p_2} - Y'_{q_2}| - |X'_{p_1} - Y'_{q_1}|}{|U|^2} + \sum_{j=1, j\neq q_2}^{n+1} \frac{|X'_{p_2} \cap Y_j|}{|U|^2} - \sum_{j=1, j\neq q_1}^{n+1} \frac{|X'_{p_1} \cap Y_j|}{|U|^2}\\
&= E_U(D|B) + \frac{|X'_{p_2} - Y'_{q_2}| - |X'_{p_1} - Y'_{q_1}|}{|U|^2} + \sum_{j=1, j\neq q_2}^{n+1} \frac{|X'_{p_2} \cap Y_j|}{|U|^2} + \frac{|X'_{p_2} \cap Y'_{q_2}|}{|U|^2} - \frac{|X'_{p_2} \cap Y'_{q_2}|}{|U|^2} - \sum_{j=1, j\neq q_1}^{n+1} \frac{|X'_{p_1} \cap Y_j|}{|U|^2} - \frac{|X'_{p_1} \cap Y'_{q_1}|}{|U|^2} + \frac{|X'_{p_1} \cap Y'_{q_1}|}{|U|^2}\\
&= E_U(D|B) + \frac{|X'_{p_2} - Y'_{q_2}| - |X'_{p_1} - Y'_{q_1}|}{|U|^2} + \frac{|X'_{p_2}| - |X'_{p_2} \cap Y'_{q_2}|}{|U|^2} - \frac{|X'_{p_1}| - |X'_{p_1} \cap Y'_{q_1}|}{|U|^2}\\
&= E_U(D|B) + \frac{2|X'_{p_2} - Y'_{q_2}| - 2|X'_{p_1} - Y'_{q_1}|}{|U|^2}. \text{ This completes the proof. } \square
\end{aligned}$$

For convenience of introducing combination entropy, here gives a deformation of the definition of combination entropy (see Definition 2). According to $C_N^2 = \frac{N(N-1)}{2}$, the following deformation can be got. Then the updating mechanism of combination conditional entropy is introduced on the basis of this deformation.

Definition 9. *Let $S = (U, C \cup D)$ be a decision table and $B \subseteq C$. One can obtain the condition partition $U/B = \{X_1, X_2, \cdots, X_m\}$ and $U/D = \{Y_1, Y_2, \cdots, Y_n\}$. The combination conditional entropy of B relative to D is defined as*

$$CE(D|B) = \sum_{i=1}^{m}(\frac{|X_i|^2(|X_i|-1)}{|U|^2(|U|-1)} - \sum_{j=1}^{n}\frac{|X_i \cap Y_j|^2(|X_i \cap Y_j|-1)}{|U|^2(|U|-1)}). \tag{6}$$

Theorem 2. *Let $S = (U, C \cup D)$ be a decision table and $B \subseteq C$. The combination conditional entropy of D with respect to B is $E_U(D|B)$. Then, one can obtain the partitions $U/B = \{X_1, X_2, \cdots, X_m\}$ and $U/D = \{Y_1, Y_2, \cdots, Y_n\}$. $x \in X_{p_1}$ and $x \in Y_{q_1}$. If one and only object $x \in U$ is changed to x', then $x' \in X'_{p_2}$ and $x' \in Y'_{q_2}$ ($X'_{p_2} \in U_{x'}/B$ and $Y'_{q_2} \in U_{x'}/D$). The new combination conditional entropy becomes*

$$CE_{U_{x'}}(D|B) = CE_U(D|B) + \Delta,$$

where $\Delta = \frac{|X'_{p_2} - Y'_{q_2}|(3|X'_{p_2}|+3|X'_{p_2}\cap Y'_{q_2}|-5) - |X'_{p_1} - Y'_{q_1}|(3|X'_{p_1}|+3|X'_{p_1}\cap Y'_{q_1}|+1)}{|U|^2}$, $X'_{p_1} = X_{p_1} - \{x\}$ *and* $Y'_{q_1} = Y_{q_1} - \{x\}$.

Proof. Similarly, when x is added to S, there are four same situations as the proof in Theorem 1. Then, the combination conditional entropy is

$$\begin{aligned}
&CE_{U_{x'}}(D|B)\\
&= \sum_{i=1}^{m-1}(\frac{|X_i|^2(|X_i|-1)}{|U|^2(|U|-1)} - \sum_{j=1}^{n-1}\frac{|X_i\cap Y_j|^2(|X_i\cap Y_j|-1)}{|U|^2(|U|-1)} - \frac{|X_i\cap Y'_{q_1}|^2(|X_i\cap Y'_{q_1}|-1)}{|U|^2(|U|-1)} - \frac{|X_i\cap Y'_{q_2}|^2(|X_i\cap Y'_{q_2}|-1)}{|U|^2(|U|-1)}) + \frac{|X'_{p_1}|^2(|X'_{p_1}|-1)}{|U|^2(|U|-1)} - \sum_{j=1}^{n-1}\frac{|X'_{p_1}\cap Y_j|^2(|X'_{p_1}\cap Y_j|-1)}{|U|^2(|U|-1)}\\
&\quad - \frac{|X'_{p_1}\cap Y'_{q_1}|^2(|X'_{p_1}\cap Y'_{q_1}|-1)}{|U|^2(|U|-1)} - \frac{|X'_{p_1}\cap Y'_{q_2}|^2(|X'_{p_1}\cap Y'_{q_2}|-1)}{|U|^2(|U|-1)} + \frac{|X'_{p_2}|^2(|X'_{p_2}|-1)}{|U|^2(|U|-1)} - \sum_{j=1}^{n-1}\frac{|X'_{p_2}\cap Y_j|^2(|X'_{p_2}\cap Y_j|-1)}{|U|^2(|U|-1)} - \frac{|X'_{p_2}\cap Y'_{q_1}|^2(|X'_{p_2}\cap Y'_{q_1}|-1)}{|U|^2(|U|-1)}\\
&\quad - \frac{|X'_{p_2}\cap Y'_{q_2}|^2(|X'_{p_2}\cap Y'_{q_2}|-1)}{|U|^2(|U|-1)}\\
&= \sum_{i=1}^{m-1}(\frac{|X_i|^2(|X_i|-1)}{|U|^2(|U|-1)} - \sum_{j=1}^{n+1}\frac{|X_i\cap Y_j|^2(|X_i\cap Y_j|-1)}{|U|^2(|U|-1)}) + \frac{(|X_{p_1}|-1)^2(|X_{p_1}|-2)}{|U|^2(|U|-1)} - \sum_{j=1}^{n-1}\frac{|X_{p_1}\cap Y_j|^2(|X_{p_1}\cap Y_j|-1)}{|U|^2(|U|-1)} - \frac{(|X_{p_1}\cap Y_{q_1}|-1)^2(|X_{p_1}\cap Y_{q_1}|-2)}{|U|^2(|U|-1)} -\\
&\quad \frac{|X_{p_1}\cap Y_{q_2}|^2(|X_{p_1}\cap Y_{q_2}|-1)}{|U|^2(|U|-1)} + \frac{(|X_{p_2}|+1)^2|X_{p_2}|}{|U|^2(|U|-1)} - \sum_{j=1}^{n-1}\frac{|X_{p_2}\cap Y_j|^2(|X_{p_2}\cap Y_j|-1)}{|U|^2(|U|-1)} - \frac{|X_{p_2}\cap Y_{q_1}|^2(|X_{p_2}\cap Y_{q_1}|-1)}{|U|^2(|U|-1)} - \frac{(|X_{p_2}\cap Y_{q_2}|+1)^2|X_{p_2}\cap Y_{q_2}|}{|U|^2(|U|-1)}\\
&= \sum_{i=1}^{m+1}(\frac{|X_i|^2(|X_i|-1)}{|U|^2(|U|-1)} - \sum_{j=1}^{n+1}\frac{|X_i\cap Y_j|^2(|X_i\cap Y_j|-1)}{|U|^2(|U|-1)}) + \frac{-3|X_{p_1}|^2+5|X_{p_1}|+3|X_{p_1}\cap Y_{q_1}|^2-5|X_{p_1}\cap Y_{q_1}|}{|U|^2(|U|-1)} + \frac{3|X_{p_2}|^2+|X_{p_2}|-3|X_{p_2}\cap Y_{q_2}|^2-|X_{p_2}\cap Y_{q_2}|}{|U|^2(|U|-1)}\\
&= CE_U(D|B) + \frac{|X_{p_2}-Y_{q_2}|(3|X_{p_2}|+3|X_{p_2}\cap Y_{q_2}|+1)}{|U|^2(|U|-1)} - \frac{|X_{p_1}-Y_{q_1}|(3|X_{p_1}|+3|X_{p_1}\cap Y_{q_1}|-5)}{|U|^2(|U|-1)}.
\end{aligned}$$

And because $X'_{p_2} = X_{p_2} \cup \{x\}$ and $Y'_{q_2} = Y_{q_2} \cup \{x\}$, from Theorem 2, one can also get that

$CE_{U_{x'}}(D|B)$

$= CE_U(D|B) + \frac{|X_{p_2}-Y_{q_2}|(3|X_{p_2}|+3|X_{p_2}\cap Y_{q_2}|+1)}{|U|^2(|U|-1)} - \frac{|X_{p_1}-Y_{q_1}|(3|X_{p_1}|+3|X_{p_1}\cap Y_{q_1}|-5)}{|U|^2(|U|-1)}$. This completes the proof. □

The following two theorems are the updating mechanisms of Shannon's conditional entropy shown in Definition 3.

Theorem 3. *Let $S = (U, C \cup D)$ be a decision table and $B \subseteq C$. The Shannon's conditional entropy of D with respect to B is $E_U(D|B)$. Then, one can obtain the partitions $U/B = \{X_1, X_2, \cdots, X_m\}$ and $U/D = \{Y_1, Y_2, \cdots, Y_n\}$. $x \in X_{p_1}$ and $x \in Y_{q_1}$. If one and only object $x \in U$ is changed to x', then $x' \in X'_{p_2}$ and $x' \in Y'_{q_2}$ ($X'_{p_2} \in U_{x'}/B$ and $Y'_{q_2} \in U_{x'}/D$). The new Shannon's conditional entropy becomes*

$$H_{U_{x'}}(D|B) = H_U(D|B) - \Delta,$$

where $\Delta = \sum_{j=1,j\neq q_1}^{n}\frac{|X_{p_1}\cap Y_j|}{|U|}log\frac{|X_{p_1}|}{|X'_{p_1}|} + \sum_{j=1,j\neq q_2}^{n}\frac{|X_{p_2}\cap Y_j|}{|U|}log\frac{|X_{p_2}|}{|X'_{p_2}|} + \frac{|X_{p_1}\cap Y_{q_1}|}{|U|}\cdot log\frac{|X'_{p_1}\cap Y'_{q_1}||X_{p_1}|}{|X_{p_1}\cap Y_{q_1}||X'_{p_1}|} + \frac{|X_{p_2}\cap Y_{q_2}|}{|U|}log\frac{|X'_{p_2}\cap Y'_{q_2}||X_{p_2}|}{|X_{p_2}\cap Y_{q_2}||X'_{p_2}|} + \frac{1}{|U|}log\frac{|X'_{p_1}||X'_{p_2}\cap Y'_{q_2}|}{|X'_{p_2}||X'_{p_1}\cap Y'_{q_1}|}$, $X'_{p_1} = X_{p_1} - \{x\}$ *and* $Y'_{q_1} = Y_{q_1} - \{x\}$.

Proof. Similar to the proof in Theorem 4, it can be easily proved. □

In view of that the formula of Δ is complicated, thus, for the large-scale data tables, an approximate computational formula is proposed in the following theorem.

Theorem 4. *Let $S = (U, C \cup D)$ be a large-scale decision table and $B \subseteq C$. The conditional Shannon's entropy of D with respect to B is $E_U(D|B)$. Then, one can obtain the partitions $U/B = \{X_1, X_2, \cdots, X_m\}$ and $U/D = \{Y_1, Y_2, \cdots, Y_n\}$. $x \in X_{p_1}$ and $x \in Y_{q_1}$. If one and only object $x \in U$ is changed to x', then $x' \in X'_{p_2}$ and $x' \in Y'_{q_2}$ ($X'_{p_2} \in U_{x'}/B$ and $Y'_{q_2} \in U_{x'}/D$). The new Shannon's conditional entropy becomes*

$$H_{U_{x'}}(D|B) \approx H_U(D|B) - \frac{1}{|U|} log \frac{|X'_{p_1}||X'_{p_2} \cap Y'_{q_2}|}{|X'_{p_2}||X'_{p_1} \cap Y'_{q_1}|},$$

where $X'_{p_1} = X_{p_1} - \{x\}$ and $Y'_{q_1} = Y_{q_1} - \{x\}$.

Proof. Similarly, when x is added to S, there are four same situations as the proof in Theorem 1. Then, the Shannon's conditional entropy is

$$\begin{aligned}
&H_{U_{x'}}(D|B)\\
&= -(\sum_{i=1}^{m-1} \frac{|X_i|}{|U|}(\sum_{j=1}^{n-1} \frac{|X_i \cap Y_j|}{|X_i|} log \frac{|X_i \cap Y_j|}{|X_i|} + \frac{|X_i \cap Y'_{q_1}|}{|X_i|} log \frac{|X_i \cap Y'_{q_1}|}{|X_i|} + \frac{|X_i \cap Y'_{q_2}|}{|X_i|} log \frac{|X_i \cap Y'_{q_2}|}{|X_i|}) + \frac{|X'_{p_1}|}{|U|}(\sum_{j=1}^{n-1} \frac{|X'_{p_1} \cap Y_j|}{|X'_{p_1}|} log \frac{|X'_{p_1} \cap Y_j|}{|X'_{p_1}|} + \frac{|X'_{p_1} \cap Y'_{q_1}|}{|X'_{p_1}|} \cdot\\
&\quad log \frac{|X'_{p_1} \cap Y'_{q_1}|}{|X'_{p_1}|} + \frac{|X'_{p_1} \cap Y'_{q_2}|}{|X'_{p_1}|} \cdot log \frac{|X'_{p_1} \cap Y'_{q_2}|}{|X'_{p_1}|}) + \frac{|X'_{p_2}|}{|U|}(\sum_{j=1}^{n-1} \frac{|X'_{p_2} \cap Y_j|}{|X'_{p_2}|} log \frac{|X'_{p_2} \cap Y_j|}{|X'_{p_2}|} + \frac{|X'_{p_2} \cap Y'_{q_1}|}{|X'_{p_2}|} log \frac{|X'_{p_2} \cap Y'_{q_1}|}{|X'_{p_2}|} + \frac{|X'_{p_2} \cap Y'_{q_2}|}{|X'_{p_2}|} \cdot log \frac{|X'_{p_2} \cap Y'_{q_2}|}{|X'_{p_2}|}))\\
&= -(\sum_{i=1}^{m-1} \frac{|X_i|}{|U|}(\sum_{j=1}^{n+1} \frac{|X_i \cap Y_j|}{|X_i|} log \frac{|X_i \cap Y_j|}{|X_i|}) + \frac{|X_{p_1}|}{|U|} \sum_{j=1}^{n-1} \frac{|X_{p_1} \cap Y_j|}{|X_{p_1}|} log \frac{|X_{p_1} \cap Y_j|}{|X'_{p_1}|} + \frac{|X_{p_1}|}{|U|} \cdot \frac{|X_{p_1} \cap Y_{q_1}|-1}{|X_{p_1}|} log \frac{|X'_{p_1} \cap Y'_{q_1}|}{|X'_{p_1}|} + \frac{|X_{p_1}|}{|U|} \cdot \frac{|X_{p_1} \cap Y_{q_2}|}{|X_{p_1}|} \cdot log \frac{|X'_{p_1} \cap Y'_{q_2}|}{|X'_{p_1}|}\\
&\quad + \frac{|X_{p_2}|}{|U|} \sum_{j=1}^{n-1} \frac{|X_{p_2} \cap Y_j|}{|X_{p_2}|} \cdot log \frac{|X_{p_2} \cap Y_j|}{|X'_{p_2}|} + \frac{|X_{p_2}|}{|U|} \cdot \frac{|X_{p_2} \cap Y_{q_1}|}{|X_{p_2}|} log \frac{|X_{p_2} \cap Y_{q_1}|}{|X'_{p_2}|} + \frac{|X_{p_2}|}{|U|} \cdot \frac{|X_{p_2} \cap Y_{q_2}|+1}{|X_{p_2}|} \cdot log \frac{|X'_{p_2} \cap Y'_{q_2}|}{|X'_{p_2}|})\\
&= -(\sum_{i=1}^{m-1} \frac{|X_i|}{|U|} \sum_{j=1}^{n+1} \frac{|X_i \cap Y_j|}{|X_i|} log \frac{|X_i \cap Y_j|}{|X_i|} + \frac{|X_{p_1}|}{|U|} \sum_{j=1}^{n-1} \frac{|X_{p_1} \cap Y_j|}{|X_{p_1}|}(log \frac{|X_{p_1} \cap Y_j|}{|X_{p_1}|} + log \frac{|X_{p_1}|}{|X'_{p_1}|}) + \frac{|X_{p_1}|}{|U|} \cdot \frac{|X_{p_1} \cap Y_{q_1}|}{|X_{p_1}|}(log \frac{|X_{p_1} \cap Y_{q_1}|}{|X_{p_1}|} + log \frac{|X'_{p_1} \cap Y'_{q_1}||X_{p_1}|}{|X_{p_1} \cap Y_{q_1}||X'_{p_1}|})\\
&\quad - \frac{1}{|U|} log \frac{|X'_{p_1} \cap Y'_{q_1}|}{|X'_{p_1}|} + \frac{|X_{p_1}|}{|U|} \cdot \frac{|X_{p_1} \cap Y_{q_2}|}{|X_{p_1}|} \cdot (log \frac{|X_{p_1} \cap Y_{q_2}|}{|X_{p_1}|} + log \frac{|X_{p_1}|}{|X'_{p_1}|}) + \frac{|X_{p_2}|}{|U|} \sum_{j=1}^{n-1} \frac{|X_{p_2} \cap Y_j|}{|X_{p_2}|} \cdot (log \frac{|X_{p_2} \cap Y_j|}{|X_{p_2}|} + log \frac{|X_{p_2}|}{|X'_{p_2}|}) + \frac{|X_{p_2}|}{|U|} \cdot \frac{|X_{p_2} \cap Y_{q_1}|}{|X_{p_2}|}(log \frac{|X_{p_2} \cap Y_{q_1}|}{|X_{p_2}|}\\
&\quad + log \frac{|X_{p_2}|}{|X'_{p_2}|}) + \frac{|X_{p_2}|}{|U|} \cdot \frac{|X_{p_2} \cap Y_{q_2}|}{|X_{p_2}|} \cdot (log \frac{|X_{p_2} \cap Y_{q_2}|}{|X_{p_2}|} + log \frac{|X'_{p_2} \cap Y'_{q_2}||X_{p_2}|}{|X_{p_2} \cap Y_{q_2}||X'_{p_2}|}) + \frac{1}{|U|} log \frac{|X'_{p_2} \cap Y'_{q_2}|}{|X'_{p_2}|}).
\end{aligned}$$

To simplify the calculations of above formula, for the large-scale decision tables, here are some approximated expressions. In view of that $|X_{p_1}|$ and $|X_{p_2}|$ based on the large-scale decision tables are relatively large, respectively, one can get that $|X_{p_1}| \approx |X'_{p_1}|$ and $|X_{p_2}| \approx |X'_{p_2}|$ ($X'_{p_1} = X_{p_1} - \{x\}$ and $X'_{p_2} = X_{p_2} \cup \{x\}$), e.t. $log \frac{|X_{p_1}|}{|X'_{p_1}|} \approx 0$ and $log \frac{|X_{p_2}|}{|X'_{p_2}|} \approx 0$. Similarly, one can also get from $|X_{p_1} \cap Y_{q_1}| \approx |X'_{p_1} \cap Y'_{q_1}|$ and $|X_{p_2} \cap Y_{q_2}| \approx |X'_{p_2} \cap Y'_{q_2}|$ that $log \frac{|X'_{p_1} \cap Y'_{q_1}||X_{p_1}|}{|X_{p_1} \cap Y_{q_1}||X'_{p_1}|} \approx 0$ and $log \frac{|X'_{p_2} \cap Y'_{q_2}||X_{p_2}|}{|X_{p_2} \cap Y_{q_2}||X'_{p_2}|} \approx 0$. Hence, the above formula can be simplified to

$$\begin{aligned}
&H_{U_{x'}}(D|B)\\
&\approx -(\sum_{i=1}^{m} \frac{|X_i|}{|U|} \sum_{j=1}^{n} \frac{|X_i \cap Y_j|}{|X_i|} log \frac{|X_i \cap Y_j|}{|X_i|} - \frac{1}{|U|} log \frac{|X'_{p_1} \cap Y'_{q_1}|}{|X'_{p_1}|} + \frac{1}{|U|} \cdot log \frac{|X'_{p_2} \cap Y'_{q_2}|}{|X'_{p_2}|})\\
&= H_U(D|B) - \frac{1}{|U|} log \frac{|X'_{p_2} \cap Y'_{q_2}||X'_{p_1}|}{|X'_{p_2}||X'_{p_1} \cap Y'_{q_1}|}.
\end{aligned}$$

Thus, this completes the proof. □

5. Attribute reduction algorithm for decision tables with dynamically varying attribute values

Based on the updating mechanisms of the three entropies, this section introduces an attribute reduction algorithm based on information entropy for decision tables with dynamically varying attribute values. In view of that core is another key concept besides redcut in rough set theory [14, 15], this section also gives an algorithm for core computation. In rough set theory, core is the intersection of all reducts of a given table, and core attributes are considered as the indispensable attributes in a reduct. Note that, for the three entropies, the following algorithms are commonly used to update core and reduct.

Algorithm 2. An algorithm to core computation for a dynamic decision table ($ACORE_{x'}$)

Input: A decision table $S = (U, C \cup D)$ and object $x \in U$ is changed to x'
Output: Core attribute $CORE_{x'}$ on $U_{x'}$

Step 1 : Find X'_{p_1} and X'_{p_2}: in $U/C = \{X_1, X_2, \cdots, X_m\}$ and $x \in X_{p_1}$. If x is changed to x', and $x' \in X'_{p_2}$. One have $X'_{p_1} = X_{p_1} - \{x'\}$ and $U_{x'}/C = \{X_1, X_2, \cdots, X'_{p_1}, \cdots, X'_{p_2}, \cdots, X_m\}$.

Step 2 : Find Y'_{q_1} and Y'_{q_2}: in $U/D = \{Y_1, Y_2, \cdots, Y_n\}$ and $x \in Y_{q_1}$. If x is changed to x', and $x' \in Y'_{q_2}$. We have $Y'_{q_1} = Y_{q_1} - \{x'\}$ and $U_{x'}/D = \{Y_1, Y_2, \cdots, Y'_{q_1}, \cdots, Y'_{q_2}, \cdots, Y_n\}$.

Step 3 : Compute $ME_{U_{x'}}(D|C)$ (according to Theorems 1, 2 or 4);

Step 4 : $CORE_{U_{x'}} \leftarrow \emptyset$, for each $a \in C$

1) In $U/(C - \{a\}) = \{M_1, M_2, \cdots, M_{m'}\}$ ($m' \leq m$) and $x \in M_{t_1}$. If x is changed to x', one have $x' \in M'_{t_2}$, $M'_{t_1} = M_{t_1} - \{x'\}$ and $U_{x'}/(C - \{a\}) = \{M_1, M_2, \cdots, M'_{t_1}, \cdots, M'_{t_2}, \cdots, M_{m'}\}$.

2) Compute $ME_{U_{x'}}(D|C - \{a\})$ (according to Theorems 1, 2 or 4).

3) If $ME_{U_{x'}}(D|C - \{a\}) \neq ME_{U_{x'}}(D|C)$, then $CORE_{U_{x'}} = CORE_{U_{x'}} \cup \{a\}$.

Step 5 : Return $CORE_{U_{x'}}$ and end.

Based on updating mechanisms of the three entropies, an attribute reduction algorithm for decision tables with dynamically varying attribute values is introduced in the following. In this algorithm, the existing reduction result is one of inputs, which is used to find its new reduct after data changes.

Algorithm 3. An algorithm to reduct computation for a dynamic decision table ($ARED_{x'}$)

Input: A decision table $S = (U, C \cup D)$, reduct RED_U on U, and the changed object x which is changed to x'

Output: Attribute reduct $RED_{U_{x'}}$ on $U_{x'}$

Step 1 : Find M'_{t_1} and M'_{t_2}: in $U/B = \{M_1, M_2, \cdots, M_{m'}\}$ and $x \in M_{t_1}$. If x is changed to x', and $x' \in M'_{t_2}$. One have $M'_{t_1} = M_{t_1} - \{x'\}$ and $U_{x'}/B = \{M_1, M_2, \cdots, M'_{t_1}, \cdots, M'_{t_2}, \cdots, M_{m'}\}$.

Step 2 : Find Y'_{q_1} and Y'_{q_2}: in $U/D = \{Y_1, Y_2, \cdots, Y_n\}$ and $x \in Y_{q_1}$. If x is changed to x', and $x' \in Y'_{q_2}$. One have $Y'_{q_1} = Y_{q_1} - \{x'\}$ and $U_{x'}/D = \{Y_1, Y_2, \cdots, Y'_{q_1}, \cdots, Y'_{q_2}, \cdots, Y_n\}$.

Step 3 : Compute $ME_{U_{x'}}(D|B)$ (according to Theorems 1, 2 or 4);

Step 4 : Find X'_{p_1} and X'_{p_2}: in $U/C = \{X_1, X_2, \cdots, X_m\}$ and $x \in X_{p_1}$. If x is changed to x', and $x' \in X'_{p_2}$. One have $X'_{p_1} = X_{p_1} - \{x'\}$ and $U_{x'}/C = \{X_1, X_2, \cdots, X'_{p_1}, \cdots, X'_{p_2}, \cdots, X_m\}$.

Step 5 : Compute $ME_{U_{x'}}(D|C)$ (according to Theorems 1, 2 or 4);

Step 6 : If $ME_{U_{x'}}(D|B) = ME_{U_{x'}}(D|C)$, then $RED_{U_{x'}} \leftarrow RED_U$, turn to *Step 8*; else turn to *Step 7*.

Step 7 : $B \leftarrow RED_U$, *while* $ME_{U_{x'}}(D|B) \neq ME_{U_{x'}}(D|C)$ *do*

{ For each $a \in C - B$, compute $Sig^{outer}_{U_{x'}}(a, B, D)$ (according to Theorems 1, 2 or 4 and Definition 6);

Select $a_0 = max\{Sig^{outer}_{U_{x'}}(a, B, D)\}$, $a \in C - B$;

$B \leftarrow B \cup \{a_0\}$.

}

Step 8 : For each $a \in B$ *do*

{ Compute $Sig^{inner}_{U_{x'}}(a, B, D)$;

If $Sig^{inner}_{U_{x'}}(a, B, D) = 0$, then $B \leftarrow B - \{a\}$. }

Step 9 : $RED_{U_{x'}} \leftarrow B$, return $RED_{U_{x'}}$ and end.

The following is time complexities of above two algorithms. First is the time complexity of computing entropy according to Theorems 1, 2, and 4, which is $O(m|C| + n + |X'_{p_1}||Y'_{q_1}| + |X'_{p_2}||Y'_{q_2}|) = O(max(|X'_{p_1}||Y'_{q_1}|, |X'_{p_2}||Y'_{q_2}|))$ (the explanations of $m, n, X'_{p_1}, Y'_{q_1}, X'_{p_2}$ and Y'_{q_2} are shown in Theorems 1, 2, and 4). For convenience, Θ' is used to denote the above time complexity, i.e., $\Theta' = O(max(|X'_{p_1}||Y'_{q_1}|, |X'_{p_2}||Y'_{q_2}|))$.

In the algorithm $ACORE_{x'}$, the time complexity of *Steps 1-3* is Θ'; in *Step 4*, the time complexity is $|C|\Theta'$. Hence, the time complexity of algorithm $ACORE_{x'}$ is

$$O(\Theta' + |C|\Theta') = O(|C|(max(|X'_{p_1}||Y'_{q_1}|, |X'_{p_2}||Y'_{q_2}|))) = O(max(|C||X'_{p_1}||Y'_{q_1}|, |C||X'_{p_2}||Y'_{q_2}|)).$$

In algorithm $ARED_{x'}$, the time complexity of *Steps 1-3* is Θ'; the time complexity of *Steps 4-5* is also $O(\Theta')$; in *Step 7*, the time complexity of adding attributes is $O(|C|\Theta')$; in *Step 8*, the time complexity of deleting redundant attributes is $O(|B|\Theta')$. Hence, the total time complexity of algorithm IA_RED_x is

$$O(\Theta' + |C|\Theta' + |B|\Theta') = O(|C|\Theta') = O(|C|^2|U| + max(|C||X'_{p_1}||Y'_{q_1}|, |C||X'_{p_2}||Y'_{q_2}|)).$$

To stress above findings, the time complexities of computing core and reduct are shown in Table 1. $CA\_CORE_{x'}$ and $CA\_RED_{x'}$ denote classic algorithms based on information entropy for computing core and reduct, respectively.

Table 1: Comparison of time complexity

Entropy	Classic	Incremental
	$O(\|U\|^2)$	$O(\|U\|\|C\| + max(\|X'_{p_1}\|\|Y'_{q_1}\|, \|X'_{p_2}\|\|Y'_{q_2}\|))$
Core	$CA\_CORE_{x'}$	$ACORE_{x'}$
	$O(\|C\|\|U\|^2)$	$O(max(\|C\|\|X'_{p_1}\|\|Y'_{q_1}\|, \|C\|\|X'_{p_2}\|\|Y'_{q_2}\|))$
Reduct	$CA\_RED_{x'}$	$ARED_{x'}$
	$O(\|C\|^2\|U\| + \|C\|\|U\|^2)$	$O(\|C\|^2\|U\| + max(\|C\|\|X'_{p_1}\|\|Y'_{q_1}\|, \|C\|\|X'_{p_2}\|\|Y'_{q_2}\|))$

Table 2: Description of data sets

	Data sets	Samples	Attributes	Classes
1	Backup-large	307	35	19
2	Dermatology	366	33	6
3	Breast-cancer-wisconsin(Cancer)	683	9	2
4	Mushroom	5644	22	2
5	Letter-recognition(Letter)	20000	16	26
6	Shuttle	58000	9	7

In Table 1, $|X'_{p_1}||Y'_{q_1}|$ (or $|X'_{p_2}||Y'_{q_2}|$) is usually much smaller than $|U|^2$. Hence, based on the three entropies, the calculation of proposed algorithms ($ACORE_{x'}$ and $ARED_{x'}$) are usually much smaller than that of the classic algorithms for reduct (or core).

6. Experimental analysis

The objective of the following experiments is to show effectiveness and efficiency of the proposed reduction algorithm $ARED_{x'}$. Due to that the core is a subset of a reduct, we only run the reduction algorithms in the experiments. Data sets used in the experiments are outlined in Table 2, which were all downloaded from UCI repository of machine learning databases. All the experiments have been carried out on a personal computer with Windows XP and Inter(R) Core(TM) 2 Quad CPU Q9400, 2.66 GHz and 3.37 GB memory. The software being used is Microsoft Visual Studio 2005 and programming language is C#.

There are two objectives to conduct the experiments. The first one is to show whether the reduct found by $ARED_{x'}$ is feasible by comparing with that of *CAR* (see in section 6.1). The second one is to compare the efficiency of $ARED_{x'}$ and *CAR* (see in section 6.2). Six UCI data sets are employed to test the two algorithms. *Mushroom* and *Breast-cancer-wisconsin* are data sets with missing values, and for a uniform treatment of all data sets, the objects with missing values have been removed. Moreover, *Shuttle* is preprocessed using the data tool Rosetta.

6.1. Effectiveness analysis

In this subsection, to test the effectiveness of $ARED_{x'}$, four common evaluation measures in rough set theory are employed to evaluate the decision performance of the reducts found by *CAR* and $ARED_{x'}$. The four evaluation measures are approximate classified precision, approximate classified quality, certainty measure and consistency measure.

In [14], Pawlak defined the approximate classified precision and approximate classified quality to describe the precision of approximate classification in rough set theory.

Definition 10. *Let $S = (U, C \cup D)$ be a decision table and $U/D = \{X_1, X_2, \cdots, X_r\}$. The approximate classified precision of C with respect to D is defined as*

$$AP_C(D) = \frac{|POS_C(D)|}{\sum_{i=1}^{r} |\overline{C}X_i|}. \tag{7}$$

Definition 11. *Let $S = (U, C \cup D)$ be a decision table. The approximate classified quality of C with respect to D is defined as*

$$AQ_C(D) = \frac{|POS_C(D)|}{|U|}. \tag{8}$$

In rough set theory, by adopting reduction algorithms, one can get reducts for a given decision table. Then, based on a reduct, a set of decision rules can be generated from a decision table. Here briefly recalls the notions of decision rules [14, 52], which will be used in the following development.

Definition 12. *Let $S = (U, C \cup D)$ be a decision table. $U/C = \{X_1, X_2, \cdots, X_m\}$, $U/D = \{Y_1, Y_2, \cdots, Y_n\}$ and $\cap Y_j \neq \emptyset$. $des(X_i)$ and $des(Y_j)$ are denoted the descriptions of the equivalence classes X_i and Y_j, respectively. A decision rule induced by C is formally defined as*

$$Z_{ij} : des(X_i) \rightarrow des(Y_j), X_i \in U/C, Y_j \in U/D. \tag{9}$$

In [53, 54], certainty measure and support measure were introduced to evaluate a single decision rule. For a rule set, two measures were introduced to measure the certainty and consistency in [15]. However, it has been pointed out that those two measures cannot give elaborate depictions of the certainty and consistency for a rule set in [52]. To address this issue, Qian et al. in [52] defined certainty measure and consistency measure to evaluate the certainty and consistency of a set of decision rules, which has attracted considerable attention [55].

Definition 13. *Let $S = (U, C \cup D)$ be a decision table, $U/C = \{X_1, X_2, \cdots, X_m\}$, $U/D = \{Y_1, Y_2, \cdots, Y_n\}$, and $RULE = \{Z_{ij} | Z_{ij} : des(X_i) \rightarrow des(Y_j), X_i \in U/C, Y_j \in U/D\}$. The certainty measure α of the decision rules on S is defined as*

$$\alpha(S) = \sum_{i=1}^{m} \sum_{j=1}^{n} \frac{|X_i \cap Y_j|^2}{|U||X_i|}. \tag{10}$$

Definition 14. *Let $S = (U, C \cup D)$ be a decision table, $U/C = \{X_1, X_2, \cdots, X_m\}$, $U/D = \{Y_1, Y_2, \cdots, Y_n\}$, and $RULE = \{Z_{ij} | Z_{ij} : des(X_i) \rightarrow des(Y_j), X_i \in U/C, Y_j \in U/D\}$. The consistency measure β of the decision rules on S is defined as*

$$\beta(S) = \sum_{i=1}^{m} \frac{|X_i|}{|U|} [1 - \frac{4}{|X_i|} \sum_{j=1}^{n} \frac{|X_i \cap Y_j|^2}{|X_i|} (1 - \frac{|X_i \cap Y_j|}{|X_i|})]. \tag{11}$$

For each data set in Table 2, 50% objects are selected randomly and replaced by new ones. Then, algorithms *CAR* and $ARED_{x'}$ are employed to update reduct of each varying data set. The generated reducts are shown in Tables 3, 5 and 7, and the evaluation results of reducts based on the four evaluation measures are shown in Tables 4, 6 and 8.

• Comparison of *CAR* and $ARED_{x'}$ based on complementary entropy

It is easy to note from Table 4 the values of the four evaluation measures of the generated reducts by using the two algorithms are very close, and even identical on some data sets. But, according to Table 3, the computational time of $ARED_{x'}$ is much smaller than that of *CAR*. In other words, the performance and decision making of the reduct found by $ARED_{x'}$ are very close to that of *CAR*, but $ARED_{x'}$ is more efficient. Hence, the experimental results indicate that, compared with the classic reduction algorithm *CAR* based on complementary entropy, the algorithm $ARED_{x'}$ can find a feasible reduct in a much shorter time.

• Comparison of *CAR* and $ARED_{x'}$ based on combination entropy

From Tables 5 and 6, it is easy to get that algorithm $ARED_{x'}$ can find a reduct which has same performance and decision making as those of reduct generated by *CAR* in a much shorter time. Thus, compared with *CAR* based on combination entropy, the algorithm $ARED_{x'}$ is more efficient to deal with dynamic data sets.

• Comparison of *CAR* and $ARED_{x'}$ based on Shannon's entropy

According to experimental results in Tables 7 and 8, it is easy to see that performance and decision making of reducts generated by $ARED_{x'}$ and *CAR* are relatively close. But, $ARED_{x'}$ is more efficient than *CAR*. Hence, one can observe that algorithm $ARED_{x'}$ can find a feasible reduct, and save lots of computational time.

Table 3: Comparison of reducts based on complementary entropy

	CAR		$ARED_{x'}$	
Data sets	Reduct	Time/s	Reduct	Time/s
Backup-large	1,4,7,8,10,13,16,22	4.5937	1,4,7,8,10,13,22	0.1406
Dermatology	1,2,3,4,5,14,16,18,19	5.7812	1,2,3,4,5,14,18,19	0.2031
Cancer	1,2,4,6	2.0000	1,2,4,6	0.5000
Mushroom	1,2,3,5,7,8,9,20	486.26	1,2,3,5,7,9,20	37.312
Letter	1,2,3,4,5,8,9,10,11,12,13,15,16	9602.6	1,2,3,4,5,8,9,10,11,12,15,16	358.81
Shuttle	1,2,3,5	17010.1	1,2,3,5	5122.1

Table 4: Comparison of evaluation measures based on complementary entropy

	CAR				$ARED_{x'}$			
Data sets	AQ	AP	α	β	AQ	AP	α	β
Backup-large	1.0000	0.9999	1.0000	1.0000	1.0000	0.9999	1.0000	1.0000
Dermatology	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Cancer	0.9999	1.0000	1.0000	1.0000	0.9999	1.0000	1.0000	1.0000
Mushroom	1.0000	1.0000	1.0000	1.0000	0.9996	0.9993	0.9993	0.9986
Letter	0.9999	1.0000	1.0000	1.0000	0.9999	1.0000	1.0000	1.0000
Shuttle	0.9988	0.9977	0.9976	0.9953	0.9988	0.9977	0.9976	0.9953

Table 5: Comparison of reducts based on combination entropy

	CAR		$ARED_{x'}$	
Data sets	Reduct	Time/s	Reduct	Time/s
Backup-large	1,4,7,8,10,13,16,22	4.5312	1,4,7,8,10,13,22	0.1366
Dermatology	1,2,3,4,5,14,16,18,19	5.7500	1,2,3,4,5,14,18,19	0.2030
Cancer	1,2,4,6	1.9843	1,2,4,6	0.4987
Mushroom	1,2,3,4,7,8,9,20	478.37	1,2,3,4,7,8,9,20	37.301
Letter	1,2,3,4,5,8,9,10,11,12,13,15,16	8825.6	1,2,3,4,5,8,9,11,12,13,15,16	358.81
Shuttle	1,2,3,5	19935.7	1,2,3,5	5089.1

Table 6: Comparison of evaluation measures based on combination entropy

	CAR				$ARED_{x'}$			
Data sets	AQ	AP	α	β	AQ	AP	α	β
Backup-large	1.0000	0.9999	1.0000	1.0000	1.0000	0.9999	1.0000	1.0000
Dermatology	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Cancer	0.9999	1.0000	1.0000	1.0000	0.9999	1.0000	1.0000	1.0000
Mushroom	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Letter	0.9999	1.0000	1.0000	1.0000	0.9999	1.0000	1.0000	1.0000
Shuttle	0.9988	0.9977	0.9976	0.9953	0.9988	0.9977	0.9976	0.9953

Table 7: Comparison of reducts based on Shannon's entropy

	CAR		$ARED_{x'}$	
Data sets	Reduct	Time/s	Reduct	Time/s
Backup-large	1,4,5,6,7,8,10,16,22	5.3281	1,4,5,6,7,8,10,22	0.1368
Dermatology	1,4,5,9,12,14,17,18,21,22,26	5.7500	1,4,5,9,12,14,17,18,26	0.2030
Cancer	2,3,5,6	1.9843	2,3,5,6	0.5125
Mushroom	1,2,3,4,5,9,20,22	482.75	1,2,3,5,9,20,22	37.751
Letter	1,2,3,4,5,8,9,10,11,12,13,15,16	8389.8	1,2,3,4,5,8,9,10,11,13,16	358.87
Shuttle	1,2,3,5	23698.5	1,2,3,5	5130.6

Table 8: Comparison of evaluation measures based on Shannon's entropy

	CAR				$ARED_{x'}$			
Data sets	AQ	AP	α	β	AQ	AP	α	β
Backup-large	0.9999	1.0000	1.0000	1.0000	0.9999	1.0000	1.0000	1.0000
Dermatology	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Cancer	0.9999	1.0000	1.0000	1.0000	0.9999	1.0000	1.0000	1.0000
Mushroom	1.0000	1.0000	1.0000	1.0000	0.9996	0.9993	0.9993	0.9986
Letter	0.9999	1.0000	1.0000	1.0000	0.9999	1.0000	1.0000	1.0000
Shuttle	0.9988	0.9977	0.9976	0.9953	0.9988	0.9977	0.9976	0.9953

Table 9: Comparison of computational time based on LE

Data sets	CAR					$ARED_{x'}$				
	10%	20%	30%	40%	50%	10%	20%	30%	40%	50%
Backup-large	4.6875	4.8594	5.0625	4.4688	4.5937	0.0312	0.0468	0.0781	0.1250	0.1406
Dermatology	5.3906	5.5781	5.6093	6.6250	5.7812	0.0468	0.0781	0.1250	0.1562	0.2031
Cancer	1.4843	1.5937	1.7187	1.9062	2.0000	0.0781	0.1718	0.2656	0.4062	0.5000
Mushroom	221.01	283.78	368.71	411.28	468.26	4.3125	9.9218	17.906	29.000	37.312
Letter	8133.2	8283.2	8630.1	9260.2	9602.6	83.625	140.25	220.48	292.68	358.81
Shuttle	12909.8	13905.3	14523.9	16100.1	17010.1	776.10	1707.3	2979.3	4098.7	5122.1

Table 10: Comparison of computational time based on CE

Data sets	CAR					$ARED_{x'}$				
	10%	20%	30%	40%	50%	10%	20%	30%	40%	50%
Backup-large	4.8125	4.8750	5.0468	4.4531	4.5312	0.0412	0.0568	0.0781	0.1193	0.1366
Dermatology	5.3906	5.4687	5.5156	5.7500	5.7500	0.0478	0.0781	0.1250	0.1563	0.2030
Cancer	1.4843	1.5781	1.7187	1.9062	1.9843	0.0781	0.1718	0.2656	0.4063	0.4987
Mushroom	222.75	253.15	371.06	424.51	478.37	4.3712	9.9118	17.860	28.937	37.301
Letter	7201.4	7669.8	8011.8	8485.2	8825.6	83.505	142.05	200.45	289.66	358.81
Shuttle	14122.1	15468.9	16236.1	18896.9	19935.7	758.11	1668.3	3005.3	4120.8	5089.1

6.2. Efficiency analysis

The objective of experiments in this subsection is to further illustrate efficiency of algorithm $ARED_{x'}$. For each data set in Table 2, 10%, 20%, $\cdots$, 50% objects are selected, in order, and are replaced by new ones. For each data set after each variation (from 10% to 50%), algorithms CAR and $ARED_{x'}$ are used to update reducts, respectively. The efficiency of the two algorithms are demonstrated by comparing their computational time. Experimental results are shown in Tables 9-11. 10%, 20%, $\cdots$, 50% in the tables mean 10%, 20%, $\cdots$, 50% objects with data values being varied, respectively.

Based on the three entropies, it is easy to see from the Tables 9-11 that, for each data set after each variation, the computational time of algorithm $ARED_{x'}$ is much smaller than that of the classic reduction algorithm CAR, especially for the larger data sets *Mushroom* and *Letter*. In addition, with the number of varying objects increasing (from 10% to 50%), the computational time of $ARED_{x'}$ is always much smaller than that of CAR. Hence, the experimental results show that algorithm $ARED_{x'}$ is efficient to solving data sets with dynamically varying data values.

6.3. Related discussion

This subsection summarizes the advantages of algorithm $ARED_{x'}$ for generating reduct and offers explanatory comments. Obviously, in above two subsections, the experimental results better illustrate effectiveness and efficiency of $ARED_{x'}$.

• Algorithm $ARED_{x'}$ based on each of the three entropies can find a feasible reduct of a given dynamic decision table.

According to experimental results in Section 6.1, it is easy to get that the decision performance of reducts generated by CAR and $ARED_{x'}$ are very close, and even identical on some data sets. Hence, compared with the classic reduction algorithms based on the three entropies, the reduct generated by $ARED_{x'}$ can be considered as a feasible reduct.

• Compared with the classic reduction algorithms (CAR) based the three entropies, $ARED_{x'}$ finds a reduct in a very efficient manner.

Table 11: Comparison of computational time based on SE

Data sets	CAR					$ARED_{x'}$				
	10%	20%	30%	40%	50%	10%	20%	30%	40%	50%
Backup-large	4.5937	4.7343	5.0000	5.2187	5.3281	0.0402	0.0568	0.0781	0.1194	0.1368
Dermatology	5.1875	5.5781	4.2500	6.3125	6.6562	0.0478	0.0750	0.1250	0.1565	0.2036
Cancer	1.4843	1.6093	1.7500	1.8750	1.9843	0.0781	0.1709	0.2625	0.4063	0.5125
Mushroom	221.84	256.36	330.31	370.59	482.75	4.3825	9.9288	17.876	28.937	37.751
Letter	7189.7	7502.8	7686.7	8015.2	8389.8	83.625	141.25	223.48	295.48	358.87
Shuttle	12968.3	15126.7	17356.1	20123.3	23698.5	775.10	1705.3	2985.3	4101.5	5130.6

Experimental results in Section 6.2 show that, based on the three entropies, the computational time of generating reduct by using $ARED_{x'}$ is much shorter than that of CAR.

• The development in the paper may make an important contribution to deal with large-scale dynamic data sets in applications.

The experimental results show that the efficiency of $ARED_{x'}$ is obvious in solving large-scale dynamic data sets. In reality, acquiring knowledge from large-scale complicated data sets is still a challenging issue. It is our wish that this paper provides new techniques for dealing with large-scale dynamic data sets.

7. Conclusions and future work

Feature selection for dynamic data sets is still a challenging issue in the field of artificial intelligence. In this paper, based on three representative entropies, an attribute reduction algorithm is proposed to update reduct of data sets with dynamically varying data values. The experimental results show that, compared with the classic reduction algorithms based on the three entropies, this algorithm can generate a feasible reduct in a much shorter time. It is our wish that this study provides new views and thoughts on dealing with large-scale and complicated dynamic data sets in applications.

It should be pointed out that updating mechanisms of the three entropies introduced in this paper are only applicable when data are varied one by one, whereas many real data may vary in groups in application. This gives rise to many difficulties for the proposed feature selection algorithm to deal with. Hence, it is expected to carry out the following work to improve efficiency of selecting useful features in dynamic data sets in the future:

• Developing group updating mechanisms of entropies and relative feature selection algorithms.

• Discernibility matrix is one of key concepts in rough set. Future work may include analyzing discernibility matrix for data sets with dynamically varying data values.

• Designing efficient feature selection algorithms based on generalized rough set models such as incomplete rough set model, dominance rough set model and multi-granulation rough set model.

Acknowledgment

This work was supported by National Natural Science Fund of China (Nos. 71031006, 60903110, 70971080), Initial Special Research for 973 Program of China(973)(No. 2011CB311805), The Research Fund for the Doctoral Program of Higher Education (20101401110002).

References

[1] F. Hu, G. Y. Wang, H. Huang, Y. Wu, Incremental attribute reduction based on elementary sets // Proceedings of the 10th International Conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing, Regina, Canada (2005) 185-193.

[2] J.Y.Liang, W.Wei, Y.H.Qian, An incremental approach to computation of a core based on conditional entropy, Chinese Journal of System Engineering Theory and Practice 4 (2008) 81-89.

[3] D. Liu, T. R. Li, D. Ruan, W. L. Zou, An incremental approach for inducing knowledge from dynamic information systems, Fundamenta Informaticae 94 (2009) 245-260.
[4] M. Orlowska, Maintenance of knowledge in dynamic information systems//R. Slowinski ed. Proceeding of the intelligent decision support, Handbook of Applications and Advances of the Rough Set Theory, Dordrecht: Kluwer Academic Publishers (1992) 315-330.
[5] L. Shan, W. Ziarko, Data-based acquisition and incremental modification of classification rules, Computational Intelligence 11 (1995) 357-370.
[6] M. Yang, An incremental updating algorithm for attributes reduction based on the improved discernibility matrix, Chinese Journal of Computers 30(5) (2007) 815-822.
[7] Z. Zheng, G. Wang G, RRIA: a rough set and rule tree based incremental knowledge acquisition algorithm, Fundamenta Informaticae 59 (2004) 299-313.
[8] C. C. Chan, A rough set approach to attribute generalization in data mining, Information Science 107 (1998) 169-176.
[9] T. R. Li, D. Ruan, W. Geert, J. Song, Y. Xu, A rough sets based characteristic relation approach for dynamic attribute generalization in data mining, Knowledge-Based System 20 (2007) 485-494.
[10] Y. Cheng, The incremental method for fast computing the rough fuzzy approximations, Data & Knowledge Engineering 70 (2011) 84-100.
[11] D. Liu, J. B. Zhang, T. R. Li, A probabilistic rough set approach for incremental learning knowledge on the change of attribute. In: Proceedings 2010 International Conference on Foundations and Applications of Computational Intelligence (2010) 722-727.
[12] H. M. Chen, T. R. Li, S. J. Qiao, D. Ruan, A rough set based dynamic maintenance approach for approximations in coarsening and refining attribute values. International Journal of Intellegent Systems 25 (2010) 1005-1026.
[13] D. Liu, T. R. Li, G. R. Liu, P. Hu, An incremental approach for inducing interesting knowledge based on the change of attribute values. In: Proceedings 2009 IEEE International Conference on Granular Computing, Nanchang, China (2009) 415-418.
[14] Z. Pawlak, Rough Sets: Theoretical Aspects of Reasoning about Data, Kluwer Academic Publishers, Boston, 1991.
[15] Z. Pawlak, A. Skowron, Rudiments of rough sets, Information Sciences 177(1) (2007) 3-27.
[16] H. S. Own, A. Abraham, A new weighted rough set framework based classification for Egyptian NeoNatal Jaundice, Applied Soft Computing 12 (3) (2012) 999-1005.
[17] K. Kaneiwa, A rough set approach to multiple dataset analysis, Applied Soft Computing 11 (2) (2011) 2538-2547.
[18] A. K. Das, J. Sil, An efficient classifier design integrating rough set and set oriented database operations, Applied Soft Computing 11 (2) (2011) 2279-2285.
[19] P. Dey, S. Dey, S. Datta, J. Sil, Dynamic discreduction using Rough Sets, Applied Soft Computing 11 (5) (2011) 3887-3897.
[20] R. R. Chen, Y. I. Chiang, P. P. Chong, Y. H. Lin, H. K Chang, Rough set analysis on call center metrics, Applied Soft Computing 11 (4) (2011) 3804-3811.
[21] W. Wei, J. Y. Liang, Y. H. Qian, A comparative study of rough sets for hybrid data, Information Sciences 190 (1) (2012) 1-16.
[22] D. G. Chen, Q. H. Hu, Y. P. Yang, Parameterized attribute reduction with Gaussian kernel based fuzzy rough sets, Information Sciences 181 (23) (2011) 5169-5179.
[23] Q.H. Hu, Z.X. Xie, D.R. Yu, Hybrid attribute reduction based on a novel fuzzy-rough model and information granulation, Pattern Recognition 40(2007) : 3509 - 3521.
[24] Q.H. Hu, D.R. Yu, Z.X. Xie, Information-preserving hybrid data reduction based on fuzzy-rough techniques, Pattern Recognition Letters 27(5) (2006) 414 - 423.
[25] Y. Y. Yao, The superiority of three-way decisions in probabilistic rough set models, Information Sciences 181 (6) (2011) 1080-1096.
[26] G. Gediga, I. Düntsch, Rough approximation quality revisited, Artificial Intelligence 132 (2001) 219 - 234.
[27] R. Jensen, Q. Shen, Fuzzy-rough sets assisted attribute selection, IEEE Transactions on Fuzzy Systems 15 (1) (2007) 73 - 89.
[28] J.Y. Liang, Y.H. Qian, Information granules and entropy theory in information systems, Science in China(Series F) 51(10) (2008) 1427 - 1444.
[29] J.S. Mi, Y. Leung, W.Z. Wu, An uncertainty measure in partition-based fuzzy rough sets, International Journal General Systems 34 (2005) 77 - 90.
[30] W. Pedrycz, G. Vukovich, Feature analysis through information granulation and fuzzy sets, Pattern Recognition 35 (2002) 825 - 834.
[31] Y.H. Qian, C.Y. Dang, J.Y. Liang, D.W. Tang, Set-valued ordered information systems, Information Sciences, 179 (2009) 2809 - 2832.
[32] Z.B. Xu, J.Y. Liang, C.Y. Dang, K.S. Chin, Inclusion degree: a perspective on measures for rough set data analysis, Information Sciences 141 (2002) 227 - 236.
[33] J. Y. Liang, K. S. Chin, C. Y. Dang, C.M. Yam Richid, A new method for measuring uncertainty and fuzziness in rough set theory, International Journal of General Systems 31(4) (2002) 331 - 342.
[34] J.Y. Liang, Z.Z. Shi, The information entropy, rough entropy and knowledge granulation in rough set theory, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 12(1) (2004) 37 - 46.
[35] Y.H. Qian, J.Y. Liang, Combination entropy and combination granulation in rough set theory, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 16 (2) (2008) 179 - 193.
[36] C.E.Shannon, The mathematical theory of communication, The Bell System Technical Journal, 27(3,4) (1948) 373 - 423.
[37] Y.H. Qian , J.Y. Liang, W. Pedrycz, C.Y. Dang, Positive approximation: an accelerator for attribute reduction in rough set theory, Artificial Intelligence 174 (2010) 597 - 618.
[38] G. Y. Wang, H. Yu, D.C. Yang, Decision table reduction based on conditional information entropy, Chinese Journal of Computer 25 (7) (2002) 759 - 766.
[39] M. Dash, H. Liu, Consistency-based search in feature selection, Artificial Intelligence 151 (2003) 155 - 176.
[40] A. Skowron, C. Rauszer, The discernibility matrices and functions in information systems, In: R Slowiński(Eds), Intelligent Decision Support, Handbook of Applications and Advances of the Rough Sets Theory, Kluwer Academic Publisher, Dordrecht, 1992.
[41] M. Kryszkiewicz, P. Lasek, FUN: fast discovery of minimal sets of attributes functionally determining a decision attribute, Transactions on Rough Sets 9 (2008) 76 - 95.
[42] W. Ziarko, Variable precision rough set model, Journal of Computer and System Science 46(1) (1993) 39 - 59.

[43] Y. Y. Yao, Decision-theoretic rough set models, Lecture Notes in Artificial Intelligence 4481 (2007) 1 - 12.
[44] Y. Y. Yao, Y. Zhao, Attribute reduction in decisiontheoretic rough set models, Information Sciences 178 (2008) 3356 - 3373.
[45] S. Greco, B. Matarazzo, R. Slowinski, Rough approximation by dominance relations, International Journal of Intelligent Systems 17 (2002) 153-171.
[46] Y. H. Qian, C. Y. Dang, J. Y. Liang, D. W. Tang, Set-valued ordered information systems, Information Sciences, 179 (2009) 2809-2832.
[47] M. W. Shao, W. X. Zhang, Dominance relation and rules in an incomplete ordered information system, International Journal of Intelligent Systems 20 (2005) 13-27.
[48] D. Dubois, H. Prade, Rough fuzzy sets and fuuzy rough sets, International Journal of General Systems 17 (1990) 191 - 209.
[49] X. H. Hu, N. Cercone, Learning in relational databases: a rough set approach, International Journal of Computational Intelligence 11(2) (1995) 323 - 338.
[50] D. Slezak, Approximate entropy reducts, Fundamenta Informaticae 53 (3-4) (2002) 365 - 390.
[51] Z.Y. Xu, Z.P. Liu, B.R. Yang, W. Song, A quick attribute reduction algorithm with complexity of $\max(O(|C||U|), O(|C|^2|U/C|))$, Chinese Journal of Computer 29 (3) (2006) 391 - 398.
[52] Y. H. Qian, J.Y. Liang, D.Y. Li, H.Y. Zhang, C.Y. Dang, Measures for evaluating the decision performance of a decision table in rough set theory, Information Sciences 178 (2008) 181 - 202.
[53] I. Düntsch, G. Gediga, Uncertainty measures of rough set prediction, Artificial Intelligence 106 (1998) 109 - 137.
[54] V.N. Huynh, Y. Nakamori, A roughness measure for fuzzy sets, Information Sciences 173 (2005) 255 - 275.
[55] W. Wei, J.Y. Liang, Y.H. Qian, F. Wang, C.Y. Dang, Comparative study of decision performance of decision tables induced by attribute reductions, International Journal of General Systems, 39 (8) (2010) 813 - 838.

Appendix

This appendix lists the functions (m-files) developed by the authors and used in the examples in this book. Functions used that are part of MATLAB's commercial distribution have been omitted; the reader is referred to the respective MATLAB manuals.

In the following list, functions are ordered alphabetically by chapter. For further function details, including descriptions of input and output arguments, refer to MATLAB's help utility. Also see the complete source code of the listed m-files, provided as part of the software on the companion website.

Chapter 1

bayes_classifier Bayesian classification rule for c classes, modeled by Gaussian distributions (also used in Chapter 2).

comp_gauss_dens_val Computes the value of a Gaussian distribution at a specific point (also used in Chapter 2).

compute_error Computes the error of a classifier based on a data set (also used in Chapter 4).

em_alg_function EM algorithm for estimating the parameters of a mixture of normal distributions, with diagonal covariance matrices.

EM_pdf_est EM estimation of the pdfs of c classes. It is assumed that the pdf of each class is a mixture of Gaussians and that the respective covariance matrices are diagonal.

euclidean_classifier Euclidean classifier for the case of c classes.

Gaussian_ML_estimate Maximum Likelihood parameters estimation of a multivariate Gaussian distribution.

generate_gauss_classes Generates a set of points that stem from c classes, given the corresponding a priori class probabilities and assuming that each class is modeled by a Gaussian distribution (also used in Chapter 2).

k_nn_classifier k-nearest neighbor classifier for c classes (also used in Chapter 4).

knn_density_estimate k-nn-based approximation of a pdf at a given point.

mahalanobis_classifier Mahalanobis classifier for c classes.

mixt_model Generates a set of data vectors that stem from a mixture of normal distributions (also used in Chapter 2).

mixt_value Computes the value of a pdf that is given as a mixture of normal distributions, at a given point.

mixture_Bayes Bayesian classification rule for c classes, whose pdf's are mixtures of normal distributions.

Parzen_gauss_kernel Parzen approximation of a pdf using a Gaussian kernel.

plot_data Plotting utility, capable of visualizing 2-dimensional data sets that consist of, at most, 7 classes.

Auxiliary functions gauss.

Chapter 2

base_clas_coord Implements a specific weak classifier.

***base_clas_coord*_out** Computes the output of the weak classifier implemented by the base_clas_coord function.

boost_clas_coord Generation of a "strong" classifier, using the Adaboost algorithm, that utilizes weak classifiers generated by the base_clas_coord function.

boost_clas_coord_out Computes the output of a "strong" classifier B as a weighted sum of the outputs of the weak classifiers.

CalcKernel Computes the value of a kernel function between two points.

kernel_perce Implements the kernel perceptron algorithm.

NN_evaluation Returns the classification error of a neural network based on a data set.

NN_training Returns a trained multilayer perceptron.

perce Realizes the perceptron learning rule, in a batch mode.

perce_online Realizes the online perceptron learning rule.

plot_kernel_perce_reg Plots the decision boundary that is generated by the kernel perceptron algorithm.

plot_NN_reg Plots the decision boundary that is formed by a neural network.

SMO2 Generates a SVM classifier using either Platt's algorithm or one of its two modifications proposed by Keerthi.

SSErr Generates the linear classifier that optimizes the sum of error squares criterion.

svcplot_book Support Vector Machine plotting utility. It plots the decision regions, the decision surfaces and the margin obtained by a SVM classifier.

Chapter 3

cut_cylinder_3D Generates a cut cylinder in the 3-dimensional space.

im_point Performs the projection of a vector on the subspace spanned by the first m principal components, that result after performing kernel PCA on a data set.

K_fun Computes the value of a kernel function (polynomial or exponential) for two vectors.

kernel_PCA Performs kernel PCA based on a given set of data vectors.

lapl_eig Performs Laplacian eigenmap based on a given data set.

pca_fun Performs Principal Component Analysis (PCA) based on a data set.

plot_orig_trans_kPCA Plots, in different figures, (a) the data points and the classifier in the original (2-dimensional) data space and (b) the projections of the data points and the classifier in the space spanned by the two most significant principal components, as they are computed using the kernel PCA method.

scatter_mat Computes the within scatter matrix, the between scatter matrix and the mixture scatter matrix for a *c*-class classification problem, based on a given data set.

spiral_3D Creates a 3-dimensional Archimedes spiral.

svd_fun Performs Singular Value Decomposition (SVD) of a matrix.

Chapter 4

compositeFeaturesRanking Scalar feature ranking that takes into account the cross-correlation coefficient.

divergence Computes the divergence between two classes.

divergenceBhata Computes the Bhattacharyya distance between two classes.

exhaustiveSearch Exhaustive search for the best feature combination, depending on the adopted class separability measure.

Fisher Computes Fisher's discriminant ratio of a scalar feature in a 2-class problem.

normalizeMnmx Performs MinMax normalization in a given interval [*l r*].

normalizeSoftmax Performs Softmax normalization in the interval [0 1].

normalizeStd Performs data normalization to zero mean and standard deviation equal to 1.

plotData Plotting utility for class data.

plotHist Plots the histograms of two classes for the same feature.

ROC Plots the ROC curve and computes the area under the curve.

ScalarFeatureSelection Ranking Features are treated individually and are ranked according to the adopted class separability criterion.

ScatterMatrices Class separability measure, which is computed using the within-class and mixture scatter matrices.

SequentialBackward Selection Feature vector selection by means of the Sequential Backward Selection technique.

SequentialForward FloatingSelection Feature vector selection by means of the Sequential Forward Floating Selection technique.

SequentialForward Selection Feature vector selection by means of the Sequential Forward Selection technique.

simpleOutlierRemoval Removes outliers from a normally distributed data set by means of the thresholding method.

Chapter 5

BackTracking Performs backtracking on a matrix of node predecessors and returns the best path. This function is also used in Chapter 6.

DTWItakura Computes the Dynamic Time Warping cost between two feature sequences, based on the standard Itakura local constraints.

DTWItakuraEndp Similar to *DTWItakura*, with the addition that endpoints constraints are allowed in the test sequence.

DTWSakoe Computes the Dynamic Time Warping cost between two feature sequences, based on the Sakoe-Chiba local constraints.

DTWSakoeEndp Similar to *DTWSakoe*, with the addition that endpoints constraints are allowed in the test sequence.

editDistance Computes the Edit (Levenstein) distance between two sequences of characters.

Auxiliary functions stEnergy, stZeroCrossingRate, IsoDigitRec.

Chapter 6

BWDoHMMsc Computes the recognition probability of an HMM, given a sequence of discrete observations, by means of the scaled version of the Baum-Welch (any-path) method.

BWDoHMMst Same as *BWDoHMMSc*, except that no scaling is employed.

MultSeqTrainDoHMMBWsc Baum-Welch training (scaled version) of a Discrete Observation HMM, given multiple training sequences.

MultSeqTrain DoHMMVITsc Viterbi training (scaled version) of a Discrete Observation HMM, given multiple training sequences.

MultSeqTrainCoHMMBWsc Baum-Welch training (scaled version) of a Continuous Observation HMM, given multiple training sequences.

VitCoHMMsc Computes the scaled Viterbi score of an HMM, given a sequence of l-dimensional vectors of continuous observations, under the assumption that the pdf of each state is a Gaussian mixture.

VitCoHMMst Same as *VitCoHMMsc* except that no scaling is employed.

VitDoHMMsc Computes the scaled Viterbi score of a Discrete Observation HMM, given a sequence of observations.

VitDoHMMst Same as *VitDoHMMsc*, except that no scaling is employed.

Chapter 7

agglom Generalized Agglomerative Scheme (GAS) for data clustering. It runs, on demand, either the single-link or the complete-link algorithm.

BSAS Basic Sequential Algorithmic Scheme (BSAS algorithm) for data clustering.

CL_step Performs a step of the complete-link algorithm.

dendrogram_cut Determines the clusterings of a hierarchy that best fit the underlying clustering structure of the data set at hand.

fuzzy_c_means FCM algorithm for data clustering.

GMDAS Generalized Mixture Decomposition Algorithmic Scheme (GMDAS algorithm) for data clustering.

k_means k-means clustering algorithm.

k_medoids k-medoids clustering algorithm.

LLA Competitive leaky learning algorithm for data clustering.

possibi Possibilistic clustering algorithm, adopting the squared Euclidean distance.

SL_step Performs a step of the single-link algorithm.

spectral_Ncut2 Spectral clustering based on the normalized cut criterion.

valley_seeking Valley-seeking algorithm for data clustering.

Auxiliary functions cost_comput, distan, distant_init, rand_data_init, rand_init, reassign.

CHAPTER

Template Matching

5

5.1 INTRODUCTION

In this chapter, we assume that each class is represented by a single pattern. A set of such *reference* patterns (or *prototypes*) is available and stored in a database. Given an unknown *test* pattern, template matching consists of searching the database for the reference pattern most "similar" to the given test pattern. This is equivalent to defining a matching cost that quantifies similarity between the test pattern and the reference patterns.

Template-matching techniques are very common in string matching, speech recognition, alignment of molecular sequences, image retrieval, and so forth. They often come with different names depending on the application. For example, in speech recognition the term *dynamic time warping* is used, whereas in string matching *Edit* (or *Levenstein*) *distance* is quite common.

This chapter is devoted to a series of examples of increasing complexity, culminating in an example from speech recognition.

5.2 THE EDIT DISTANCE

A *string* pattern is defined as an *ordered sequence of symbols* taken from a discrete and finite set. For example, if the finite set consists of the letters of the alphabet, the strings are words. The *Edit distance* between two string patterns A and B, denoted $D(A,B)$, is defined as the minimum total number of changes (C), insertions (I), and deletions (R) required to change pattern A into pattern B,

$$D(A,B) = \underbrace{\min}_{j}[C(j) + I(j) + R(j)] \tag{5.1}$$

where j runs over all possible combinations of symbol variations in order to obtain B from A. If the two strings are exactly the same, then $D(A,B) = 0$. For every symbol "change," "insertion," or "deletion," the cost increases by one.

The required minimum is computed by means of the dynamic programming methodology [Theo 09, Section 8.2.1]. That is, an optimal path is constructed in the 2-dimensional grid formed by the two sequences in the 2-dimensional space, by locating one sequence across the horizontal axis and the other across the vertical axis. The Edit distance is commonly used in spell-checking systems where the prototypes stem from the vocabulary of words.

DOI: 10.1016/B978-0-12-374486-9.00005-1

Example 5.2.1. Compute the Edit distance between the words "book" and "bokks," taking the former as the reference string. Plot the optimal matching path and comment on the sequence of operations needed to change "bokks" to "book." Repeat for the words "template" (reference) and "teplatte."

Solution. Use function *editDistance* by typing

```
[editCost,Pred]=editDistance('book','bokks');
```

The Edit distance equals 2 and is stored in the variable *editCost*. The value of 2 is the result of a symbol change (*k* to *o*) and a deletion (*s* at the end of the word). To extract the matching path, give matrix *Pred* as input to function *BackTracking* as follows:

```
L_test=length('bokks'); % number of rows of the grid
L_ref=length('book'); % number of columns of the grid
[BestPath]=BackTracking(Pred,L_test,L_ref,1,'book','bokks');
% The fourth input argument indicates that a plot of the best
% path will be generated. The last two arguments serve as labels for the
resulting axes.
```

The resulting best path is stored in the vector variable *BestPath* and is presented in Figure 5.1(a), where the reference pattern has been placed on the horizontal axis. Each element of vector *BestPath* is a complex number and stands for a node in the path. The real part of the element is the node's row index and the imaginary part is its column index. Inspection of the path in Figure 5.1(a) reveals that the first

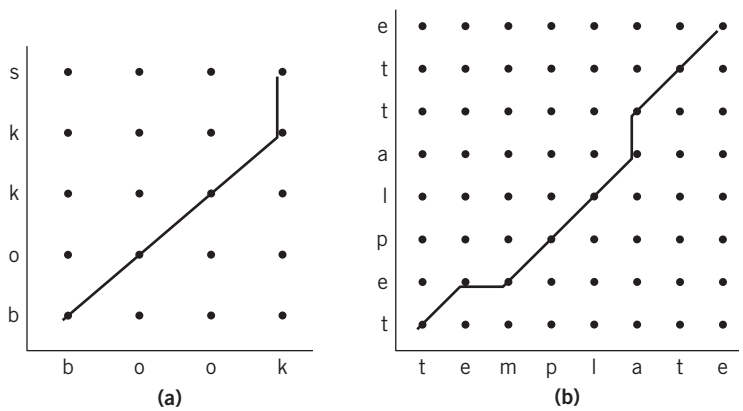

FIGURE 5.1

(a) Optimal matching path between "book" and "bokks." (b) Optimal matching path between "template" and "teplatte." A diagonal transition between two successive nodes amounts either to zero cost (if the corresponding symbols are the same) or to one (if the corresponding symbols are different). Horizontal transitions (symbol insertions) and vertical transitions (deletions) contribute one to the total cost.

occurrence of k in "bokks" has been changed to o. In addition, the vertical segment of the best path is interpreted as the deletion of s in "bokks." A total of one symbol change and one symbol deletion is needed to convert "bokks" to "book" in an optimal way.

Similarly, the Edit cost for matching "teplatte" against "template" equals 2, and the resulting best path can be seen in Figure 5.1(b). In this case, an insertion (horizontal segment) and a deletion (vertical segment) are required to convert "teplatte" to the reference pattern in an optimal way. ■

Exercise 5.2.1

Given the words "impose," "ignore," and "restore" as prototypes, determine which one stands for the most likely correction of the mistyped word "igposre" in terms of the edit distance. Note that, as is the case with most spell checkers, ties result in multiple correction possibilities.

5.3 MATCHING SEQUENCES OF REAL NUMBERS

In this section, we focus on a slightly different task, that of matching sequences of real numbers. In contrast to Section 5.2, where the goal was to change one string pattern to another, the aim here is to measure how similar/dissimilar are two given *ordered* sequences of numbers. For example, if we are given two real numbers, x, y, their similarity can be quantified by the absolute value of their difference. If we are given two vectors (i.e., two strings of real numbers of *equal* length), we can use the respective Euclidean distance. A more interesting case is when two sequences of numbers are of different length. One approach to this problem is to allow local "stretching"/"compressing," known as *warping*, achieved by constructing the optimal (low-cost) path through nodes of the respective grid. The grid is formed by the two sequences in the 2-dimensional space by locating one sequence along the horizontal axis and the other along the vertical axis.

Assuming that the reference sequence is placed on the horizontal axis, the dimensions of the grid are $J \times I$, where I and J are the lengths of the reference and test sequences, respectively. In the simplest case, the cost assigned to each node of the grid is equal to the absolute value of the difference between the respective numbers associated with a specific node. The type and the allowable degree of expansion/compression are determined by the so-called local constraints. Popular choices include the Sakoe-Chiba and Itakura constraints [Theo 09, Section 8.2]. Basically, these are constraints imposed on the allowable jumps among nodes in the grid.

The purpose of matching sequences of numbers is pedagogical and is used as an intermediate step to help the reader acquire a better understanding of the concepts underlying dynamic time warping for speech recognition, which is treated in the next section (see [Theo 09, Section 8.2.3]).

Example 5.3.1. Let $P = \{-1, -2, 0, 2\}$ be the prototype and $T = \{-1, -2, -2, 0, 2\}$ be the unknown pattern.

1. Compute the matching cost and the resulting best path by adopting the Sakoe-Chiba local constraints. Comment on the shape of the resulting best path.
2. Repeat with $T = \{-1, -2, -2, -2, -2, 0, 2\}$.

Solution

Step 1. For $T = \{-1, -2, -2, -2, -2, 0, 2\}$, use function *DTW Sakoe* and type

```
P=[-1,-2,0,2];
T=[-1,-2,-2,0,2];
[MatchingCost,BestPath,D,Pred]=DTWSakoe(P,T,1);
```

where D is the array having as elements the costs associated with optimally reaching each node of the grid. The value 1 is used if a plot is required; if not, 0 is used. Although the two sequences differ by one symbol, the matching cost equals 0. This is due to the fact that a) the absolute difference was employed as the (node) cost and b) the only difference between the two sequences is symbol repetition.

To further interpret the result, observe the respective best path in Figure 5.2(a). It can be seen that the vertical segment of the path corresponds to a local stretching operation; that is, the symbol -2 of the prototype (horizontal axis) is matched against two consecutive occurrences of -2 in sequence T.

Step 2. To repeat the experiment with $T = \{-1, -2, -2, -2, -2, 0, 2\}$ type

```
P=[-1,-2,0,2];
T=[-1,-2,-2,-2,-2,0,2];
[MatchingCost,BestPath,D,Pred]=DTWSakoe(P,T,1);
```

The matching cost remains 0 and the resulting best path is presented in Figure 5.2(b). It can be seen that the vertical segment of the path is now four nodes long. This should not come as a surprise

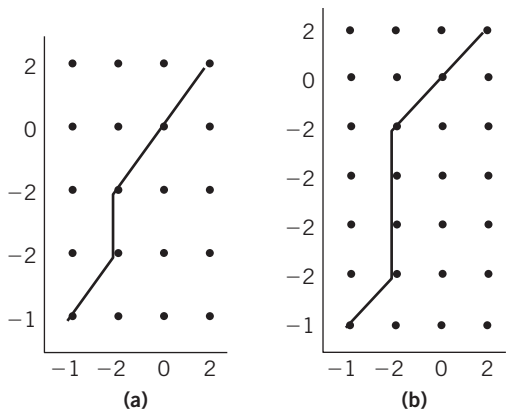

FIGURE 5.2

(a) Best path for $P = \{-1, -2, 0, 2\}$ and $T = \{-1, -2, -2, 0, 2\}$. (b) Best path for $P = \{-1, -2, 0, 2\}$ and $T = \{-1, -2, -2, -2, -2, 0, 2\}$.

because, as before, T differs from P in terms of *symbol repetition*. *The length of repetition does not affect the cost*; it only causes a more intense time-warping effect. To draw an analogy with speech, we may have the same phoneme but one time it is said fast and another time slowly. Long horizontal or vertical segments are very common when the Sakoe-Chiba local constraints are employed. If no global constraints are specified, the horizontal (vertical) segments can become arbitrarily long. This behavior is often undesirable with real-world signals, such as in speech.

Example 5.3.2. Let $P = \{1,0,1\}$ be the prototype, and let $T_1 = \{1,1,0,0,0,1,1,1\}$, $T_2 = \{1,1,0,0,1\}$ be two unknown patterns. Compute the matching cost for the standard Itakura local constraints between P and T_1 and between P and T_2.

Solution. To compute the matching cost for the two unknown patterns using the standard Itakura local constraints, use function *DTWItakura* and type

```
P=[1,0,1];
T1=[1,1,0,0,0,1,1,1];
T2=[1,1,0,0,1];
[MatchCost1,BestPath1,D1,Pred1]=DTWItakura(P,T1,1);
[MatchCost2,BestPath2,D2,Pred2]=DTWItakura(P,T2,1);
```

The returned value of *MatchCost*1 is ∞, whereas the value of *MatchCost*2 is 0. This is because one property of the standard Itakura constraints is that the maximum allowed stretching factor for the prototype is 2. In other words, the length of the unknown pattern has to be, in the worst case, twice the length of the prototype. If this rule is violated, the *DTWItakura* function returns ∞. In the case of P and T_2, this rule is not violated and the best path can be seen in Figure 5.3.

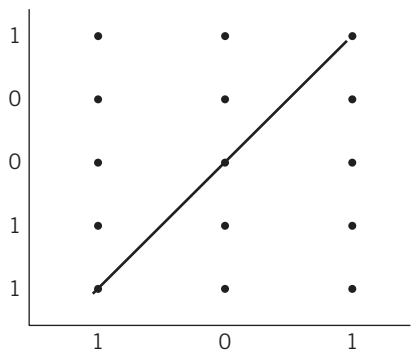

FIGURE 5.3

Best path for $P = \{1,0,1\}$ and $T = \{1,1,0,0,1\}$ using the standard Itakura local constraints.

Example 5.3.3. This example demonstrates the importance of the endpoint constraints [Theo 09, Section 8.2.3]. Let the sequence $P = \{-8, -4, 0, 4, 0, -4\}$ be a prototype. Also let the sequence $T = \{0, -8, -4, 0, 4, 0, -4, 0, 0\}$ be the unknown pattern.

1. Compute the matching cost by adopting the Sakoe-Chiba local constraints and comment on the result.
2. Repeat, allowing for endpoint constraints. Specifically, omit at most two symbols from each endpoint of T.

Solution

Step 1. For the first case, type

```
P=[-8,-4,0,4,0,-4];
T=[0,-8,-4,0,4,0,-4,0,0];
[MatchingCost,BestPath,D,Pred]=DTWSakoe(P,T,1);
```

The matching cost turns out to be 16. In practice, the cost is normalized by dividing it by the length of the best path. In this example, the best path is 9 nodes long, as can be seen in Figure 5.4(a), and so the normalized cost is 1.778. Hence, although P and T can be considered practically the same (they only differ in the trailing zeros), the matching cost is nonzero.

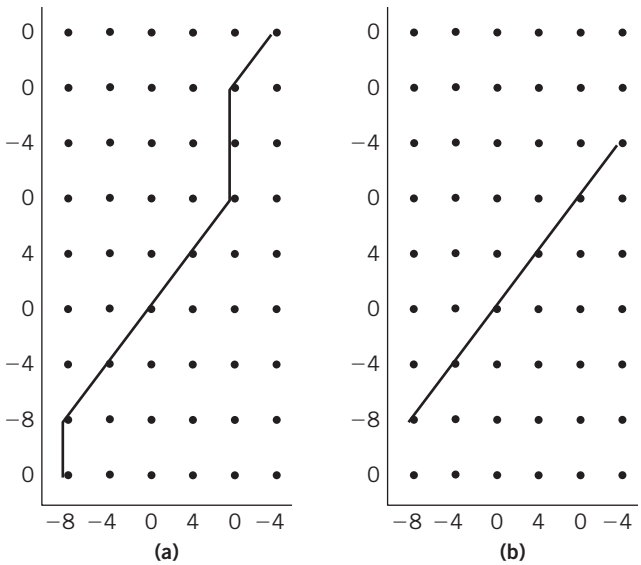

FIGURE 5.4

(a) Best path for $P = \{-8, -4, 0, 4, 0, -4\}$ and $T = \{0, -8, -4, 0, 4, 0, -4, 0, 0\}$, no endpoint constraints.
(b) Previous sequences matched while allowing endpoint constraints.

Inspection of Figure 5.4(a) reveals that time stretching occurs at the beginning and end of the path and is due to the existence of zeros at the endpoints. This is a common situation; that is, the unknown pattern contains "garbage" near the endpoints. As a remedy, we can resort to a variation of the standard matching scheme, where it is possible to omit a number of symbols at the endpoints of the unknown pattern; it suffices to specify the maximum number of symbols to omit. This type of enhancement to the standard matching mechanism can be easily embedded in the dynamic programming methodology.

Step 2. To employ the endpoint constraints in the current example, use function *DTWSakoeEndp* and type

```
P=[-8,-4,0,4,0,-4];
T=[0,-8,-4,0,4,0,-4,0,0];
[MatchingCost,BestPath,D,Pred]=DTWSakoeEndp(P,T,2,2,1);
```

In this function call, the third and fourth arguments stand for the number of symbols that can be omitted at each endpoint (2 in this example). The fifthindicates that a plot of the best path will be generated. The resulting matching cost is zero, as can be verified from Figure 5.4(b), where the first row and the last two rows of the grid have been skipped by the algorithm.

Endpoint constraints are very useful in speech recognition because the unknown pattern usually has silence periods around the endpoints, whereas the prototypes are free of such phenomena. ■

5.4 DYNAMIC TIME WARPING IN SPEECH RECOGNITION

Here we focus on a simple task in speech recognition known as *isolated word recognition (IWR)*. We assume that the spoken utterance consists of discrete words; that is, there exist sufficient periods of silence between successive words (hence the term "isolated"). This is a convenient assumption, since it allows for employing segmentation algorithms capable of locating the boundaries (endpoints) of the spoken words with satisfactory precision. Note that in practice a certain amount of silence/noise is likely to exist close to the endpoints of the detected word after the segmentation stage.

At the heart of any IWR system is an architecture consisting of a set of reference patterns (prototypes) and a distance measure. Recognition of a test (unknown) pattern is achieved by searching for the best match between the test and each one of the reference patterns, on the basis of the adopted measure.

As a first stage, a feature extraction algorithm converts each signal into a sequence of feature vectors—instead of matching sequences of real numbers, the task is computing the matching cost between two sequences of vectors. However, the rationale is the same, and the only difference lies in replacing the absolute value with the Euclidean distance between vectors. In speech recognition, this type of matching is known as *dynamic time warping* (DTW).

We now develop a simple, speaker-dependent IWR system for a 10-word vocabulary consisting of "zero," "one," . . . , "nine" and uttered by a single male speaker. We are actually building an "isolated digit recognition" system. We need a total of 10 utterances to use as prototypes and a number of utterances for testing.

At this point, you may record your own audio files or you may use the files available via this book's website. If you record your own files, name the prototypes as follows: *zero.wav*, *one.wav*, and so on, for the sake of compatibility, and place all 10 in a single folder (this is the naming convention we have adopted for the audio samples on the website). Note that "clean" prototypes are desirable. That is, make sure silence/noise has been removed, at least to the extent possible, from the endpoints of the prototypes before storing. Such care need not be taken with the samples that will be used for testing. In addition, the file names for the test patterns need not follow any naming convention and it suffices to store the unknown patterns in the folder where the prototypes are held.

To build the system, we use short-term energy and short-term zero-crossing rate as features [Theo 09, Section 7.5.4], so that each signal is represented by a sequence of 2-dimensional feature vectors. Note that this is not an optimal feature set in any sense and has only been adopted for simplicity.

The feature extraction stage is accomplished by typing the following code:

```
protoNames={'zero','one','two','three','four','five',...
   'six','seven','eight','nine'};
for i=1:length(protoNames)
   [x,Fs,bits]=wavread(protoNames{i});
   winlength = round(0.02*Fs); % 20 ms moving window length
   winstep = winlength; % moving window step. No overlap
   [E,T]=stEnergy(x,Fs,winlength,winstep);
   [Zcr,T]=stZeroCrossingRate(x,Fs,winlength,winstep);
   protoFSeq{i}=[E;Zcr];
end
```

which performs feature extraction per prototype and uses a single cell array (*protoFSeq*) to hold the feature sequences from all prototypes. To extract the features, a moving windows technique is employed [Theo 09, Section 7.5.1]. The length of the moving windows is equal to 20 milliseconds and there is no overlap between successive windows.

To find the best match for an unknown pattern—for example, a pattern stored in file *upattern1.wav*—type the following code:

```
[test,Fs,bits]=wavread('upattern1');
winlength = round(0.02*Fs); % use the same values as before
winstep = winlength;
[E,T]=stEnergy(test,Fs,winlength,winstep);
[Zcr,T]=stZeroCrossingRate(test,Fs,winlength,winstep);
Ftest=[E;Zcr];

tolerance=0.1;
LeftEndConstr=round(tolerance/winstep); % left endpoint constraint
RightEndConstr = LeftEndConstr;
for i=1:length(protoNames)
   [MatchingCost(i),BestPath{i},D{i},Pred{i}]=DTWSakoeEndp(...
        protoFSeq{i},Ftest,LeftEndConstr,RightEndConstr,0);
end
```

```
[minCost,indexofBest]=min(MatchingCost);
fprintf('The unknown pattern has been identified as %s \n',...
        protoNames{indexofBest});
```

This code uses the standard Sakoe local constraints and allows for endpoint constraints. Specifically, it is assumed that the length of silence/noise at each endpoint may not exceed 0.1 seconds (i.e., at most 5 frames can be skipped from each endpoint of the test utterance). Note that, instead of using the function *DTWSakoeEndp* to compute the matching cost, we could have used *DTWItakuraEndp*. To avoid repeating the code for each unknown pattern, the whole system is available on the website as a single m-file under the name *IsoDigitRec.m*.

Remarks

- Errors may occur in the experiments. This is also expected in practice, and is even more true here since only two features have been used for pedagogic simplicity.
- Moreover, this a speaker-dependent speech recognition example. Hence, if you record your own voice and test the system using the provided prototypes, the performance may not to be a good one, especially if the accent of the speaker is very different from the accent we used to record the prototypes. You can reconstruct the whole example by using your own voice for both the test data and the prototypes.

CHAPTER

Data Transformation
Feature Generation and Dimensionality Reduction

3

3.1 INTRODUCTION

In this chapter, we deal with linear and nonlinear transformation techniques, which are used to generate a set of features from a set of measurements or from a set of originally generated features. The goal is to obtain new features that encode the classification information in a more compact way compared with the original features. This implies a reduction in the number of features needed for a given classification task, which is also known as *dimensionality reduction* because the dimension of the new feature space is now reduced. The goal, of course, is to achieve this dimensionality reduction in some optimal sense so that the loss of information, which in general is unavoidable after reducing the original number of features, is as small as possible.

3.2 PRINCIPAL COMPONENT ANALYSIS

Principal component analysis (PCA) is one of the most popular techniques for dimensionality reduction. Starting from an original set of l samples (features), which form the elements of a vector $x \in \mathcal{R}^l$, the goal is to apply a linear transformation to obtain a new set of samples:

$$y = A^T x$$

so that the components of y are uncorrelated. In a second stage, one chooses the most significant of these components. The steps are summarized here:

1. Estimate the covariance matrix S. Usually the mean value is assumed to be zero, $E[x] = 0$. In this case, the covariance and autocorrelation matrices coincide, $R \equiv E[xx^T] = S$. If this is not the case, we subtract the mean. Recall that, given N feature vectors, $x_i \in \mathcal{R}^l$, $i = 1, 2, \ldots, N$, the autocorrelation matrix estimate is given by

$$R \approx \frac{1}{N} \sum_{i=1}^{N} x_i x_i^T \tag{3.1}$$

2. Perform the eigendecomposition of S and compute the l eigenvalues/eigenvectors, λ_i, $a_i \in \mathcal{R}^l$, $i = 0, 2, \ldots, l-1$.

DOI: 10.1016/B978-0-12-374486-9.00003-8

3. Arrange the eigenvalues in descending order, $\lambda_0 \geq \lambda_1 \geq \cdots \geq \lambda_{l-1}$.
4. Choose the m largest eigenvalues. Usually m is chosen so that the gap between λ_{m-1} and λ_m is large. Eigenvalues $\lambda_0, \lambda_1, \ldots, \lambda_{m-1}$ are known as the *m principal components*.
5. Use the respective (column) eigenvectors a_i, $i = 0, 1, 2, \ldots, m-1$ to form the transformation matrix

$$A = \begin{bmatrix} a_0 & a_1 & a_2 & \cdots & a_{m-1} \end{bmatrix}$$

6. Transform each l-dimensional vector x in the original space to an m-dimensional vector y via the transformation $y = A^T x$. In other words, the ith element $y(i)$ of y is the *projection* of x on a_i $\left(y(i) = a_i^T x\right)$.

As pointed out in [Theo 09, Section 6.3], the total variance of the elements of x, $\sum_{i=0}^{l-1} E[x^2(i)]$ (for zero mean), is equal to the sum of the eigenvalues $\sum_{i=0}^{l-1} \lambda_i$. After the transformation, the variance of the ith element, $E[y^2(i)]$, $i = 0, 2, \ldots, l-1$, is equal to λ_i. Thus, selection of the elements that correspond to the m largest eigenvalues retains the maximum variance.

To compute the principal components, type

$$[eigenval, eigenvec, explain, Y, mean\_vec] = pca\_fun(X, m)$$

where

X is an $l \times N$ matrix with columns that are the data vectors,

m is the number of the most significant principal components taken into account,

eigenval is an m-dimensional column vector containing the m largest eigenvalues of the covariance matrix of X in descending order,

eigenvec is an $l \times m$-dimensional matrix, containing in its columns the eigenvectors that correspond to the m largest eigenvalues of the covariance matrix of X,

explain is an l-dimensional column vector whose ith element is the percentage of the total variance retained along (in the MATLAB terminology *explained* by) the ith principal component,

Y is an $m \times N$ matrix containing the projections of the data points of X to the space spanned by the m vectors of *eigenvec*,

mean_vec is the mean vector of the column vectors of X.

Example 3.2.1

1. Generate a set X_1 of $N = 500$ 2-dimensional vectors from a Gaussian distribution with zero mean and covariance matrix

$$S_1 = \begin{bmatrix} 0.3 & 0.2 \\ 0.2 & 1.0 \end{bmatrix}$$

Perform PCA on X_1; that is, compute the two eigenvalues/eigenvectors of the estimate $\hat{S}_1$ of S_1 obtained using the vectors of X_1. Taking into account that the ith eigenvalue "explains" the variance along the direction of the ith eigenvector of $\hat{S}_1$, compute the percentage of the total variance explained

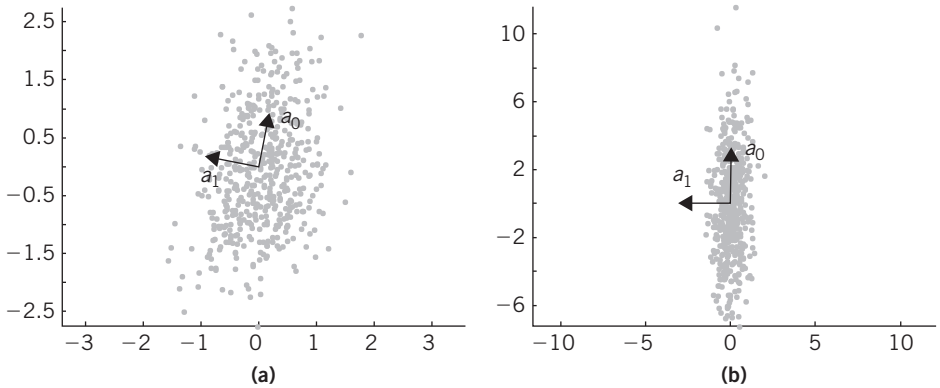

FIGURE 3.1

Data points of X_1(a) and X_2(b) considered in Example 3.2.1, together with the (normalized) eigenvectors of $\hat{S}_1$ and $\hat{S}_2$, respectively.

by each of the two components as the ratio $\frac{\lambda_i}{\lambda_0+\lambda_1}$, $i = 0, 1$. Plot the data set X_1 as well as the eigenvectors of $\hat{S}_1$. Comment on the results.

2. Similarly generate a data set X_2, now with the covariance matrix $S_2 = \begin{bmatrix} 0.3 & 0.2 \\ 0.2 & 9.0 \end{bmatrix}$. Repeat the previous experiment.

Solution. Take the following steps:

Step 1. To generate the data set X_1, type

```
randn('seed',0)  %For reproducubility of the results
S1=[.3 .2; .2 1];
[l,l]=size(S1);
mv=zeros(1,l);
N=500;
m=2;
X1=mvnrnd(mv,S1,N)';
```

To apply PCA on X_1 and to compute the percentage of the total variance explained by each component, type

```
[eigenval,eigenvec,explained,Y,mean_vec]=pca_fun(X1,m);
```

To plot the points of the data set X_1 together with the (normalized) eigenvectors of $\hat{S}_1$, type (see Figure 3.1(a))

```
figure(1), hold on
figure(1), plot(X1(1,:),X1(2,:),'r.')
```

```
figure(1), axis equal
figure(1), line([0; eigenvec(1,1)],[0; eigenvec(2,1)])
figure(1), line([0; eigenvec(1,2)],[0; eigenvec(2,2)])
```

The percentages of the total variance explained by the first and second components are 78.98% and 21.02%, respectively. This means that if we project the points of X_1 along the direction of the principal eigenvector (that corresponds to the largest eigenvalue of $\hat{S}_1$), we retain 78.98% of the total variance of X_1; 21.02% of the total variance, associated with the second principal component, will be "lost."

Step 2. To generate the data set X_2, repeat the previous code, where now X_1 and S_1 are replaced by X_2 and S_2, respectively. In this case, the percentages of the total variance explained by the first and second components are 96.74% and 3.26%, respectively. This means that if we project the points of X_2 along the direction of the eigenvector that corresponds to the largest eigenvalue of $\hat{S}_1$, we retain almost all the variance of X_2 in the 2-dimensional space (see Figure 3.1(b)). Explain this using physical reasoning. ■

The goal of the next example is to demonstrate that projecting in a lower-dimensional space, so as to retain most of the variance, does not necessarily guarantee that the classification-related information is preserved.

Example 3.2.2

1. **a.** Generate a data set X_1 consisting of 400 2-dimensional vectors that stem from two classes. The first 200 stem from the first class, which is modeled by the Gaussian distribution with mean $m_1 = [-8, 8]^T$; the rest stem from the second class, modeled by the Gaussian distribution with mean $m_2 = [8, 8]^T$. Both distributions share the covariance matrix

$$S = \begin{bmatrix} 0.3 & 1.5 \\ 1.5 & 9.0 \end{bmatrix}$$

   b. Perform PCA on X_1 and compute the percentage of the total amount of variance explained by each component.

   c. Project the vectors of X_1 along the direction of the first principal component and plot the data set X_1 and its projection to the first principal component. Comment on the results.
2. Repeat on data set X_2, which is generated as X_1 but with $m_1 = [-1, 0]^T$ and $m_2 = [1, 0]^T$.
3. Compare the results obtained and draw conclusions.

Solution. Take the following steps:

Step 1(a). To generate data set X_1 and the vector y_1, whose ith coordinate contains the class label of the ith vector of X_1, type

```
randn('seed',0)  %For reproducibility of the results
S=[.3 1.5; 1.5 9];
[l,l]=size(S);
mv=[-8 8; 8 8]';
N=200;
```

```
X1=[mvnrnd(mv(:,1),S,N); mvnrnd(mv(:,2),S,N)]';
y1=[ones(1,N), 2*ones(1,N)];
```

Step 1(b). To compute the eigenvalues/eigenvectors and variance percentages required in this step, type

```
m=2;
[eigenval,eigenvec,explained,Y,mean_vec]=pca_fun(X1,m);
```

Step 1(c). The projections of the data points of X_1 along the direction of the first principal component are contained in the first row of Y, returned by the function *pca_ fun*. To plot the data vectors of X_1 as well as their projections, type

```
%Plot of X1
figure(1), hold on
figure(1), plot(X(1,y==1),X(2,y==1),'r.',X(1,y==2),X(2,y==2),'bo')
%Computation of the projections of X1
w=eigenvec(:,1);
t1=w'*X(:,y==1);
t2=w'*X(:,y==2);
X_proj1=[t1;t1].*((w/(w'*w))*ones(1,length(t1)));
X_proj2=[t2;t2].*((w/(w'*w))*ones(1,length(t2)));
%Plot of the projections
figure(1), plot(X_proj1(1,:),X_proj1(2,:),'k.',...
X_proj2(1,:),X_proj2(2,:),'ko')
figure(1), axis equal
%Plot of the eigenvectors
figure(1), line([0; eigenvec(1,1)], [0; eigenvec(2,1)])
figure(1), line([0; eigenvec(1,2)], [0; eigenvec(2,2)])
```

The percentages of the total amount of variance explained by the first and second components are 87.34% and 12.66%, respectively. That is, the projection to the direction of the first component retains most of the total variance of X_1.

Step 2(a). To generate X_2, the code for X_1 is executed again, with $m_1 = [-1, 0]^T$ and $m_2 = [1, 0]^T$.

Step 2(b)–(c). The codes in the (b)–(c) branches of step 1 are executed for X_2. The percentages of the total amount of variance explained by the two principal components are 90.19% and 9.81%, respectively.

Step 3. In the two previous cases approximately 90% of the total variance of the data sets is retained after projecting along the first principal component. However, there is no guarantee that class discrimination is retained along this direction. Indeed, in the case of X_1 the data in the two classes, after projection along the first principal eigenvector, remain well separated. However, this is not the case for data set X_2 (see Figure 3.2).

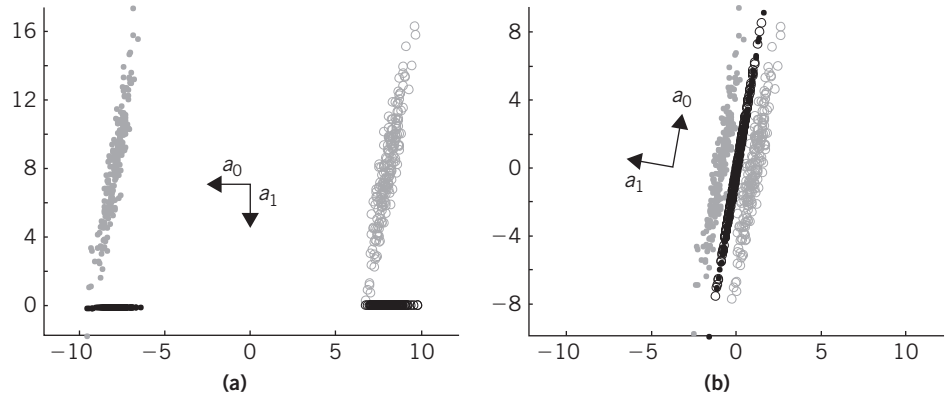

FIGURE 3.2

Data points (gray) of X_1 (a) and X_2 (b) considered in Example 3.2.2, along with the (normalized) eigenvectors of $\hat{S}_1$ and $\hat{S}_2$, respectively. The data points of the two classes of X_1, after projection along the first principal eigenvector (black), remain well separated. This is not the case for X_2.

Exercise 3.2.1

Take the following steps:

1. Generate a data set X_1 consisting of 400 3-dimensional vectors that stem from two classes. The first half of them stem from the first class, which is modeled by the Gaussian distribution with mean $m_1 = [-6, 6, 6]^T$; the rest stem from the second class, modeled by the Gaussian distribution with mean $m_2 = [6, 6, 6]^T$. Both distributions share the covariance matrix

$$S = \begin{bmatrix} 0.3 & 1.0 & 1.0 \\ 1.0 & 9.0 & 1.0 \\ 1.0 & 1.0 & 9.0 \end{bmatrix}$$

   Perform PCA on X_1 and compute the percentage of the total amount of variance explained by each principal component. Project the vectors of X_1 on the space spanned by the first two principal components Y_1 and Y_2. Plot the data in the X_{11}-X_{12}, X_{11}-X_{13}, X_{12}-X_{13}, Y_1-Y_2, Y_1-Y_3, Y_2-Y_3 subspaces (six MATLAB figures in total).
2. Generate a data set X_2 as in step 1, now with $m_1 = [-2, 0, 0]^T$ and $m_2 = [2, 0, 0]^T$. Repeat the process as described in step 1.
3. Compare the results obtained from each data set and draw conclusions.

3.3 THE SINGULAR VALUE DECOMPOSITION METHOD

Given an $l \times N$ matrix, X, there exist square *unitary* matrices U and V of dimensions $l \times l$ and $N \times N$, respectively, so that

$$X = U \begin{bmatrix} \Lambda^{\frac{1}{2}} & O \\ O & 0 \end{bmatrix} V^T$$

where Λ is a square $r \times r$ matrix, with $r \leq \min\{l, N\}$ (r is equal to the rank of X). Since matrices U and V are unitary, their column vectors are, by definition, orthonormal and $UU^T = I$ and $VV^T = I$. Matrix $\Lambda^{\frac{1}{2}}$ is given by

$$\Lambda^{\frac{1}{2}} = \begin{bmatrix} \sqrt{\lambda_0} & & & \\ & \sqrt{\lambda_1} & & \\ & & \ddots & \\ & & & \sqrt{\lambda_{r-1}} \end{bmatrix}$$

where λ_i, $i = 0, 2, \ldots, r-1$, are the r *nonzero* eigenvalues of XX^T, which are the same with the eigenvalues of X^TX [Theo 09, Section 6.4], and known as the *singular values* of X. Equivalently, we can write

$$X = \sum_{i=0}^{r-1} \sqrt{\lambda_i} u_i v_i^T \tag{3.2}$$

where u_i, v_i, $i = 0, 1, \ldots, r-1$, are the corresponding eigenvectors of XX^T and X^TX, respectively. Moreover, u_i and v_i, $i = 0, \ldots, r-1$ are the first r column vectors of U and V, respectively. The rest of the column vectors of U and V correspond to zero eigenvalues.

If we retain $m \leq r$ terms in the summation of Eq. (3.2), that is,

$$\hat{X} = \sum_{i=0}^{m-1} \sqrt{\lambda_i} u_i v_i^T \tag{3.3}$$

then $\hat{X}$ is the best approximation (in the Frobenius sense) of X of rank m [Theo 09, Section 6.4].

To compute the Singular Value Decomposition (SVD) of a matrix X, type

$$[U, s, V, Y] = svd\_fun(X, m)$$

where

X is an $l \times N$ matrix whose columns contain the data vectors,

m is the number of the largest singular values that will be taken into account,

U is an $l \times l$ matrix containing the eigenvectors of XX^T in descending order,

s is an r-dimensional vector containing the singular values in descending order,

V is an $N \times N$ matrix containing the eigenvectors of X^TX in descending order,

Y is an $m \times N$ matrix containing the projections of the data points of X on the space spanned by the m leading eigenvectors contained in U.

More on SVD can be found in [Theo 09, Section 6.4].

Exercise 3.3.1

1. Consider the data set X_1 of Exercise 3.2.1. Perform singular value decomposition using *svd_fun*. Then project the vectors of X_1 on the space spanned by the m leading eigenvectors contained in U (that correspond to Y_1 and Y_2). Finally, plot the data in the X_{11}-X_{12}, X_{11}-X_{13}, X_{12}-X_{13}, Y_1-Y_2, Y_1-Y_3, Y_2-Y_3 spaces (six MATLAB figures in total).

2. Repeat for the data set X_2 of Exercise 3.2.1 and compare the results.
   Observe that the results obtained for the SVD case are similar to those obtained in Exercise 3.2.1 for the PCA case (why?).

Example 3.3.1. Generate a data set of $N = 100$ vectors of dimensionality $l = 2000$. The vectors stem from the Gaussian distribution with a mean equal to the l-dimensional zero vector and a diagonal covariance matrix, S, having all of its nonzero elements equal to 0.1 except $S(1,1)$ and $S(2,2)$, which are equal to 10,000. Apply PCA and SVD on the previous data set and draw your conclusions.

Solution. To generate matrix X, containing the vectors of the data set, type

```
N=100;
l=2000;
mv=zeros(1,l);
S=0.1*eye(l);
S(1,1)=10000;
S(2,2)=10000;
randn('seed',0)
X=mvnrnd(mv,S,N)';
```

Note that the data exhibit significant spread along the first two axes, that is, along the vectors $e_1 = [1, \overbrace{0,\ldots,0}^{l-1}]^T$ and $e_2 = [0, 1, \overbrace{0,\ldots,0}^{l-2}]^T$.

To run PCA and SVD on X and to measure the execution time of each method, type

```
%PCA
t0=clock;
m=5;
[eigenval,eigenvec,explain,Y]=pca_fun(X,m);
time1=etime(clock,t0)
'----'
%SVD
t0=clock;
m=min(N,l);
[U,S,V,Y]=svd_fun(X,m);
time2=etime(clock,t0)
```

From the previously obtained results, two conclusions can be drawn.

First, both methods identify (approximately) e_1 and e_2 as the most significant directions. To verify this, compare the first two columns of *eigenvec* produced by PCA with the first two columns of U, that is, $U(:, 1:2)$, produced by SVD, by typing

```
[eigenvec(:,1:2) U(:,1:2)]'
```

Second, provided that enough computer memory is available, PCA will take orders of magnitude more time than SVD (if not enough memory is available, PCA will not run at all). The difference in

the performance of the two methods lies in the fact that in PCA we perform eigendecomposition on the $l \times l$ covariance matrix while in SVD we perform eigendecomposition on the $N \times N$ $X^T X$ and then, with a simple transformation, compute the eigenvectors of XX^T (which may be viewed as a scaled approximation of the autocorrelation matrix). Moreover, it has to be emphasized that, in general for such cases where $N < l$, the obtained estimate of the autocorrelation matrix is not a good one. Such cases, where $N < l$, arise in image-processing applications, in Web mining, in microarray analysis, and the like.

3.4 FISHER'S LINEAR DISCRIMINANT ANALYSIS

In PCA, the dimensionality reduction is performed in an unsupervised mode. Feature vectors are projected on the subspace spanned by the dominant eigenvectors of the covariance (autocorrelation) matrix. In this section, computation of the subspace on which one projects, in order to reduce dimensionality, takes place in a supervised mode. This subspace is also determined via the solution of an eigendecomposition problem, but the corresponding matrix is different.

In the 2-class case, the goal is to search for a *single* direction, w, so that the respective projections y of the l-dimensional feature vectors $x \in \mathcal{R}^l$ maximize Fisher's discriminant ratio.

Fisher's discriminant ratio of a *scalar* feature y in a 2-class classification task is defined as

$$FDR = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2}$$

where μ_1, μ_2 are the mean values of y, and σ_1^2, σ_2^2 are the variances of y in the two classes, respectively. In other words, after the projection on w the goal is for the mean values of the data points in the two classes to be as far apart as possible and for the variances to be as small as possible. It turns out that w is given by the maximum eigenvector of the matrix product $S_w^{-1} S_b$ [Theo 09, Section 5.8], where for two equiprobable classes

$$S_w = \frac{1}{2}(S_1 + S_2)$$

is known as the *within-class scatter matrix*, with S_1, S_2 being the respective covariance matrices. S_b is known as the *between-class scatter matrix*, defined by

$$S_b = \frac{1}{2}(m_1 - m_0)(m_1 - m_0)^T + \frac{1}{2}(m_2 - m_0)(m_2 - m_0)^T$$

where m_0 is the overall mean of the data x in the original $\mathcal{R}^l$ space and m_1, m_2 are the mean values in the two classes, respectively [Theo 09, Section 5.6.3]. It can be shown, however, that in this special 2-class case the eigenanalysis step can be bypassed and the solution directly given by

$$w = S_w^{-1}(m_1 - m_2)$$

In the c-class case, the goal is to find the $m \leq c - 1$ directions (m-dimensional subspace) so that the so-called J_3 criterion, defined as

$$J_3 = \text{trace}\{S_w^{-1} S_b\}$$

is maximized. In the previous equation

$$S_w = \sum_{i=1}^{c} P_i S_i,\ S_b = \sum_{i=1}^{c} P_i (m_i - m_0)(m_i - m_0)^T$$

and the P_i's denote the respective class a priori probabilities. This is a generalization of the FDR criterion in the multiclass case with different a priori probabilities. The m directions are given by the m dominant eigenvectors of the matrix product $S_w^{-1} S_b$.

It must be pointed out that the rank of the S_b matrix is $c - 1$ at the most (although it is given as a sum of c matrices, only $c - 1$ of these terms are independent [Theo 09, Section 5.6.3]. This is the reason that m was upper-bounded by $c - 1$; only the $c - 1$ largest eigenvalues (at most) are nonzero. In some cases, this may be a drawback because the maximum number of features that this method can generate is bounded by the number of classes [Theo 09, Section 5.8].

Example 3.4.1

1. Apply linear discriminant analysis (LDA) on the data set X_2 generated in the second part of Example 3.2.2.
2. Compare the results obtained with those obtained from the PCA analysis.

Solution. Take the following steps:

Step 1. To estimate the mean vectors of each class using the available samples, type

```
mv_est(:,1)=mean(X2(:,y2==1)')';
mv_est(:,2)=mean(X2(:,y2==2)')';
```

To compute the within-scatter matrix S_w, use the *scatter_mat* function, which computes the within class (S_w), the between class (S_b), and the mixture class (S_m) [Theo 09, Section 5.6.3] for a c-class classification problem based on a set of data vectors. This function is called by

```
[Sw,Sb,Sm]=scatter_mat(X2,y2);
```

Since the two classes are equiprobable, the direction w along which Fisher's discriminant ratio is maximized is computed as $w = S_w^{-1}(m_1 - m_2)$. In MATLAB terms this is written as

```
w=inv(Sw)*(mv_est(:,1)-mv_est(:,2))
```

Finally, the projection of the data vectors of X_2 on the direction w as well as the plot of the results is carried out through the following statements (see Figure 3.3)

```
%Plot of the data set
figure(1), plot(X(1,y==1),X(2,y==1),'r.',...
X(1,y==2),X(2,y==2),'bo')
figure(1), axis equal
%Computation of the projections
t1=w'*X(:,y==1);
```

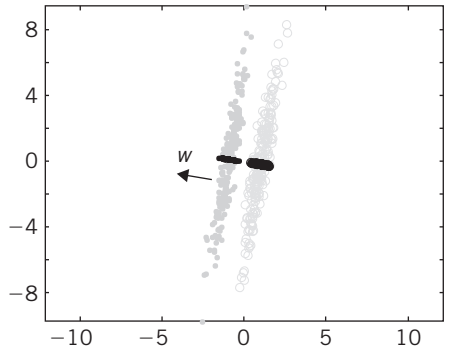

FIGURE 3.3

Points of the data set X_2 (gray) and their projections (black) along the direction of w, from Example 3.4.1.

```
t2=w'*X(:,y==2);
X_proj1=[t1;t1].*((w/(w'*w))*ones(1,length(t1)));
X_proj2=[t2;t2].*((w/(w'*w))*ones(1,length(t2)));
%Plot of the projections
figure(1), hold on
figure(1), plot(X_proj(1,y==1),X_proj(2,y==1),'y.',...
X_proj(1,y==2),X_proj(2,y==2),'co')
```

Step 2. Comparing the result depicted in MATLAB figure 1, which was produced by the execution of the previous code, to the corresponding result obtained by the PCA analysis, it is readily observed that the classes remain well separated when the vectors of X_2 are projected along the w direction that results from Fisher's discriminant analysis. In contrast, classes were heavily overlapped when they were projected along the principal direction provided by PCA.

Example 3.4.2

1a. Generate a data set of 900 3-dimensional data vectors, which stem from two classes—the first 100 vectors from a zero-mean Gaussian distribution with covariance matrix

$$S_1 = \begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 0.01 \end{bmatrix}$$

The rest grouped in 8 groups of 100 vectors. Each group stems from a Gaussian distribution. All of these distributions share the covariance matrix

$$S_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.01 \end{bmatrix}$$

while their means are

- $m_1^2 = [a, 0, 0]^T$
- $m_2^2 = [a/2, a/2, 0]^T$
- $m_3^2 = [0, a, 0]^T$
- $m_4^2 = [-a/2, a/2, 0]^T$
- $m_5^2 = [-a, 0, 0]^T$
- $m_6^2 = [-a/2, -a/2, 0]^T$
- $m_7^2 = [0, -a, 0]^T$
- $m_8^2 = [a/2, -a/2, 0]^T$

where $a = 6$ (m_i^2 denotes the mean of the ith Gaussian distribution of the second class).

Take the following steps:

1b. Plot the 3-dimensional data set and view it from different angles to get a feeling of how the data are spread in the 3-dimensional space (use the Rotate-3D MATLAB utility).

1c. Perform Fisher's discriminant analysis on the previous data set. Project the data on the subspace spanned by the eigenvectors that correspond to the nonzero eigenvalues of the matrix product $S_w^{-1}S_b$. Comment on the results.

2. Repeat step 1 for a 3-class problem where the data are generated like those in step 1, with the exception that the last group of 100 vectors, which stem from the Gaussian distribution with mean m_8^2, is labeled class 3.

Solution. Take the following steps:

Step 1(a). To generate a 3×900-dimensional matrix whose columns are the data vectors, type

```
%Initialization of random number generator
randn('seed',10)
%Definition of the parameters
S1=[.5 0 0; 0 .5 0; 0 0 .01];
S2=[1 0 0; 0 1 0; 0 0 .01];
a=6;
mv=[0 0 0; a 0 0; a/2 a/2 0; 0 a 0; -a/2 a/2 0;...
 -a 0 0; -a/2 -a/2 0; 0 -a 0; a/2 -a/2 0]';
N=100;
% Generation of the data set
X=[mvnrnd(mv(:,1),S1,N)];
for i=2:9
    X=[X; mvnrnd(mv(:,i),S2,N)];
end
X=X';
c=2; %No of classes
y=[ones(1,N) 2*ones(1,8*N)]; %Class label vector
```

Step 1(b). To plot the data set X in the 3-dimensional space, type

```
figure(1), plot3(X(1,y==1),X(2,y==1),X(3,y==1),'r.',...
X(1,y==2),X(2,y==2),X(3,y==2),'b.')
figure(1), axis equal
```

With the Rotate-3D button of MATLAB figure 1, you can view the data set from different angles. It is easy to notice that the variation of data along the third direction is very small (because of the small values of $S_1(3,3)$ and $S_2(3,3)$). The data set in the 3-dimensional space may be considered as lying across the $x-y$ plane, with a very small variation along the z axis.

Clearly, the projection of the data set in the $x-y$ plane retains the separation of the classes, but this is not the case with the projections on the $x-z$ and $y-z$ planes. In addition, observe that there is no single direction (1-dimensional space) w that retains the separation of the classes after projecting X on it.

Step 1(c). To perform Fisher's discriminant analysis, first compute the scatter matrices S_w and S_b; then perform eigendecomposition on the matrix $S_w^{-1}S_b$; finally, project the data on the subspace spanned by the eigenvectors of $S_w^{-1}S_b$ that correspond to the nonzero eigenvalues. The following MATLAB code may be used:

```
% Scatter matrix computation
[Sw,Sb,Sm]=scatter_mat(X,y);
% Eigendecomposition of Swˆ(-1)*Sb
[V,D]=eig(inv(Sw)*Sb);
% Sorting the eigenvalues in descending order
% and rearranging accordingly the eigenvectors
s=diag(D);
[s,ind]=sort(s,1,'descend');
V=V(:,ind);
% Selecting in A the eigenvectors corresponding
% to non-zero eigenvalues
A=V(:,1:c-1);
% Project the data set on the space spanned by
% the column vectors of A
Y=A'*X;
```

Here we used the code for the multiclass case with $c=2$. Since the number of classes is equal to 2, only one eigenvalue of $S_w^{-1}S_b$ is nonzero (0.000234). Thus, Fisher's discriminant analysis gives a single direction (1-dimensional space) along which the data will be projected.

To plot the projections of X on the subspace spanned by the eigenvector of $S_w^{-1}S_b$, which corresponds to the nonzero eigenvalue, type

```
figure(2), plot(Y(y==1),0,'ro',Y(y==2),0,'b.')
figure(2), axis equal
```

Observe that the projections of the data of the two classes coincide. Thus, in this case Fisher's discriminant analysis cannot provide a smaller subspace where the class discrimination is retained. This happens because the number of classes is equal to 2 and so the dimensionality of the reduced subspace is at most 1, which is not sufficient for the current problem.

Step 2(a). To generate a 3×900-dimensional matrix whose columns are data vectors, repeat the code given in step 1(a), replacing the last two lines with

```
% Definition of the number of classes
c=3;
% Definition of the class label of each vector
y=[ones(1,N) 2*ones(1,7*N) 3*ones(1,N)];
```

Step 2(b). To plot the data set X in the 3-dimensional space, type

```
figure(1), plot3(X(1,y==1),X(2,y==1),X(3,y==1),'r.',X(1,y==2),...
X(2,y==2),X(3,y==2),'b.',X(1,y==3),X(2,y==3),X(3,y==3),'g.')
figure(1), axis equal
```

Step 2(c). Adopt the MATLAB code of step 1(c) for the current data set. In this case, since there are three classes, we may have at most two nonzero eigenvalues of $S_w^{-1}S_b$. Indeed, the nonzero eigenvalues now turn out to be 0.222145 and 0.000104.

To plot the projections of X on the space spanned by the eigenvectors of $S_w^{-1}S_b$ that correspond to the nonzero eigenvalues (2-dimensional space), type

```
figure(3), plot(Y(1,y==1),Y(2,y==1),'ro',...
Y(1,y==2),Y(2,y==2),'b.',Y(1,y==3),Y(2,y==3),'gx')
figure(3), axis equal
```

In this case, observe that the projection in the 2-dimensional subspace retains the separation among the classes at a satisfactory level.

Finally, keep in mind that there are data sets where the dimensionality reduction from projection in any subspace of the original space may cause substantial loss of class discrimination. In such cases, nonlinear techniques may be useful. ■

3.5 THE KERNEL PCA

The three methods considered so far for dimensionality reduction are *linear*. A subspace of low dimension is first constructed as, for example, the span of the m dominant directions in the original $\mathcal{R}^l$, $l > m$ space.

The choice of dominant directions depends on the method used. In a second stage, all vectors of interest in $\mathcal{R}^l$ are (linearly) projected in the low-dimensional subspace. Such techniques are appropriate whenever our data in $\mathcal{R}^l$ lie (approximately) on a linear manifold (e.g., hyperplane). However, in many

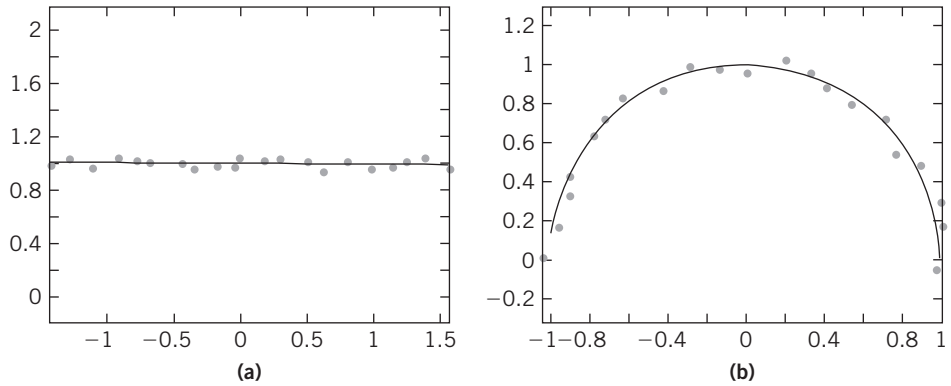

FIGURE 3.4

(a) The 2-dimensional data points lying (approximately) on a line (linear manifold). (b) The 2-dimensional data points lying (approximately) on a semicircle (nonlinear manifold).

cases the data are distributed around a lower-dimensional manifold, which is not linear (e.g., around a circle or a sphere in a 3-dimensional space).

Figures 3.4(a,b) show two cases where data in the 2-dimensional space lie (approximately) on a linear and a nonlinear manifold, respectively. Both manifolds are 1-dimensional since a straight line and the circumference of a circle can be parameterized in terms of a *single* parameter.

The kernel PCA is one technique for dimensionality reduction when the data lie (approximately) on a nonlinear manifold. According to the method, data are first mapped into a high-dimensional space via a *nonlinear* mapping:

$$x \in \mathcal{R}^l \mapsto \phi(x) \in H$$

PCA is then performed in the new space H, chosen to be an RKHS. The inner products can be expressed in terms of the kernel trick, as discussed in Section 2.5.

Although a (linear) PCA is performed in the RKHS space H, because of the nonlinearity of the mapping function $\phi(\cdot)$, the method is equivalent to a nonlinear function in the original space. Moreover, since every operation can be expressed in inner products, the explicit knowledge of $\phi(\cdot)$ is not required. All that is necessary is to adopt the kernel function that defines the inner products. More details are given in [Theo 09, Section 6.7.1].

To use the kernel PCA, type

$$[s, V, Y] = kernel\_PCA(X, m, choice, para)$$

where

X is an $l \times N$ matrix whose columns contain the data vectors,

m is the number of (significant) principal components that will be considered,

choice is the type of kernel function to be used ('*pol*' for polynomial, '*exp*' for exponential),

para is a 2-dimensional vector containing the parameters for the kernel function; for polynomials it is $(x^T y + para(1))^{para(2)}$ and for exponentials it is $exp(-(x-y)^T(x-y)/(2para(1)^2))$,

s is an N-dimensional vector that contains the computed eigenvalues after applying the kernel PCA,

V is an $N \times N$ matrix whose columns are the eigenvectors corresponding to the principal components of the Gram matrix, $\mathcal{K}$, which is involved in the kernel PCA [Theo 09, Section 6.7.1],

Y is an $m \times N$ dimensional matrix that contains the projections of the data vectors of X on the subspace spanned by the m principal components.

Example 3.5.1. This example illustrates the rationale behind the kernel PCA. However, since kernel PCA implies, first, a mapping to a higher-dimensional space, visualization of the results is generally not possible. Therefore, we will "cheat" a bit and use a mapping function $\phi(\cdot)$ that does not correspond to a kernel function $k(\cdot,\cdot)$. (After all, the mapping to an RKHS is required only for the computational tractability needed to compute inner products efficiently.) However, this function allows transformation of a 2-dimensional space, where our data points lie around a nonlinear manifold, into another 2-dimensional space, where the data points are mapped around a linear manifold.

Consider a data set X consisting of 21 2-dimensional points of the form $x_i = (x_i(1), x_i(2)) = (\cos\theta_i + s_i, \sin\theta_i + s_i')$, where $\theta_i = (i-1) * (\pi/20)$, $i = 1,\ldots,21$ and s_i, s_i' are random numbers that stem from the uniform distribution in $[-0.1,\ 0.1]$ (see Figure 3.5(a)). These points lie around the semicircle modeled by $x^2(1) + x^2(2) = 1$, which is centered at the origin and is positive along the x_2 axis.

The mapping function $\phi(\cdot)$ is defined as

$$\phi\left(\begin{bmatrix} x(1) \\ x(2) \end{bmatrix}\right) = \begin{bmatrix} \tan^{-1}\left(\frac{x(2)}{x(1)}\right) \\ \sqrt{x^2(1) + x^2(2)} \end{bmatrix}$$

By applying $\phi(\cdot)$ on the data set X, we get the set $Y = \{y_i = \phi(x_i),\ i = 1,\ldots,21\}$, which is illustrated in Figure 3.5(b). Note that the points of Y lie around a linear manifold (straight line) in the transformed domain. Then we apply linear PCA on Y and keep only the first principal component, since almost all of the total variance of the data set (99.87%) is retained along this direction.[1] Let $Z = \{z_i = [z_i(1), z_i(2)]^T,\ i = 1,\ldots,21\}$ be the set containing the projections of y_i's on the first principal component in the transformed space (see Figure 3.5(c)). Mapping z_i's back to the original space via the inverse function of $\phi(\cdot)$, which is given by

$$\phi^{-1}\left(\begin{bmatrix} z(1) \\ z(2) \end{bmatrix}\right) = \begin{bmatrix} z(2)\cos z(1) \\ z(2)\sin z(1) \end{bmatrix} \equiv \begin{bmatrix} x'(1) \\ x'(2) \end{bmatrix}$$

the points $(x'(1), x'(2))$ are obtained that lie on the semicircle defined by $x^2(1) + x^2(2) = 1$, with $x(2) > 0$ (see Figure 3.5(d)). ■

[1] Linear PCA requires subtraction of the mean of the data vectors (which is performed in the *pca_fun* function). In our case, this vector equals $[0, 1]^T$. After PCA, this vector is added to each projection of the points along the direction of the first principal component.

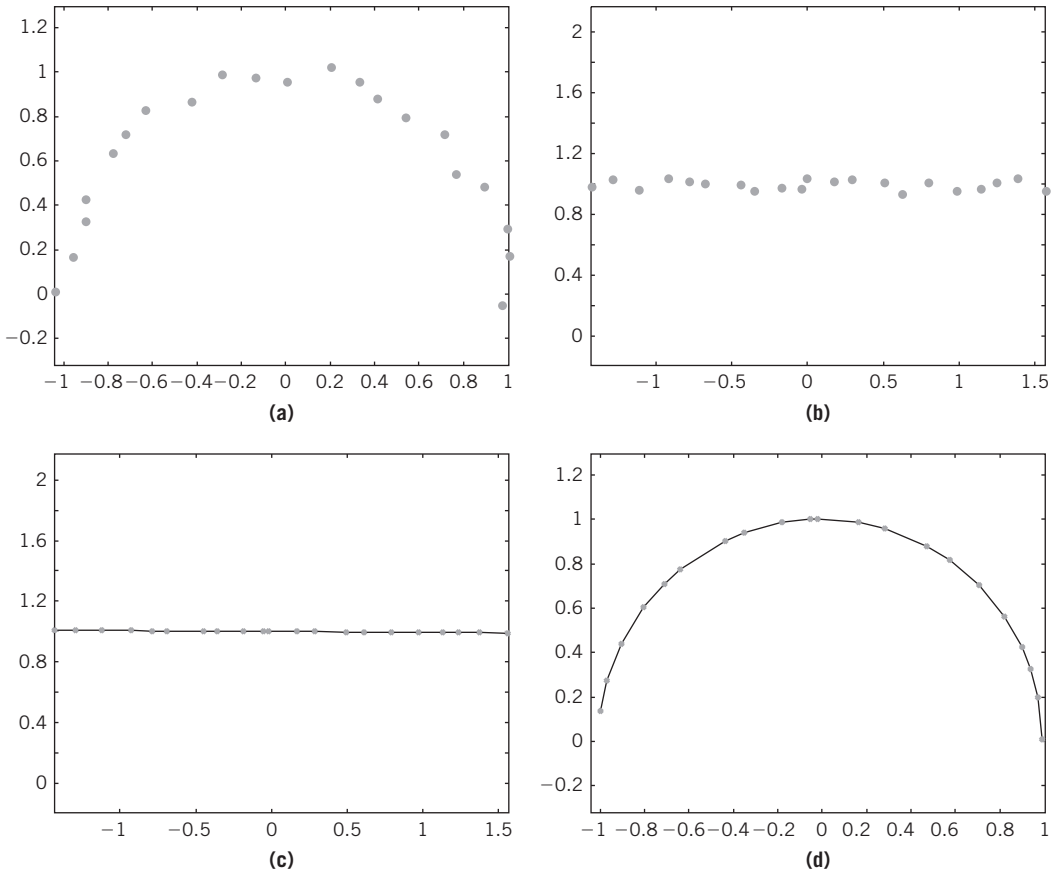

FIGURE 3.5

Example 3.5.1: (a) Data set in the original space (around a semicircle). (b) Data set in the transformed space (around a straight line). (c) Direction corresponding to the first principal component and the images (projections) of the points on it in the transformed space. (d) Images of the points in the original space.

Remark

- Observe that the nonlinear mapping transformed the original manifold to a linear one. Thus, the application of the (linear) PCA on the transformed domain is fully justified. Of course, in the general case one should not expect to be so "lucky"—that is, to have the transformed data lying across a linear manifold.

In the next example we consider kernel PCA in the context of a classification task. More specifically, the goal is to demonstrate the potential of the kernel PCA to transform a nonlinear classification problem, in the (original) l-dimensional space, into a linear one in an $m(< l)$ dimensional space. If this is achieved, the original classification problem can be solved in the transformed space by a linear classifier.

Example 3.5.2

1. Generate two data sets X and X_{test}, each one containing 200 3-dimensional vectors. In each, the first $N_1 = 100$ vectors stem from class 1, which is modeled by the uniform distribution in $[-0.5,\ 0.5]^3$, while the rest, $N_2 = 100$, stem from class -1 and lie around the sphere with radius $r = 2$ and centered at the origin. The N_2 points for each data set are generated as follows.

   Randomly select a pair of numbers, $x(1)$ and $x(2)$, that stem from the uniform distribution in the range $[-2, 2]$, and check if $x^2(1) + x^2(2)$ is less than r^2. If this is not the case, choose a different pair. Otherwise, generate two points of the sphere as $(x(1), x(2), \sqrt{r^2 - x^2(1) - x^2(2)} + \varepsilon_1)$ and $(x(1), x(2), -\sqrt{r^2 - x^2(1) - x^2(2)} + \varepsilon_2)$, where ε_1 and ε_2 are random numbers that stem from the uniform distribution in the interval $[-0.1,\ 0.1]$.

   Repeat this procedure $N_2/2$ times to generate N_2 points. Also generate the vectors y and y_{test} which contain the class labels of the points of X and X_{test}, respectively. Then plot the data sets.
2. Perform kernel PCA on X using the exponential kernel function with $\sigma = 1$ and keep only the first two most significant principal components. Project the data points of X onto the subspace spanned by the two principal components and let Y be the set of these projections (plot Y).
3. Design a least squares (LS) classifier based on Y.
4. Evaluate the performance of the previous classifier based on X_{test} as follows: For each vector in $x \in X_{test}$, determine its projection onto the space spanned by the two most significant principal components, computed earlier, and classify it using the LS classifier generated in step 3. Assign x to the class where its projection has been assigned. Plot the projections of the points of X_{test} onto the subspace spanned by the two principal components along with the straight line that realizes the classifier.
5. Repeat steps 2 through 4 with $\sigma = 0.6$.

Solution. Take the following steps:

Step 1. To generate the points of X that belong to class 1, type

```
rand('seed',0)
noise_level=0.1;
n_points=[100 100];  %Points per class
l=3;
X=rand(l,n_points(1))- (0.5*ones(l,1))*ones(1,n_points(1));
```

To generate the points of X that belong to class -1, type

```
r=2; %Radius of the sphere
for i=1:n_points(2)/2
    e=1;
    while(e==1)
        temp=(2*r)*rand(1,l-1)-r;
        if(r^ 2-sum(temp.^ 2)>0)
```

```
            e=0;
        end
    end
    t=sqrt(r^2-sum(temp.^2))+noise_level*(rand-0.5);
    qw=[temp t; temp -t]';
    X=[X qw];
end
```

The data set X_{test} is generated similarly (use the value 100 as the seed for the *rand* function). To define the class labels of the data vectors, type

```
[l,N]=size(X);
y=[ones(1,n_points(1)) -ones(1,n_points(2))];
y_test=[ones(1,n_points(1)) -ones(1,n_points(2))];
```

To plot the data set X, type[2]

```
figure(1), plot3(X(1,y==1),X(2,y==1),X(3,y==1),'r.',...
X(1,y==-1),X(2,y==-1),X(3,y==-1),'b+')
figure(1), axis equal
```

X_{test} is plotted similarly. Clearly, the two classes are nonlinearly separable.

Step 2. To perform kernel PCA with kernel exponential and $\sigma = 1$, type

```
[s,V,Y]=kernel_PCA(X,2,'exp',[1 0]);
```

Note that Y contains in its columns the images of the points of X on the space spanned by the first two principal components, while V contains the respective principal components.

To plot Y, type

```
figure(2), plot(Y(1,y==1),Y(2,y==1),'r.',Y(1,y==-1),Y(2,y==-1),'b+')
```

Step 3. To design the LS classifier based on Y, type

```
w=SSErr([Y; ones(1,sum(n_points))],y,0);
```

Note that each column vector of Y has been augmented by 1. The resulting w is [33.8001, 2.4356, −0.8935].

Step 4. Type the following to generate the Y_{test} set, containing in its columns the projections of the vectors of X_{test} to the space spanned by the principal components:

```
[l,N_test]=size(X_test);
```

[2]Use the Rotate 3D button to observe the data set from different angles.

```
Y_test=[];
for i=1:N_test
    [temp]=im_point(X_test(:,i),X,V,2,'exp',[1 0]);
    Y_test=[Y_test temp];
end
```

To classify the vectors of X_{test} (Y_{test}) and compute the classification error, type

```
y_out=2*(w'*[Y_test; ones(1,sum(n_points))]>0)-1;
class_err=sum(y_out.*y_test<0)/sum(n_points);
```

Figure 3.6(a) shows the Y_{test} set together with the line that corresponds to the linear classifier. This is produced by typing

```
figure(6), plot(Y_test(1,y==1),Y_test(2,y==1),'r.',...
Y_test(1,y==-1),Y_test(2,y==-1),'b+')
figure(6), axis equal
% Ploting the linear classifier (works only if w(1)~=0)
y_lin=[min(Y_test(2,:)') max(Y_test(2,:)')];
x_lin=[(-w(3)-w(2)*y_lin(1))/w(1) (-w(3)-w(2)*y_lin(2))/w(1)];
figure(6), hold on
figure(6), line(x_lin,y_lin)
```

Step 5. For this step, repeat the codes given in steps 2 through 4. Now in the call of the kernel PCA function (step 2), [1, 0] is replaced by [0.6, 0] (see also Figure 3.6(b)).

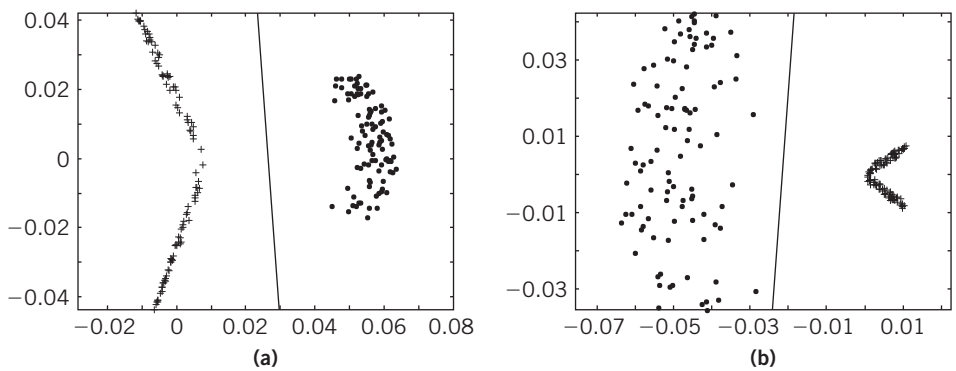

FIGURE 3.6

Y_{test} set produced in step 4 and linear classifier determined in step 3 of Example 3.5.2 for $\sigma = 1$ (a) and $\sigma = 0.6$ (b).

The previous steps having been performed, three conclusions can be drawn:

- First, kernel PCA *may* lead to mappings in lower-dimensional spaces, where the involved classes can be linearly separated, even though this is not the case in the original space. This cannot be achieved with linear PCA.
- Second, the choice of the kernel function parameters is critical. To verify this, try step 5 with $\sigma = 3$. In this case, the arrangement of the classes in the transformed space looks very similar to the arrangement in the original space.
- Third, for this problem the two classes remain linearly separable even in the 1-dimensional space as defined by the first principal component.

Example 3.5.3

1. Generate two data sets X_1 and X_2 as follows:
   - X_1 consists of 300 2-dimensional points. The first $N_1 = 100$ points stem from class 1, which is modeled by the uniform distribution in the region $[-2.5, -1.5] \times [-0.5, 0.5]$; the next $N_2 = 100$ points stem from class 2, which is modeled by the uniform distribution in the area $[0.5, 1.5] \times [-2.5, -1.5]$; and the final, $N_3 = 100$ points stem from class 3 and lie around the circle C with radius $r = 4$ and centered at the origin.

     More specifically, the last N_3 points are generated as follows: The points of the form $x_i = -r + \frac{2r}{N_2/2-1} i$, $i = 0, \ldots, N_2/2 - 1$ in the x axis are selected (they lie in the interval $[-2, 2]$). For each x_i the quantities $v_i^1 = \sqrt{r^2 - x_i^2} + \varepsilon_i^1$ and $v_i^2 = -\sqrt{r^2 - x_i^2} + \varepsilon_i^2$ are computed, where $\varepsilon_i^1, \varepsilon_i^2$ stem from the uniform distribution in the interval $[-0.1,\ 0.1]^T$. The points (x_i, v_i^1) and (x_i, v_i^2), $i = 0, \ldots, N_2 - 1$, lie around C.
   - X_2 consists of 300 2-dimensional points. Points $N_1 = 100$, $N_2 = 100$, and $N_3 = 100$ stem from classes 1, 2, and 3, respectively. They lie around the circles centered at the origin and having radii $r_1 = 2$, $r_2 = 4$, and $r_3 = 6$, respectively (the points of each class are generated by adopting the procedure used for generating the last N_3 points of the previous data set, X_1).

   Generate the vectors y_1 and y_2, which contain the class labels of the data points of the sets X_1 and X_2, respectively. Plot X_1 and X_2.
2. Perform kernel PCA on X_1, using the exponential kernel function with $\sigma = 1$ and keep only the first two principal components. Determine and plot the set Y_1, which contains the projections of the data points of X_1 onto the subspace spanned by the two principal components. Repeat the steps for X_2 and draw conclusions.

Solution. Take the following steps:

Step 1. To generate the data set X_1, type

```
rand('seed',0)
noise_level=0.1;
%%%
n_points=[100 100 100];
```

```
l=2;
% Production of the 1st class
X1=rand(l,n_points(1))- [2.5 0.5]'*ones(1,n_points(1));
% Production of the 2nd class X1=[X1
rand(l,n_points(2))- [-0.5 2.5]'*ones(1,n_points(2))];
% Production of the 3rd class
c1=[0 0];
r1=4;
b1=c1(1)-r1;
b2=c1(1)+r1;
step=(b2-b1)/(n_points(2)/2-1);
for t=b1:step:b2
    temp=[t c1(2)+sqrt(r1^2-(t-c1(1))^2)+noise_level*(rand-0.5);...
    t c1(2)-sqrt(r1^2-(t-c1(1))^2)+noise_level*(rand-0.5)]';
    X1=[X1 temp];
end
```

To generate the vector of the labels y_1, type

```
y1=[ones(1,n_points(1)) 2*ones(1,n_points(2)) 3*ones(1,n_points(3))];
```

To plot the data set X_1, type

```
figure(1), plot(X1(1,y1==1),X1(2,y1==1),'r.',...
X1(1,y1==2),X1(2,y1==2),'bx',...
X1(1,y1==3),X1(2,y1==3),'go')
```

Step 2. To perform kernel PCA on X_1 with the exponential kernel function ($\sigma = 1$), type

```
m=2;
[s,V,Y1]=kernel_PCA(X1,m,'exp',[1 0]);
```

To plot Y_1, type

```
figure(2), plot(Y1(1,y1==1),Y1(2,y1==1),'r.',...
Y1(1,y1==2),Y1(2,y1==2),'bx',...
Y1(1,y1==3),Y1(2,y1==3),'go')
```

Work similarly for X_2.

In Y_1 the classes are linearly separable, but this is not the case in Y_2. That is, kernel PCA does not necessarily transform nonlinear classification problems into linear ones. ■

Exercise 3.5.1

In this exercise, we see how the original space is transformed using kernel PCA. To ensure visualization of the results, we consider a 2-dimensional problem, which will be mapped to the 2-dimensional space spanned by the first two principal components that result from the kernel PCA. More specifically, go through the following steps:

Step 1. Generate a data set X, which contains 200 2-dimensional vectors. $N_1 = 100$ vectors stem from class 1, which is modeled by the uniform distribution in $[-0.5, 0.5]^2$; $N_2 = 100$ vectors stem from class -1 and lie around the circle with radius 1 and centered at the origin. The N_2 points are produced as the last N_3 points of the X_1 data set in Example 3.5.3.

Step 2. Repeat step 1 from Example 3.5.2, with $\sigma = 0.4$.

Step 3. Repeat step 2 from Example 3.5.2.

Step 4. Consider the points x of the form $(-2 + i * 0.1, -2 + j * 0.1)$, $i,j = 0,\ldots,40$ (these form a rectangular grid in the region $[-2,2]^2$) and their images in the space spanned by the two principal components, determined in step 2. Classify the image of each point x using the LS classifier designed in step 3 and produce two figures. The first corresponds to the transformed space and an image point is plotted with a green o if it is classified to the first class $(+1)$ and with a cyan x if it is classified to the second class (-1). The second figure corresponds to the original space and the corresponding points x are drawn similarly. Observe how the implied nonlinear transformation deforms the space.

Step 5. Repeat steps 2 through 4, with $\sigma = 1$.

Hint

To generate X, work as in Example 3.5.3. To define the class labels of the data vectors and to plot X, work as in Example 3.5.2. To perform kernel PCA and define the linear classifier, also work as in Example 3.5.2. To carry out step 4, use the MATLAB function *plot_orig_trans_kPCA* by typing

```
m=2;
choice='exp';
para=[0.4 0];
reg_spec=[-2 0.1 2; -2 0.1 2];
fig_or=5;
fig_tr=6;
plot_orig_trans_kPCA(X,V,m,choice,para,w,reg_spec,fig_or,fig_tr)
```

The results are shown in Figure 3.7. Observe how the points inside the circle (denoted by o) in the original space are expanded in the transformed space. Also notice the difference between the transformed spaces for $\sigma = 0.4$ and $\sigma = 1$, respectively (Figure 3.7(a, c)).

3.6 LAPLACIAN EIGENMAP

The Laplacian eigenmap method belongs to the family of so-called graph-based methods for dimensionality reduction. The idea is to construct a graph, $G(V, E)$, where nodes correspond to the data points x_i, $i = 1, 2, \ldots, N$. If two points are close, the corresponding nodes are connected via an edge, which is weighted accordingly. The closer two points are the higher the value of the weight of the respective edge.

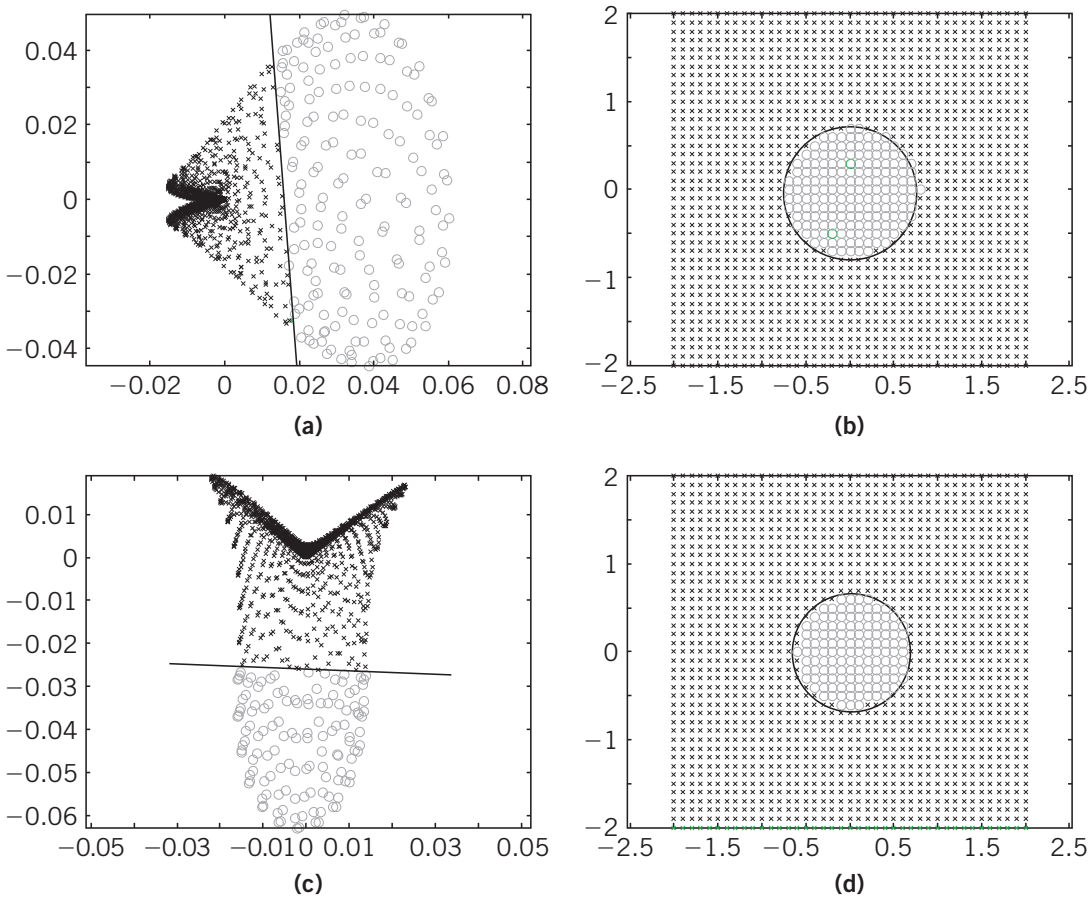

FIGURE 3.7

(a) Linear classifier in the space spanned by the first two principal components resulting from the kernel PCA, using exponential kernel function with $\sigma = 0.4$, from Exercise 3.5.1. (b) Equivalent of the linear classifier in the original space. (c) Linear classifier in the space spanned by the first two principal components using the exponential kernel function with $\sigma = 1$, from Exercise 3.5.1. (d) Equivalent of the linear classifier in the original space. Observe the influence of the value of σ.

Closeness is determined via a threshold value. For example, if the squared Euclidean distance between two points, $||x_i - x_j||^2$, is less than a user-defined threshold, say e, the respective nodes are connected and are said to be *neighbors*. The corresponding weight can be defined in different ways. A common choice for the weights $W(i, j)$ between the nodes i and j is, for some user-defined variable σ^2,

$$W(i,j) = \begin{cases} \exp\left(-\frac{||x_i - x_j||^2}{\sigma^2}\right), & \text{if } ||x_i - x_j||^2 < e \\ 0 & \text{otherwise} \end{cases}$$

Thus, the graph encodes the *local information* in the original, high-dimensional, space $\mathcal{R}^l$.

It is further assumed that the data $x_i \in \mathcal{R}^l$ lie on a *smooth* manifold of dimension m. The value of m is assumed to be known. For example, the original data may live in the 3-dimensional space $\mathcal{R}^3$, but lie around a sphere. The latter is a 2-dimensional smooth manifold since two parameters suffice to describe the surface of a sphere.

The goal of the method is to obtain an m-dimensional representation of the data so that the *local neighborhood information* in the manifold is *optimally* retained. In this way, the local geometric structure of the manifold is reflected in the obtained solution.

The Laplacian eigenmap turns out to be equivalent to an eigenvalue/eigenvector problem of the so-called *Laplacian matrix*. This matrix encodes the local information as described by the weights of the edges in the graph [Theo 09, Section 6.7.2].

To use the Laplacian eigenmap, type

$$y = lapl\_eig(X, e, sigma2, m)$$

where

e is the value of the threshold,

$sigma2$ is the σ^2 parameter,

y is an $m \times N$ matrix whose ith column defines the projection of the ith data vector to the m-dimensional subspace.

Example 3.6.1. Generate a 3-dimensional Archimedes spiral as a pack of 11 identical 2-dimensional Archimedes spirals, one above the other. A 2-dimensional spiral is described in polar coordinates by the equation $r = a\theta$, where a is a user-defined parameter. In our case, the points of a 3-dimensional spiral are generated as follows: For θ, take the values from θ_{init} to θ_{fin} with step θ_{step} and compute

$$r = a\theta$$
$$x = r\cos\theta$$
$$y = r\sin\theta$$

The 11 points of the form (x, y, z), where $z = -1, -0.8, -0.6, \ldots, 0.8, 1$, are points of the spiral. Use $a = 0.1$, $\theta_{init} = 0.5$, $\theta_{fin} = 2.05 * \pi$, $\theta_{step} = 0.2$.

Plot the 3-dimensional spiral so that all points of the same 2-dimensional spiral are plotted using the same symbol and all groups of 11 points of the form (x, y, z), where x and y are fixed and z takes the values $-1, -0.8, -0.6, \ldots, 0.8, 1$, are plotted using the same color.

1. Perform the Laplacian eigenmap on the points of the previous data set for manifold dimension $m = 2$. Plot the results.
2. Perform linear PCA on the same data set, using the first two principal components. Plot the results.
3. Compare the results obtained in steps 1 and 2.

Solution. To generate and plot the 3-dimensional spiral, call the function *spiral_3D* by typing

```
a=0.1;
init_theta=0.5;
```

```
fin_theta=2.05*pi;
step_theta=0.2;
plot_req=1; % Request for plot the spiral
fig_id=1;   % Number id of the figure
% Producing the 3D spiral
[X,color_tot,patt_id]=spiral_3D(a,init_theta,...
fin_theta,step_theta,plot_req,fig_id);
[l,N]=size(X);
```

Use Rotate 3D to see the spiral from different viewpoints.

Do the following:

Step 1. To perform the Laplacian eigenmap, call the function *lapl_eig* by typing

```
e=0.35;
sigma2=sqrt(0.5);
m=2;
y=lapl_eig(X,e,sigma2,m);
```

To plot the results, type

```
figure(2), hold on
for i=1:N
    figure(2), plot(y(1,i),y(2,i),patt_id(i),'Color',color_tot(:,i)')
end
```

Step 2. To perform linear PCA, call the function *pca_fun* by typing:

```
[eigenval,eigenvec,explain,Y]=pca_fun(X,m);
```

To plot the results, type

```
figure(3), hold on
for i=1:N
    figure(3), plot(Y(1,i),Y(2,i),patt_id(i),'Color',color_tot(:,i)')
end
```

Step 3. Observing MATLAB figure 1 on the computer screen, notice how the color of each 2-dimensional spiral varies from red to yellow to green to cyan to blue. On the mapping produced by the Laplacian eigenmap method (MATLAB figure 2), the following two comments are in order: Each 2-dimensional spiral is "stretched" so that the points are placed on a line segment (horizontal direction); in each vertical direction the succession of the colors observed in MATLAB figure 1 is the same as that observed in MATLAB figure 2. Thus, for the given choice of parameters for the Laplacian eigenmap, the method successfully "unfolds" the 3-dimensional spiral.

The linear PCA (MATLAB figure 3) also succeeds in "stretching" each 2-dimensional spiral so that the points are placed on a line segment (horizontal direction). However, the succession of colors

in the vertical direction is not the same as that observed in MATLAB figure 1 (green, yellow, red, cyan, blue). This indicates that after the projection produced by PCA, points that are distant from each other in the 3-dimensional space lie close in the 2-dimensional representation (see, for example, the red and cyan points in the original 3-dimensional space and in the reduced 2-dimensional space).

Finally, the results obtained by the Laplacian eigenmap are sensitive to the choice of parameters. If, for example, the previous experiment is performed for $e = 0.5$ and $sigma2 = 1$, the Laplacian eigenmap fails to completely "unfold" the spiral (in this case, red is close to green). However, this unfolding, although not perfect, is still better than the one produced by linear PCA. In general, extensive experimentation is required before choosing the right values for the parameters involved in the Laplacian eigenmap method.

Exercise 3.6.1

1. Generate a "cut cylinder" of radius $r = 0.5$ in the 3-dimensional space as a pack of 11 identical "cut circles," one above the other, as in Example 3.6.1. For the jth circle, with center $c_j = [c_{j1}, c_{j2}]^T$, the following points are selected:
   - $(x_i, c_{j2} + \sqrt{r^2 - (x_i - c_{j1})^2})$ for x_i ranging from $c_{j1} - r$ to $c_{j1} + r$, with step $(2r)/(N-1)$.
   - $(x_i, c_{j2} - \sqrt{r^2 - (x_i - c_{j1})^2})$ for x_i ranging from $c_{j1} + r$ down to $(c_{j1} - r)/4$, with the previously chosen step, where N is the number of points on which x_i is sampled in the range $[c_{j1} - r, c_{j1} + r]$.

   Plot the cut cylinder so that all points of the same 2-dimensional cut circle are plotted using the same symbol, and all groups of 11 points of the form (x, y, z), where x and y are fixed and z takes the values $-1, -0.8, -0.6, \ldots, 0.8, 1$ are plotted using the same color.
2. Perform the Laplacian eigenmap on the points of the previous data set for manifold dimension $m = 2$. Plot the results.
3. Perform linear PCA on the same data set using the first two principal components and plot the results.
4. Compare the results obtained in steps 2 and 3.

Hints

1. To generate and plot the 3-dimensional cut cylinder, call the function *cut_cylinder_3D* by typing

```
r=0.5;
center=[0 0];
N=25;
plot_req=1; %Request for plot the cylinder
fig_id=1;   %Number id of the figure
% Producing the 3D cut cylinder
[X,color_tot,patt_id]=cut_cylinder_3D(r,center,N,plot_req,...
fig_id);
[l,N]=size(X);
```

   Use Rotate 3D to observe the cut cylinder from different viewpoints.
2. Apply step 1 of Example 3.6.1, using the same parameters for the Laplacian eigenmap method.

3. Apply step 2 of Example 3.6.1.
4. Observe that, once again, the Laplacian eigenmap gives better results compared with those of the linear PCA. However, this is not the case when the cut cylinder has a radius equal to 5. (Why? Compare the height of the cylinder with its radius.)

Information Sciences 277 (2014) 755–776

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Reduction target structure-based hierarchical attribute reduction for two-category decision-theoretic rough sets

Xianyong Zhang [a,b,c,*], Duoqian Miao [b,c]

[a] College of Mathematics and Software Science, Sichuan Normal University, Chengdu 610068, PR China
[b] Department of Computer Science and Technology, Tongji University, Shanghai 201804, PR China
[c] Key Laboratory of Embedded System and Service Computing, Ministry of Education, Shanghai 201804, PR China

ARTICLE INFO

Article history:
Received 6 November 2012
Received in revised form 22 February 2014
Accepted 28 February 2014
Available online 19 March 2014

Keywords:
Rough set theory
Attribute reduction
Decision-theoretic rough set
Granular computing
Consistency-preservation
Structure target

ABSTRACT

Attribute reduction is an essential subject in rough set theory, but because of quantitative extension, it becomes a problem when considering probabilistic rough set (PRS) approaches. The decision-theoretic rough set (DTRS) has a threshold semantics and decision feature and thus becomes a typical and fundamental PRS. Based on reduction target structures, this paper investigates hierarchical attribute reduction for a two-category DTRS and is divided into five parts. (1) The knowledge-preservation property and reduct are explored by knowledge coarsening. (2) The consistency-preservation principle and reduct are constructed by a consistency mechanism. (3) Region preservation is analyzed, and the separability between consistency preservation and region preservation is concluded; thus, the double-preservation principle and reduct are studied. (4) Structure targets are proposed by knowledge structures, and an attribute reduction is further described and simulated; thus, general reducts are defined to preserve the structure targets or optimal measures. (5) The hierarchical relationships of the relevant four targets and reducts are analyzed, and a decision table example is provided for illustration. This study offers promotion, rationality, structure, hierarchy and generalization, and it establishes a fundamental reduction framework for two-category DTRS. The relevant results also provide some new insights into the attribute reduction problem for PRS.

1. Introduction

Rough set theory (RS theory) can effectively address data analysis problems that involve uncertain, imprecise or incomplete information; Refs. [6,11,35,55,56] report relevant developments in these respects. The classical model (Pawlak model) [28,29] is a qualitative model that does not involve quantitative information. Thus, quantitative and extended models are important to study. Probability is an important tool for describing and measuring uncertainty, and it was further introduced into RS theory to construct the probabilistic rough set (PRS). The PRS approach exhibits outstanding merits, including measurability, generality and flexibility, and has many application models, such as the decision-theoretic rough set (DTRS) [46,50], 0.5-probabilistic rough set [44], variable precision rough set (VPRS) [64], rough membership function [30], parameterized rough set [4,31], Bayesian rough set [39,40] and game-theoretic rough set [1,5]. At the same time, the graded rough set (GRS) [21,48,49] also becomes a fundamentally quantitative and extended model by utilizing an absolute measure,

* Corresponding author at: College of Mathematics and Software Science, Sichuan Normal University, Chengdu 610068, PR China.
E-mail addresses: xianyongzh@sina.com.cn (X. Zhang), miaoduoqian@163.com.cn (D. Miao).

http://dx.doi.org/10.1016/j.ins.2014.02.160
0020-0255/

which is compared to the relative measure used in PRS. We conducted a comparative study of VPRS and GRS in [59], and we further investigated the double quantification with both relative and absolute measures in [57,58].

PRS models generally require thresholds to divide the universe into application regions; as a result, threshold determination becomes a critical task. However, most thresholds usually lack underlying semantics. Yao et al. [46,50] proposed DTRS by the Bayesian risk-decision and three-way decision semantics and provided a threshold calculation and an explanation. In particular, DTRS can implement the unification of most PRS models, and it also provides a PRS fundamental exploration platform [51]. Much valuable research on DTRS has been performed. Refs. [46,47] investigated three-way decisions; Refs. [15,23,24,34] discussed model development and threshold calculation; Refs. [8,9,13,51,61] studied DTRS attribute reduction; Refs. [12,18,20,53] applied DTRS to the clustering problem; moreover, in contrast to the classical two-category case, Refs. [18,19,22,62] explored multi-category problems.

Attribute reduction is an essential subject in RS theory and underlies many practical applications. A reduct – the core notion of this theory – is a minimum attribute subset that preserves a certain classification property, as provided by entire attributes; thus, reducts hold great significance for optimization and generalization. Different approaches and algorithms have been extensively proposed in recent studies to determine all reducts or a single reduct [14,16,41,42,52,60]. Miao et al. [26] investigated reducts in consistent and inconsistent decision tables of the Pawlak model, and in particular, they analyzed the hierarchy of three types of reducts. With respect to PRS reduction, Refs. [8,9,13,51,61] studied DTRS reduction, Refs. [2,7,25,43,63] studied VPRS reduction and Refs. [40,54] concerned Bayesian rough set reduction. As is well known, a Pawlak reduct preserves the classification positive region (C-POS) due to the monotonicity of C-POS [28,29]. However, PRS usually exhibits C-POS non-monotonicity due to quantitative extension; thus, directly applying a Pawlak reduct would introduce some anomalies [2,25,43,51]. Therefore, quantitative PRS reduction has already transcended qualitative Pawlak reduction, and the relevant construction becomes a problem. For the DTRS reduction issue, Yao and Zhao [51] analyzed the separability of the dependency degree γ and proposed different measures beyond γ (such as confidence, coverage, generality and cost); furthermore, general reducts were explored by considering multiple measures. Moreover, Jia et al. [8,9] proposed a reduct based on minimum cost; Zhao et al. [61] proposed a reduct based on C-POS preservation; and Li et al. [13] proposed a reduct based on the C-POS measure.

DTRS serves as one type of fundamental PRS, and its attribute reduction can well reflect the nature of quantitative reduction. Thus, this paper mainly concentrates on DTRS reduction. In particular, the two-category case is a fundamental form of DTRS and features relatively clear relationships for both set regions and rule consistency; thus, two-category DTRS underlies generalization exploration and can also provide effective degeneration analysis. In view of its relative difficulty, the use of the two-category approach appears to be a rational strategy for conducting a breakthrough study on quantitative attribute reduction. In particular, it should be noted that rough logic is also an important research direction in RS theory. Decision logic [29] is a fundamental approach, and more extensive logic approaches have been reported in [3,10,17,27,36,37]. In fact, decision logic provides an effective theoretical system for attribute reduction and can simplify reduction algorithms, in which consistency/inconsistency plays a core role. Currently, quantitative attribute reduction extensively concerns region targets but rarely considers logic consistency. Determining how to select a rational reduction criterion is undoubtedly the key problem in attribute reduction. For qualitative Pawlak reduction, region preservation and consistency preservation become natural and equivalent reduction targets. For two-category DTRS reduction, we will investigate the rationality, separability and integration of the two reduction targets, establish the double-preservation principle to tackle the reduction criteria problem and further investigate the double-preservation reduct. As a result, the qualitative Pawlak reduct will be promoted and extended to quantitative attribute reduction.

In multi-granule scenarios, granular computing strongly emphasizes structure and hierarchy, and Refs. [32,33,38,45] have discussed many valuable characteristics and results regarding information granules. In fact, RS theory is usually viewed as a concrete model of granular computing, and the concept and method of granular computing can be used to effectively analyze the attribute reduction issue. Attribute reduction is related to knowledge structures, whereas reduction targets provide knowledge structure requirements. Accordingly, a structural study can fully probe the essence of attribute reduction. Moreover, hierarchical reduction targets can lead to hierarchical reducts, and relevant research can deepen our understanding of reduct structure systems. Herein, the above-mentioned separability of the two basic reduction targets provides ample space for implementing structural and hierarchical approaches. Based on reduction target structures, this paper investigates hierarchical attribute reduction for two-category DTRS as follows: (1) The knowledge-preservation property and reduct are explored, and the Pawlak reduct in the knowledge representation system is promoted. (2) The consistency-preservation principle and reduct are constructed by the consistency mechanism, and the Pawlak reduct in the decision table is promoted. (3) Region preservation is analyzed, and the separability between consistency preservation and region preservation is determined; accordingly, the double-preservation principle and reduct are explored, and the Pawlak reduct is extended. (4) The structure targets are proposed based on knowledge structures, and attribute reduction is further described and simulated; accordingly, general reducts are proposed to preserve structure targets or optimal measures. (5) The hierarchical relationships of the relevant four targets and reducts are analyzed, and a decision table example is provided for illustration. In particular, most of the conclusions drawn and results obtained, including the reduction targets and reduct notions, offer some generalization to multi-category and quantitative models, which is also analyzed.

This paper offers the following five innovations: (1) promotion and utilization of the classical Pawlak reduct; (2) rational construction with respect to consistency preservation, region preservation and double preservation; (3) structural reduction construction based on knowledge structures; (4) hierarchical exploration for attribute reduction; and (5) generalization

analysis based on two-category DTRS reduction. Through progressive and systemic discussions, this study establishes a fundamental reduction framework for two-category DTRS and also provides new insights into the problems of Pawlak reduction, DTRS reduction and PRS reduction.

The remainder of this paper is organized as follows. Section 2 first reviews the Pawlak model, Pawlak reduct, DTRS model and DTRS reduct. Sections 3–5 describe the principles and reducts applied in this study with respect to knowledge preservation, consistency preservation and double preservation. Section 6 explores the structure targets and general reducts. Section 7 presents the reduct relationships and a sample illustration. Section 8 provides concluding remarks.

2. Preliminaries

Some terms will be used repeatedly throughout this paper. For simplification, the main abbreviations are first introduced in this section. Terms that are replaced by their first letter include the following: Decision → D, Knowledge → K, Consistency → C, Region → R and Preservation → P. Accordingly, the following phrases are clear: D-Table, K-Coarsening, K-Preservation, C-Preservation, R-Preservation and CR-Preservation. Moreover, KP-Reduct, CP-Reduct and CRP-Reduct denote the K-Preservation reduct, C-Preservation reduct and CR-Preservation reduct, respectively. ST indicates structure target in terms such as ST-Preservation and STP-Reduct. KRS indicates knowledge representation system. POS, BND and NEG and C-POS, C-BND and C-NEG denote the set positive, boundary negative regions and the classification positive, boundary and negative regions, respectively.

The next section will provide a review of relevant content associated with the basic models and reducts employed in this paper.

2.1. Pawlak model and Pawlak reduct

Herein, the Pawlak model and Pawlak reduct [28,29] are reviewed.

U is a finite and non-empty universe, $\mathscr{R}$ is a family of equivalence relations over U and $(U,\mathscr{R})$ constitutes a knowledge base. Suppose that $\phi \neq R \subseteq \mathscr{R}, \cap R$ is an equivalence relation, which is denoted as $IND(R)$. The classified structure $U/IND(R)$ serves as knowledge and is sometimes directly referred to as knowledge R. $[x]_R$ denotes the relevant equivalence class, i.e., the basic knowledge granule. Suppose that $X \subseteq U$ and the lower and upper approximations of X are defined by

$$\underline{apr}_R X = \{x | [x]_R \subseteq X\}, \overline{apr}_R X = \{x | [x]_R \cap X \neq \phi\}, \tag{1}$$

$$\mathrm{POS}_R(X) = \underline{apr}_R X, \mathrm{NEG}_R(X) = U - \overline{apr}_R X, \mathrm{BND}_R(X) = \overline{apr}_R X - \underline{apr}_R X, \tag{2}$$

denote POS, NEG and BND, respectively. Given that $P, Q \subseteq \mathscr{R}$, if $IND(P) \subseteq IND(Q)$, then knowledge Q *depends* on knowledge P (while knowledge P *deduces* knowledge Q), which is denoted as $P \Rightarrow Q$, and the knowledge structure about $U/IND(Q)$ is rougher than the knowledge structure about $U/IND(P)$. The *dependency/deducibility* underlies rough reasoning.

Knowledge is usually represented as a value-attribute table, called a knowledge representation system (KRS). KRS is a pair (U, At); U is the universe; At is a finite set of primitive attributes; and each $a \in At$ is a total function $a : U \to V_a$, where V_a is the value set of a. Suppose $\phi \neq A \subseteq C$; then, $IND(A) = \{(x,y) \in U \times U | \forall a \in A, a(x) = a(y)\}$ is the equivalence relation, and $U/IND(A)$ is the knowledge. KRS Pawlak reduct aims to preserve $IND(C)$ to preserve the knowledge. The decision table (D-Table) is a special KRS with a distinguished condition and decision attributes, and it plays an important role in decision making. Let (U, At) be a KRS, and let $C, D \subseteq At$ be the condition and decision attribute subsets, respectively; then, $S = (U, C \cup D)$ constitutes the D-Table. The D-Table is consistent if all of the object pairs that have the same condition values on C also have the same decision values on D; otherwise, the D-Table is inconsistent. The positive region of D on A (i.e., C-POS) is defined by

$$\mathrm{POS}_A(D) = \bigcup_{X \in U/IND(D)} \underline{apr}_{IND(A)} X, \tag{3}$$

whereas $\mathrm{BND}_A(D) = U - \mathrm{POS}_A(D)$ is the classification boundary (i.e., C-BND). If $B' \subseteq B \subseteq C$, then $\mathrm{POS}_{B'}(D) \subseteq \mathrm{POS}_B(D)$; thus, C-POS exhibits monotonicity. In particular, $\gamma_A(D) = \frac{|\mathrm{POS}_A(D)|}{|U|}$ is the classical measure γ for the classification quality, i.e., the dependency degree of D on A.

Definition 2.1 [29]. Suppose that $B \subseteq C$. B is a Pawlak reduct of C if it satisfies the C-POS preservation and set independency, i.e., $\mathrm{POS}_B(D) = \mathrm{POS}_C(D)$, and $\forall b \in B, \mathrm{POS}_{B-\{b\}}(D) \neq \mathrm{POS}_B(D)$. $Core(C) = \{a \in C | \mathrm{POS}_{C-\{a\}}(D) \neq \mathrm{POS}_C(D)\}$ is the core of C.

If $|U/IND(D)| = 2$, then D-Table $S = (U, C \cup D)$ concerns the two-category case. Suppose that $U/IND(D) = \{X, \neg X\}$; then, $\mathrm{POS}_A(D) = \mathrm{POS}_A(X) \cup \mathrm{NEG}_A(X), \mathrm{BND}_A(D) = \mathrm{BND}_A(X), \gamma_A(D) = \frac{|\mathrm{POS}_A(X)|}{|U|} + \frac{|\mathrm{NEG}_A(X)|}{|U|}$. Thus, the set regions and classification regions exhibit clear relationships. Moreover, C-POS preservation is equivalent to region preservation.

Definition 2.2 [29]. Let (C, D) be an algorithm, $a \in C, B \subseteq C$. (1) Suppose that algorithm (C, D) is consistent. If $((C - \{a\}), D)$ is inconsistent, then a is indispensable in (C, D), and the set of all indispensable attributes in (C, D) forms the core of (C, D). B is a reduct of C if (B, D) is consistent, and $\forall b \in B, b$ is independent in (B, D). (2) Suppose that algorithm (C, D) is inconsistent. If

$\mathrm{POS}(C,D) \neq \mathrm{POS}((C-\{a\}),D)$, then a is indispensable in (C,D), and the set of all indispensable attributes in (C,D) forms the core of (C,D). B is a reduct of C if $\mathrm{POS}(B,D) = \mathrm{POS}(C,D)$, and $\forall b \in B, b$ is independent in (B,D).

By Definition 2.2, the decision logic provides the reduct concepts of the consistent/inconsistent algorithm related to the consistent/inconsistent D-Table. Herein, $\mathrm{POS}(C,D)$ is the positive region of the algorithm and reflects the set of all consistent rules in the algorithm. The reduction targets in the decision logic aim to preserve rule consistency in both consistent and inconsistent D-Tables.

2.2. DTRS model and DTRS reduct

Next, the DTRS model is illustrated by the two-category problem [47,50].

There are only two states and three actions (accept, defer and reject). The state set $\Omega = \{X, \neg X\}$ indicates that an element is in X and not in X, and the action set is $\mathscr{A} = \{a_P, a_B, a_N\}$, where a_P, a_B and a_N represent the three actions of deciding that an object is in the sets $\mathrm{POS}(X), \mathrm{BND}(X)$ and $\mathrm{NEG}(X)$, respectively. Moreover, when an object belongs to X, let $\lambda_{PP}, \lambda_{BP}$ and λ_{NP} denote the costs of taking the actions a_P, a_B and a_N, respectively; when an object does not belong to X, then let $\lambda_{PN}, \lambda_{BN}$ and λ_{NN} denote the costs of taking the same three actions, respectively. The loss functions regarding the states X and $\neg X$ can be expressed as a 2×3 matrix, as follows:

	a_P	a_B	a_N
X	λ_{PP}	λ_{BP}	λ_{NP}
$\neg X$	λ_{PN}	λ_{BN}	λ_{NN}

Here, $[x]$ denotes the description of x, and the probabilities for the two complement states are denoted as $P(X|[x]) = \frac{|X \cap [x]|}{|[x]|}$ and $P(\neg X|[x]) = 1 - P(X|[x])$.

The expected costs $\mathscr{R}(a_i|[x])$ of taking individual actions can be expressed as

$$\begin{aligned} \mathscr{R}(a_P|[x]) &= \lambda_{PP}P(X|[x]) + \lambda_{PN}P(\neg X|[x]), \\ \mathscr{R}(a_B|[x]) &= \lambda_{BP}P(X|[x]) + \lambda_{BN}P(\neg X|[x]), \\ \mathscr{R}(a_N|[x]) &= \lambda_{NP}P(X|[x]) + \lambda_{NN}P(\neg X|[x]). \end{aligned} \tag{4}$$

The Bayesian decision procedure leads to the following minimum-risk decision rules:

(P) If $\mathscr{R}(a_P|[x]) \leqslant \mathscr{R}(a_B|[x])$ and $\mathscr{R}(a_P|[x]) \leqslant \mathscr{R}(a_N|[x])$, then decide $[x] \subseteq \mathrm{POS}(X)$;
(B) If $\mathscr{R}(a_B|[x]) \leqslant \mathscr{R}(a_P|[x])$ and $\mathscr{R}(a_B|[x]) \leqslant \mathscr{R}(a_N|[x])$, then decide $[x] \subseteq \mathrm{BND}(X)$;
(N) If $\mathscr{R}(a_N|[x]) \leqslant \mathscr{R}(a_P|[x])$ and $\mathscr{R}(a_N|[x]) \leqslant \mathscr{R}(a_B|[x])$, then decide $[x] \subseteq \mathrm{NEG}(X)$.

By the reasonable loss condition, $\lambda_{PP} \leqslant \lambda_{BP} < \lambda_{NP}, \lambda_{NN} \leqslant \lambda_{BN} < \lambda_{PN}$, the minimum-risk decision rules (P)-(B) can be written as

(P1) If $P(X|[x]) \geqslant \alpha$ and $P(X|[x]) \geqslant \gamma$, then decide $[x] \subseteq \mathrm{POS}(X)$;
(B1) If $P(X|[x]) \leqslant \alpha$ and $P(X|[x]) \geqslant \beta$, then decide $[x] \subseteq \mathrm{BND}(X)$;
(N1) If $P(X|[x]) \leqslant \beta$ and $P(X|[x]) \leqslant \gamma$, then decide $[x] \subseteq \mathrm{NEG}(X)$,

where

$$\begin{aligned} \alpha &= \frac{\lambda_{PN} - \lambda_{BN}}{(\lambda_{PN} - \lambda_{BN}) + (\lambda_{BP} - \lambda_{PP})} = \frac{\lambda_{(P-B)N}}{\lambda_{(P-B)N} + \lambda_{(B-P)P}}, \\ \beta &= \frac{\lambda_{BN} - \lambda_{NN}}{(\lambda_{BN} - \lambda_{NN}) + (\lambda_{NP} - \lambda_{BP})} = \frac{\lambda_{(B-N)N}}{\lambda_{(B-N)N} + \lambda_{(N-B)P}}, \\ \gamma &= \frac{\lambda_{PN} - \lambda_{NN}}{(\lambda_{PN} - \lambda_{NN}) + (\lambda_{NP} - \lambda_{PP})} = \frac{\lambda_{(P-N)N}}{\lambda_{(P-N)N} + \lambda_{(N-P)P}}. \end{aligned} \tag{5}$$

Rule (B1) indicates that $\alpha > \beta$, and furthermore, $0 \leqslant \beta < \gamma < \alpha \leq 1$. Thus, the following three-way decisions are obtained:

(P2) If $P(X|[x]) \geqslant \alpha$, then decide $[x] \subseteq \mathrm{POS}(X)$;
(B2) If $\beta < P(X|[x]) < \alpha$, then decide $[x] \subseteq \mathrm{BND}(X)$;
(N2) If $P(X|[x]) \leqslant \beta$, then decide $[x] \subseteq \mathrm{NEG}(X)$.

Hence, the thresholds α, β are determined by the loss functions $\lambda_{\cdot\cdot}$ in formula (5), and the three regions are established:

$$\mathrm{POS}^{\alpha,\beta}(X) = \{x|P(X|[x]) \geqslant \alpha\},$$
$$\mathrm{BND}^{\alpha,\beta}(X) = \{x|\beta < P(X|[x]) < \alpha\},$$
$$\mathrm{NEG}^{\alpha,\beta}(X) = \{x|P(X|[x]) \leqslant \beta\}.$$

The two-category case is a basic item for the DTRS model in which only the three-way regions (POS, NEG and BND) can fully describe the region structures. The two-category case can be generalized to the multi-category case and can also be used to conduct relevant degeneration detection. Note that there are two main approaches for the multi-category extension. One approach transforms the multi-category case into multiple two-category cases, whereas the other resorts to performing a Bayesian decision in a high-dimensional space. C-POS is a core concept for multi-category attribute reduction and has two definitions that are related to the two approaches. Here, $S = (U, C \cup D), A \subseteq C, |U/IND(D)| \geqslant 2$.

Definition 2.3 (*Form I*). [13,51]

$$\begin{aligned}
\mathrm{POS}_A^{\alpha,\beta}(D) &= \bigcup_{X\in U/IND(D)} \mathrm{POS}_A^{\alpha,\beta}(X),\\
\mathrm{BND}_A^{\alpha,\beta}(D) &= \bigcup_{X\in U/IND(D)} \mathrm{BND}_A^{\alpha,\beta}(X),\\
\mathrm{NEG}_A^{\alpha,\beta}(D) &= U - \mathrm{POS}_A^{\alpha,\beta}(D) \cup \mathrm{BND}_A^{\alpha,\beta}(D).
\end{aligned} \tag{6}$$

Definition 2.4 (*Form II*). [8,61]

$$\begin{aligned}
\mathrm{POS}_A^{\alpha,\beta}(D) &= \{x|P(D_{max}([x]_A)|[x]_A) \geqslant \alpha\},\\
\mathrm{BND}_A^{\alpha,\beta}(D) &= \{x|\beta < P(D_{max}([x]_A)|[x]_A) < \alpha\},\\
\mathrm{NEG}_A^{\alpha,\beta}(D) &= \{x|P(D_{max}([x]_A)|[x]_A) \leqslant \beta\},
\end{aligned} \tag{7}$$

where $D_{max}([x]_A) = arg\ \max_{X\in U/IND(D)}\{\frac{|[x]_A\cap X|}{|[x]_A|}\}$.

We next conduct degeneration for Forms I and II by the two-category approach.

Proposition 2.5. *For Forms I and II, the corresponding results in the two-category approach are as follows. First,* $\mathrm{NEG}_A^{\alpha,\beta}(D) = \phi$.

(1) *If* $\alpha + \beta = 1$, *then* $\mathrm{POS}_A^{\alpha,\beta}(D) = \mathrm{POS}_A^{\alpha,\beta}(X) \cup \mathrm{NEG}_A^{\alpha,\beta}(X), \mathrm{BND}_A^{\alpha,\beta}(D) = \mathrm{BND}_A^{\alpha,\beta}(X), \mathrm{POS}_A^{\alpha,\beta}(D) \cap \mathrm{BND}_A^{\alpha,\beta}(D) = \phi$,
(2) *If* $\alpha + \beta > 1$, *then* $\mathrm{POS}_A^{\alpha,\beta}(D) \subset \mathrm{POS}_A^{\alpha,\beta}(X) \cup \mathrm{NEG}_A^{\alpha,\beta}(X), \mathrm{BND}_A^{\alpha,\beta}(D) \supset \mathrm{BND}_A^{\alpha,\beta}(X), \mathrm{POS}_A^{\alpha,\beta}(D) \cap \mathrm{BND}_A^{\alpha,\beta}(D) = \phi$,
(3) *Suppose that* $\alpha + \beta < 1$. *For Form I,* $\mathrm{POS}_A^{\alpha,\beta}(D) \supset \mathrm{POS}_A^{\alpha,\beta}(X) \cup \mathrm{NEG}_A^{\alpha,\beta}(X), \mathrm{BND}_A^{\alpha,\beta}(D) \supset \mathrm{BND}_A^{\alpha,\beta}(X), \mathrm{POS}_A^{\alpha,\beta}(D) \cap \mathrm{BND}_A^{\alpha,\beta}(D) \neq \phi$. *For Form II,* $\mathrm{POS}_A^{\alpha,\beta}(D) \supset \mathrm{POS}_A^{\alpha,\beta}(X) \cup \mathrm{NEG}_A^{\alpha,\beta}(X), \mathrm{BND}_A^{\alpha,\beta}(D) \subset \mathrm{BND}_A^{\alpha,\beta}(X), \mathrm{POS}_A^{\alpha,\beta}(D) \cap \mathrm{BND}_A^{\alpha,\beta}(D) = \phi$.

Both forms of the classification regions aim to generalize the two-category results and to multi-category attribute reduction. By degeneration, Proposition 2.5 provides the two-category result on the set regions. C-NEG can be non-empty in the multi-category DTRS model, but compared to the Pawlak model result, C-NEG is still empty in the two-category case. Moreover, form I exhibits an unconventional property in condition $\alpha + \beta < 1$.

Currently, DTRS reduction mainly focuses on multi-category cases. Jia et al. [8] provided a thorough review of the existing methods that can be used to this end, including positive-based reduct [61], nonnegative-based reduct [8] and cost-based reduct [8]. Here, the three reducts are first defined (where only Form II is concerned) and will be subsequently compared with our approach by using reduction targets and degeneration.

Definition 2.6 [61]. B is a positive-based reduct of C, if (1) $\mathrm{POS}_B^{\alpha,\beta}(D) = \mathrm{POS}_C^{\alpha,\beta}(D)$; (2) $\forall b \in B, \mathrm{POS}_{B-\{b\}}^{\alpha,\beta}(D) \neq \mathrm{POS}_B^{\alpha,\beta}(D)$.

Definition 2.7 [8]. B is a nonnegative-based reduct of C, if (1) $\neg\mathrm{NEG}_B^{\alpha,\beta}(D) = \neg\mathrm{NEG}_C^{\alpha,\beta}(D)$; (2) $\forall b \in B, \neg\mathrm{NEG}_{B-\{b\}}^{\alpha,\beta}(D) \neq \neg\mathrm{NEG}_B^{\alpha,\beta}(D)$. Here, $\neg\mathrm{NEG}^{\alpha,\beta}$ indicates the union of C-POS and C-BND.

Proposition 2.8 [8]. *Let* $\lambda_{PP} = 0 = \lambda_{NN}, p = P(D_{max}([x]_A)|[x]_A)$; *then, the cost to the knowledge A is*

$$COST(A) = \sum_{x\in \mathrm{POS}_A^{\alpha,\beta}(D)} (1-p)\lambda_{PN} + \sum_{x\in \mathrm{BND}_A^{\alpha,\beta}(D)} (p\lambda_{BP} + (1-p)\lambda_{BN}) + \sum_{x\in \mathrm{NEG}_A^{\alpha,\beta}(D)} p\lambda_{NP}.$$

Definition 2.9 [12]. B is a cost-based reduct of C, if (1) $B = \arg\min_{A\subseteq C} COST(A)$; (2) $\forall B' \subset B, COST(B') > COST(B)$.

3. Knowledge preservation (K-Preservation) and KP-Reduct

Attribute reduction is related to a change in knowledge structure. Based on the novel idea that decision attributes provide only structure criteria, the independent study of the knowledge structures of conduction attributes provides a fundamental approach to D-Table reduction. Thus, we study both the K-Preservation target and the reduct in this section.

Suppose that D-Table $S = (U, C \cup D), C' \subseteq C, a \in C$. When deleting attribute set C', C and $C - C'$ correspond to old and new knowledge, respectively. This process is specifically referred to as knowledge coarsening (K-Coarsening) in this paper and is noted as $C \xrightarrow{-C'} C - C'$ or $C \xrightarrow{-} C - C'$; if $C' = \{a\}$, then only $C \xrightarrow{-a} C - \{a\}$ is used. In fact, attribute reduction concerns mainly K-Coarsening, and the latter is directly related to granular computing. Next, Theorem 3.1 and Proposition 3.2 present common properties of K-Coarsening.

Theorem 3.1 (*Knowledge monotonicity*).
(1) $IND(C - C') \supseteq IND(C)$. *i.e.,* $C \Rightarrow C - C'$; *(2)* $|U/IND(C - C')| \leqslant |U/IND(C)|$.

Proposition 3.2.

(1) $IND(C - C') = IND(C) \Longleftrightarrow |U/IND(C - C')| = |U/IND(C)|$;
(2) $IND(C - C') \supset IND(C) \Longleftrightarrow |U/IND(C - C')| < |U/IND(C)|$.

Theorem 3.1 reflects the most basic feature of K-Coarsening – knowledge monotonicity, i.e., knowledge does not become finer and the knowledge-granular number does not increase. Proposition 3.2 further shows that the K-Coarsening feature can be represented only by the knowledge-granular number.

K-Coarsening involves two cases. First, the knowledge structure may not change when reducing the attributes, where the deleted attributes might be redundant. Second, attribute deletion leads to some changes in the knowledge structure, i.e., old but finer knowledge becomes new but rougher knowledge. Proposition 3.2 actually reflects both cases and provides an identification feature.

For both cases, knowledge preservation and knowledge non-preservation are important properties. For simplicity, these concepts are denoted as KP and KNP, respectively. Suppose that K-Coarsening $\varphi : . \xrightarrow{-} .;$ then, "$. \xrightarrow{-} . \models KP$" indicates that φ preserves knowledge, whereas "$. \xrightarrow{-} . \models KNP$" shows the opposite. Next, KP and KNP stability will be discussed. For this purpose, the K-Coarsening sequence is first studied. Here, suppose that $C = B \cup C', C' = \{a_1, a_2, \ldots, a_t\}, t \geqslant 2$.

Lemma 3.3.

$$C \xrightarrow{-C'} B \Longleftrightarrow C \xrightarrow{-a_1} B - \{a_1\} \xrightarrow{-a_2} \ldots \xrightarrow{-a_t} B - \{a_1, a_2, \ldots, a_t\}, \forall a_1, a_2, \ldots, a_t. \tag{8}$$

Lemma 3.3 reflects the decomposition/construction property and interchangeability for a K-Coarsening sequence, which are clear when considering the attribute deletion action. Moreover, they also hold for a sequence in which the deleted part is not a single attribute but instead is an attribute subset. Next, suppose that $C_0 \subseteq C'$, and $. \xrightarrow{-C_0} .$ is K-Coarsening in a sequence T for $C \xrightarrow{-C'} B$.

Theorem 3.4 (*Infection Principle*).

$$\begin{aligned} &\exists T, \exists . \xrightarrow{-C_0} . \models KNP \Longleftrightarrow C \xrightarrow{-C'} B \models KNP, \\ &\exists C_0^* \subseteq C', C \xrightarrow{-C_0^*} C - C_0^* \models KNP \Longleftrightarrow C \xrightarrow{-C'} C - C' \models KNP. \end{aligned} \tag{9}$$

Proof. (1) Based on Lemma 3.3, $. \xrightarrow{-C_0} .$ can be assumed to be the first K-Coarsening with KNP in the sequence, i.e., $IND(C - C_0) \supset IND(C)$. Based on knowledge monotonicity (Theorem 3.1), the same knowledge on $U/IND(C)$ cannot be obtained in the later K-Coarsening. Hence, $C \xrightarrow{-C'} B \models KNP$. The opposite is made clear by $C_0 = C'$ and $T = \{C \xrightarrow{-C'} B\}$. (2) $C \xrightarrow{-C'} B \Longleftrightarrow C \xrightarrow{-C_0^*} C - C_0^* \xrightarrow{-} B$. As $C \xrightarrow{-C_0^*} C - C_0^* \models KNP, C \xrightarrow{-C'} B \models KNP$ according to (1). The opposite is made clear by $C_0^* = C'$. □

Theorem 3.5 (*Purity Principle*).

$$\begin{aligned} &C \xrightarrow{-C'} B \models KP \Longleftrightarrow \forall T, \forall . \xrightarrow{-C_0} . \models KP, \\ &C \xrightarrow{-C'} C - C' \models KP \Longleftrightarrow \forall C_0^* \subseteq C', C \xrightarrow{-C_0^*} C - C_0^* \models KP. \end{aligned} \tag{10}$$

The Infection Principle shows that any sub-deletion (in any stages) with the KNP feature will lead to the KNP result for the whole deletion. In other words, KNP exhibits the infection feature from the internal part of the K-Coarsening sequence. In contrast, the Purity Principle demonstrates the KP harmony between the global and local K-Coarsening. Moreover, the Infection and Purity Principles exhibit duality.

Definition 3.6.

$$\begin{aligned} &KP(C) = \{a \in C | IND(C - \{a\}) = IND(C)\}, \\ &KNP(C) = \{a \in C | IND(C - \{a\}) \supset IND(C)\} \end{aligned} \tag{11}$$

are called the K-Preservation and KN-Preservation sets of C, respectively.

According to the single attribute deletion, Definition 3.6 describes the two cases with respect to C in K-Coarsening and proposes two classified sets for attributes, i.e., $KP(C)$ and $KNP(C)$.

Corollary 3.7.

$$\begin{aligned} &C \xrightarrow{-a} C - \{a\} \models KP \Longleftrightarrow a \in KP(C); \\ &C \xrightarrow{-a} C - \{a\} \models KNP \Longleftrightarrow a \in KNP(C). \end{aligned} \tag{12}$$

Corollary 3.7 exhibits the fundamental feature of $KP(C)$, that knowledge C does not change when deleting each attribute, whereas $KNP(C)$ reflects the opposite. Furthermore, the Core Principle will illustrate the deletion property on $KP(C)$ and $KNP(C)$.

Theorem 3.8 (*Core Principle*).

$$\begin{aligned} &(1)\ KNP(C) \cap C' \neq \phi \Rightarrow C \xrightarrow{-C'} B \models KNP; \\ &(2)\ C \xrightarrow{-C'} B \models KP \Rightarrow C' \subseteq KP(C). \end{aligned} \tag{13}$$

Proof. (1) $KNP(C) \cap C' \neq \phi$ indicates that $\exists a_i \in C', C \xrightarrow{-a_i} C - \{a_i\} \models KNP$; based on the Infection Principle, $C \xrightarrow{-C'} B \models KNP$. Moreover, (1) and (2) are dual, and thus, the proof is completed. □

Definition 3.9. $Core(C) = KNP(C)$ is the core on K-Preservation.

The attribute set C has been divided into two classes (i.e., $KP(C)$ and $KNP(C)$) according to the KP feature of the single attribute deletion. The Core Principle (Theorem 3.8) further shows that KP cannot delete attributes in $KNP(C)$ but might delete some attributes in $KP(C)$. Therefore, $KNP(C)$ becomes the K-Preservation core. The opposite of the Core Principle does not hold because both $KP(C)$ and $KNP(C)$ require the specific condition with respect to only one attribute. Example 1 provides a counterexample. Moreover, the Core Principle and its opposite become clear according to the KRS core and reduct, and Theorem 3.12 will establish the basis. For K-Preservation simplification, attribute deletion in $KP(C)$ therefore exhibits both necessity and non-sufficiency.

Example 1. In $S = (U, C \cup D), U = \{x_1, \ldots, x_8\}, C = \{a, b, c\}$.

$$\begin{aligned} &U/IND(C) = \{\{x_1, x_5\}, \{x_2, x_8\}, \{x_3\}, \{x_4\}, \{x_6\}, \{x_7\}\}, \\ &U/IND(\{a\}) = \{\{x_1, x_4, x_5\}, \{x_2, x_8\}, \{x_3\}, \{x_6, x_7\}\}; \\ &U/IND(\{b\}) = \{\{x_1, x_3, x_5\}, \{x_6\}, \{x_2, x_4, x_7, x_8\}\}; \\ &U/IND(\{c\}) = \{\{x_1, x_5\}, \{x_6\}, \{x_2, x_7, x_8\}, \{x_3, x_4\}\}. \end{aligned} \tag{14}$$

Thus, $KNP(C) = \{a\}, KP(C) = \{b, c\}$; however, $C \xrightarrow{-KP(C)} \{a\} \models KNP$.

For D-Table, the Infection and Purity Principles play a fundamental role in the independent environment with respect to knowledge C. The principles ensure the K-Preservation's stability and provide the deletion core. The Infection and Purity Principles and the Core Principle harmoniously underlie K-Preservation reduction. Moreover, they all depend on only the D-Table's formal structure (mainly on C) rather than on concrete models. Thus, a type of reduct on K-Preservation can be defined for all models, which is also related only to the formal structure.

Definition 3.10 (*KP-Reduct*). B is a KP-Reduct of C, if (1) $IND(B) = IND(C)$; (2) $\forall B' \subset B, IND(B') \supset IND(B)$.

Corollary 3.11. *If B is a KP-Reduct of C, then* $C \xrightarrow{-(C-B)} B \models KP$.

Note that a KP-Reduct has a unified definition relative to the KRS Pawlak reduct [29], but its descriptive framework is D-Table $S = (U, C \cup D)$. The reduction target of KP-Reduct, i.e., K-Preservation, is the highest requirement for the knowledge structure; thus, it should be naturally suitable for all model reduction. Corollary 3.11 indicates that deleting all of the attributes beyond KP-Reduct do not cause any change in knowledge. Thus, some redundant attributes can be first deleted by KP-Reduct if our purpose is to seek a weaker reduct. Based on the core $KNP(C)$, KP-Reduct provides an approach to searching for

concrete deletion-attribute sets in $KP(C)$. To a certain extent, therefore, K-Preservation reduction can be viewed as a pretreatment for D-Table reduction and all model reduction. Based on KP-Reduct, Section 7 will provide valuable information for other reducts.

Theorem 3.12 (*KP-Reduct Promotion*). *Based on the preservation of the knowledge of C, KP-Reduct on D-Table* $S = (U, C \cup D)$ *promotes the Pawlak reduct on KRS* (U, C) *from an applicability perspective.*

Theorem 3.12 reflects the applicability promotion of KP-Reduct based on the KRS Pawlak reduct. (1) The latter applies only to the Pawlak model, whereas the former may apply to arbitrary models, although both models are equivalent at the Pawlak model level. (2) The latter applies only to KRS (U, C), whereas the former could apply to D-Table $S = (U, C \cup D)$, with the arbitrary decision set D; as a result, we establish the reduction connection between D-Table and its sub-KRS. From another perspective, the KRS Pawlak reduct has been promoted to describe D-Table and becomes a type of general D-Table reduct, i.e., the proposed KP-Reduct. Therefore, K-Preservation and KP-Reduct exhibit a generalization feature and have applicable significance for all model reduction; of course, they should utilize the results of the KRS Pawlak reduct. Moreover, they correspond to the dependency supremum and thus provide the most natural but the strongest structure. Against this background, the structure targets and general reducts in Section 6 will provide a perfect structural explanation.

4. Consistency preservation (C-Preservation) and CP-Reduct

Consistency/inconsistency is a fundamental feature of decision rules. What the Pawlak reduct aims to preserve in decision logic is rule consistency in both consistent and inconsistent D-Tables. In this section, we explore rule consistency/inconsistency within the two-category DTRS framework; furthermore, we construct the corresponding reduction principle and concept. Thus, a fundamental principle (C-Preservation) will be rationally established and extensively generalized. Next, consistency/inconsistency will be directly described while the rules are omitted.

Definition 4.1. Given $S = (U, C \cup D), U/IND(D) = \{X, \neg X\}, \forall A \subseteq C.\ cst_A, icst_A : 2^U \to 2^U, \forall E \subseteq U,$

$$\begin{aligned} cst_A(E) &= \{x \in E | \forall [x]_A, ([x]_A \subseteq X) \vee ([x]_A \subseteq \neg X)\}, \\ icst_A(E) &= \{x \in E | \forall [x]_A, ([x]_A \cap X \neq \phi) \wedge ([x]_A \cap \neg X \neq \phi)\}. \end{aligned} \tag{15}$$

$cst_A(E)$ and $icst_A(E)$ are called consistent and inconsistent regions of E on A, respectively.

The consistent and inconsistent regions aim to extract objects that are connected with consistent and inconsistent rules, respectively.

Theorem 4.2 (*Consistency Stability*).
$S = (U, C \cup D), cst_C(U) = U - BND_C(X)$, *and* $icst_C(U) = BND_C(X)$.

The consistency stability demonstrates that both the consistent and inconsistent features of D-Tables are related to only Pawlak-BND. Thus, consistency/inconsistency is completely determined by D-Table (mainly by the formal structure on both C and D) and has no relationship with concrete models or set regions. In other words, the Pawlak-Region structure has been promoted and is related not only to the Pawlak-Region but also to consistency/inconsistency and thus underlies consistency/inconsistency for all of the models. Put simply, Pawlak-BND has been promoted to describe consistency/inconsistency.

Theorem 4.3. *D-Table S is consistent if and only if* $BND_C(X) = \phi$; *S is inconsistent if and only if* $BND_C(X) \neq \phi$.

Based on the consistency stability, Theorem 4.3 extracts the features of consistent and inconsistent D-Tables by Pawlak-BND. Thus, the consistency/inconsistency of D-Table is determined only by Pawlak-BND.

Next, we explore the applicability of DTRS to the consistent/inconsistent D-Table.

Theorem 4.4. *If three-way decisions exist and do not degenerate, then S is inconsistent. If S is consistent, then the deferred decision does not exist, and thus, only the two-way decisions or the single-way decision exists.*

Proof. $\mathrm{BND}_C^{\alpha,\beta}(X) \subseteq \mathrm{BND}_C(X)$. Based on Theorem 4.3, if three-way decisions exist, then $\mathrm{BND}_C^{\alpha,\beta}(X) \neq \phi$; thus, $\mathrm{BND}_C(X) \neq \phi$ and S is inconsistent; if S is consistent, then $\mathrm{BND}_C^{\alpha,\beta}(X) = \phi$, and thus, the deferred decision does not exist. □

Theorem 4.4 reflects the relationships between the decision number and the consistent/inconsistent D-Table. Thus, the three-way decisions are mainly related to the inconsistent D-Table, i.e., DTRS is more suitable for the inconsistent D-Table. In fact, the inconsistent D-Table can better reflect the DTRS thresholds and extension.

Proposition 4.5. $cst_C(E) = E - \mathrm{BND}_C(X), icst_C(E) = E \cap \mathrm{BND}_C(X)$.

Corollary 4.6.

$$\begin{aligned} &cst_C(\mathrm{POS}_C^{\alpha,\beta}(X)) = \mathrm{POS}_C(X), icst_C(\mathrm{POS}_C^{\alpha,\beta}(X)) = \mathrm{POS}_C^{\alpha,\beta}(X) \cap \mathrm{BND}_C(X);\\ &cst_C(\mathrm{BND}_C^{\alpha,\beta}(X)) = \phi, icst_C(\mathrm{BND}_C^{\alpha,\beta}(X)) = \mathrm{BND}_C^{\alpha,\beta}(X);\\ &cst_C(\mathrm{NEG}_C^{\alpha,\beta}(X)) = \mathrm{NEG}_C(X), icst_C(\mathrm{NEG}_C^{\alpha,\beta}(X)) = \mathrm{NEG}_C^{\alpha,\beta}(X) \cap \mathrm{BND}_C(X). \end{aligned} \tag{16}$$

Proposition 4.5 provides the computational formulas for consistent and inconsistent regions. Corollary 4.6 further describes the consistency and inconsistency regions with respect to POS, BND and NEG, where the following formulas are specially utilized: $\mathrm{POS}_C^{\alpha,\beta}(X) \supseteq \mathrm{POS}_C(X), \mathrm{BND}_C^{\alpha,\beta}(X) \subseteq \mathrm{BND}_C(X), \mathrm{NEG}_C^{\alpha,\beta}(X) \supseteq \mathrm{NEG}_C(X)$.

Theorem 4.7 (*Consistency Monotonicity*). *In K-Coarsening, the change in consistency is monotonic, i.e., the consistency region is not enlarged, whereas the inconsistent region is not lessened.*

Pawlak reduction aims to preserve the consistency in decision logic. For extended and quantitative models (such as PRS models), the consistency – an essential factor for attribute reduction – has usually been forgotten and neglected. We start by using this key notion in this paper. Consistency monotonicity (Theorem 4.7) is clear because of Pawlak-BND, and more importantly, it reveals the fact that consistency inherits the monotonicity exhibited by K-Coarsening. Therefore, the C-Preservation Principle is a natural but important requirement. Moreover, consistency monotonicity always exists regardless of the model considered; thus, C-Preservation is a general principle for D-Table reduction.

Principle 4.8 (*C-Preservation Principle*). Attribute reduction should consider C-Preservation; the consistent and inconsistent regions do not change, i.e., the consistency and inconsistency of relevant rules do not change.

The C-Preservation Principle is clearly based on the consistency mechanism and the monotonicity. Preserving consistency is equivalent to preserving inconsistency; thus, C-Preservation indicates dual preservation. In contrast, if consistency is not preserved, then rule consistency or inconsistency necessarily changes. Herein, an example is first presented to provide detailed explanations.

Example 2. Table 1 provides a D-Table with a statistical form, where $S = (U, C \cup D), U = \{x_1, x_2, \ldots, x_{29}\}, C = \{c_1, c_2, c_3\}$ and $U/IND(D) = \{X, \neg X\}$. The original Pawlak-BND, Pawlak-NEG and Pawlak-POS are $[x]_3 \cup [x]_4 \cup [x]_6 \cup [x]_7, [x]_1 \cup [x]_2, \phi$, respectively. First, we can easily verify that the inconsistent rules are truly related to only Pawlak-BND. Next, (1) if c_2 is deleted, then Pawlak-BND is U, and thus, only the changed $[x]_1 \cup [x]_2$ requires analysis. In K-Coarsening $\{c_1, c_2, c_3\} \overset{-c_2}{\rightarrow} \{c_1, c_3\}$, $[x]_1 \cup [x]_2$ is transferred from the old Pawlak-NEG to the new Pawlak-BND; the old rule $(c_1 = 1) \wedge (c_2 = 1) \wedge (c_3 = 1) \Rightarrow \neg X$ is consistent, but the new rule $(c_1 = 1) \wedge (c_3 = 1) \Rightarrow \neg X$ becomes inconsistent due to the existing rule $(c_1 = 1) \wedge (c_3 = 1) \Rightarrow X$ that is obtained by $[x]_3$ or $[x]_4$. (2) If c_3 is deleted, i.e., for $\{c_1, c_2, c_3\} \overset{-c_3}{\rightarrow} \{c_1, c_2\}$, Pawlak-BND does not change while $[x]_1$ and $[x]_2$ are still in Pawlak-NEG; the sole consistent rule $(c_1 = 1) \wedge (c_2 = 1) \wedge (c_3 = 1) \Rightarrow \neg X$ changes into $(c_1 = 1) \wedge (c_2 = 1) \Rightarrow \neg X$, which is still consistent. Therefore, deleting c_2 is prohibited, whereas deleting c_3 is allowed for the C-Preservation Principle; moreover, the consistency monotonicity is also verified.

Next, we further analyze the scientific nature of the C-Preservation Principle. C-Preservation maintains consistency and, in fact, maintains absolute and complete certainty. In other words, the position of the real certainty cannot change even to a slight degree, i.e., the absolute certainty cannot be fused with approximate certainty because both are completely different notions from a qualitative standpoint. On the other hand, inconsistency also requires a qualitative preservation. In Example 2, $[x]_1$ and $[x]_2$ have already been determined to completely belong to $\neg X$, which exhibits real certainty, or so C-Preservation suggests: they should stay in their group with absolute certainty, whereas they cannot be combined with other knowledge granules (such as $[x]_3$ and $[x]_4$) from the other group with uncertainty. This principle is extremely similar to that of social classes and activities. C-Preservation ensures certainty purity and provides the qualitative requirement; other impure certainties will make the hierarchical division and mutual combination with the quantitative criterion, which is partially addressed in the next section's discussion on quantitative regions. Therefore, the C-Preservation Principle is related to the certainty/uncertainty preservation and becomes rational and effective; furthermore, it should be and can become a generalized criterion, whereas the Pawlak-Regions or Pawlak-BND plays the core role.

Table 1
A statistical decision table.

$[x]_i$	$\lvert[x]_i\rvert$	c_1	c_2	c_3	$\lvert[x]_i \cap X\rvert$
$[x]_1$	5	1	1	1	0
$[x]_2$	3	1	1	1	0
$[x]_3$	6	1	2	1	1
$[x]_4$	5	1	2	1	1
$[x]_6$	4	2	3	1	3
$[x]_7$	6	2	3	3	5

Theorem 4.9 (*Equivalence Principle*). *In attribute reduction, C-Preservation is equivalent to Pawlak-BND preservation (or Pawlak-Region preservation).*

Theorem 4.9 indicates that C-Preservation means Pawlak-Region preservation. For the Pawlak reduct, this result is clear according to the two approaches on regions and consistency. For quantitative attribute reduction, the Equivalence Principle becomes a novel content, which means that qualitative Pawlak-Region preservation is also necessary according to the definition of C-Preservation. Thus, Pawlak-Region preservation has already been extensively promoted and generalized due to the universality of C-Preservation. In other words, Pawlak-Region preservation becomes a comprehensive requirement due to the certainty/uncertainty qualitative classification.

Thus, the Pawlak reduct has been naturally promoted for other model reductions. In another view, C-Preservation constructs another criterion, and a type of new reduct (which corresponds to the Pawlak reduct) can be established for D-Table or other models. This concept is similar to that of K-Preservation or KP-Reduct.

Definition 4.10 (*CP-Reduct*). B is the CP-Reduct of C if

(1) $cst_B(U) = cst_C(U)$, i.e., $\mathrm{BND}_B(X) = \mathrm{BND}_C(X)$;
(2) $\forall B' \subset B, cst_{B'}(U) \neq cst_B(U)$, i.e., $\mathrm{BND}_{B'}(X) \supset \mathrm{BND}_B(X)$.

Theorem 4.11 (*CP-Reduct Promotion*). *Under the significance of consistency/inconsistency preservation, CP-Reduct promotes the D-Table Pawlak reduct in terms of applicability.*

To preserve consistency, Definition 4.10 proposes CP-Reduct, and its applicable range is associated with D-Table and arbitrary models. Moreover, the core can be normally defined. Theorem 4.11 is clear and demonstrates that CP-Reduct promotes the D-Table Pawlak reduct for all model reduction, whereas C-Preservation becomes a fundamental and necessary requirement for D-Table reduction. From another perspective, C-Preservation and CP-Reduct are related to the D-Table Pawlak reduct; thus, the relevant Pawlak reduct theory underlies and promotes the CP-Reduct system.

Based on the above-discussed two-category results, both the C-Preservation Principle and CP-Reduct can be generalized to attribute reduction theory. In particular, C-Preservation indicates preserving the Pawlak model's C-BND for multiple categories, which corresponds to classical C-POS preservation in the D-Table Pawlak reduct. Thus far, we have discussed two types of reduction commonality, and they have fully utilized the classical Pawlak reduction theory [29]. Therefore, this development approach is scientific, effective and creative because of the promotion study. Next, these results will be gradually introduced into our concrete topic, which is two-category DTRS reduction, and interesting new content will emerge.

5. Double preservation (CR-Preservation) and CRP-Reduct

The above-mentioned studies are based on only the D-Table or the formal structure thereof and can be generally applied to D-Table and other models. To a certain extent, these results provide sufficient or necessary conditions for D-Table regardless of the model technology employed. Next, we investigate two-category DTRS reduction, and the basic reduct concept will be established and compared in this section. Here, two subsections are provided to discuss the reduction criterion and concept.

5.1. Double preservation (CR-Preservation)

Proposition 5.1. *In* $C \vec{\to} B, IND(B) = IND(C) \Rightarrow cst_B(U) = cst_C(U)$.

K-Preservation on C and C-Preservation on C, D have already provided a basic framework for reduction targets. Proposition 5.1 presents the two-target relationship, i.e., K-Preservation deduces C-Preservation, but the opposite does not hold. Thus, K-Preservation is a high and natural reduction target, whereas C-Preservation is a low and basic reduction target. For the reduction targets, the two concepts define the supremum and infimum and provide the largest necessary range. Furthermore, this basic range should be improved according to another basic factor – region preservation (R-Preservation).

Proposition 5.2. *C-Preservation and R-Preservation are equivalent in the Pawlak model.*

Except for C-Preservation, the Pawlak reduct usually uses a region criterion, namely C-POS preservation because of the monotonicity of C-POS; moreover, C-POS preservation is usually related to R-Preservation, especially in the two-category case. Thus, this equivalence conclusion becomes natural in Proposition 5.2. Here, the region target is first analyzed. As is well known, C-POS has non-monotonicity in PRS models, which actually exhibits new uncertainty, and thus, the region criterion becomes a focus and a source of difficulty for quantitative attribute reduction. Here, the R-Preservation criterion will be established according to two-category DTRS, which can also be generalized.

According to the closed word assumption (CWA) [29], the initial attributes are complete in D-Table. Therefore, the initial D-Table is usually perfect and fundamental, whereas attribute reduction can remove redundant information according to reduction targets. Thus, scientific reduct definitions are especially important because unreasonable reduct concepts can wear

away and even distort the initial information. Under this idea, K-Preservation is clearly the safest approach that offers high accuracy, but reduction targets must be considered in depth for generalization to other applications. As a result, a reasonable criterion in fact exists in the balance between accuracy and generalization. Regions mainly extract condensed structures for semantics and other applications, thus becoming the basic application units. Hence, R-Preservation is a reasonable strategy because it provides the same region structure with respect to the initial D-Table and thus can act as a fundamental principle and a common requirement.

Principle 5.3 (*R-Preservation Principle*). Attribute reduction should consider R-Preservation, i.e., the three regions POS, BND and NEG do not change.

Regions can link microscopic and macroscopic information; thus, they correspond to application structures and underlie model applications. For attribute reduction, R-Preservation is a natural and normal criterion in the region monotonicity environment, and furthermore, it should be a more necessary requirement in the region non-monotonicity case. In other words, R-Preservation is a natural and certain strategy in a complex quantitative environment. Thus, the R-Preservation Principle is rational and effective. Here, we want to further add powerful evidence for the concrete two-category DTRS. Based on the modeling process related to the Bayesian risk-decision, three-way decisions are already the optimal decisions; thus, we can conclude that the three regions inherit optimal structures and also exhibit equality. Therefore, the R-Preservation Principle becomes a necessary standard for two-category DTRS reduction. In particular, the R-Preservation Principle can be generalized to attribute reduction theory, especially for multi-category cases. Based on the reasonable classification of regions, R-Preservation connotes the preservation of not only C-POS and C-BND but also the preservation of basic structural units that consist of C-POS or C-BND.

The R-Preservation reduct and core can be normally defined, which yields results similar to those for the KP-Reduct and CP-Reduct. Next, an example is provided to illustrate the concept of R-Preservation.

Example 3. For D-Table in Example 2 (i.e., Table 1), let $\alpha = 0.8, \beta = 0.2$. $\mathrm{POS}_C^{\alpha,\beta}(X) = [x]_7, \mathrm{BND}_C^{\alpha,\beta}(X) = [x]_6, \mathrm{NEG}_C^{\alpha,\beta}(X) = [x]_1 \cup [x]_2 \cup [x]_3 \cup [x]_4$. (1) In $\{c_1, c_2, c_3\} \xrightarrow{-c_3} \{c_1, c_2\}$, only $[x]_6$ and $[x]_7$ are merged; suppose that $[x]_6 \cup [x]_7 = [x]_*$; then, $P(X|[x]_*) = 0.8$, which indicates that $[x]_* \subseteq \mathrm{POS}_C^{\alpha,\beta}(X)$. Then, $\mathrm{POS}_C^{\alpha,\beta}(X) = [x]_6 \cup [x]_7, \mathrm{BND}_C^{\alpha,\beta}(X) = \phi, \mathrm{NEG}_C^{\alpha,\beta}(X) = [x]_1 \cup [x]_2 \cup [x]_3 \cup [x]_4$. Thus, deleting c_3 leads to a region change; concretely, BND is reduced while POS is enlarged, which also suggests that the region has non-monotonicity. Hence, this reduction action is not allowed under the R-Preservation Principle. (2) In $\{c_1, c_2, c_3\} \xrightarrow{-c_2} \{c_1, c_3\}$, the three regions do not change (although there is a combination with respect to $[x]_1, [x]_2, [x]_3, [x]_4$); thus, it is allowed for the R-Preservation Principle. (3) For R-Preservation, there is only one reduct $\{c_1, c_3\}$, and the core is $\{c_3\}$. (4) According to (1), only $[x]_6$ exhibits a region change when deleting c_3, and it is transferred from the old BND to new POS. This action appears to yield a better result because the deferment decision becomes the acceptance decision for the solely changed $[x]_6$. Thus, can $\{c_1, c_2\}$ be preferentially chosen as a reduct (in spite of the R-Preservation Principle)? If $\{c_1, c_2\}$ becomes a reduct, then a better certainty result appears to be obtained; however, the approach is followed only by deleting an attribute c_3; thus, this hypothesis exhibits a contradiction. According to the optimal DTRS mechanism, the deferment decision is initially the best for $[x]_6$; thus, $[x]_6$ should always use the deferment decision, even in attribute reduction. This analysis also illustrates the rationality of R-Preservation.

Based on the above-discussed mechanism and example analyses, it is observed that regions exhibit the fundamental function and requirement in qualitative classification, which is similar to certainty/uncertainty classification. However, the new qualitative classification requires quantitative thresholds. Thus far, we have established two fundamental principles. C-Preservation and R-Preservation are actually equivalent in Pawlak reduction. However, they are segregated in quantitative models, a fundamental and favorable fact that will yield new situations and novel studies.

Theorem 5.4 (*Preservation Separability*). *C-Preservation and R-Preservation are separate in the two-category DTRS model.*

$\mathrm{BND}_C^{\alpha,\beta}(X) \subseteq \mathrm{BND}_C(X)$; thus, C-Preservation and R-Preservation have been split into two different parts. Therefore, preservation separability is verified. For generalization, C-Preservation and R-Preservation are associated with qualitative and quantitative content, respectively, and thus, the preservation separability still holds for the quantitative PRS models.

Two fundamental requirements are now established, i.e., C-Preservation and R-Preservation. Using only one may not be sufficient; Examples 2 and 3 specifically provide the appropriate direct evidence. In view of both the necessity and separability of the two criteria, the integration approach naturally becomes a scientific and perfect strategy. CR-Preservation (i.e., double preservation), a novel concept, is defined to describe both C-Preservation and R-Preservation and provides the required harmony between these two ideas. Next, the CR-Preservation principle and reduct will be proposed, which underlie the next in-depth investigation.

Principle 5.5 (*CR-Preservation Principle*). Attribute reduction should consider both C-Preservation and R-Preservation, i.e., CR-Preservation.

Theorem 5.6 (*Five-Region Preservation*). *In two-category DTRS reduction, CR-Preservation actually preserves the (complete) five regions:* $\mathrm{POS}_C(X), \mathrm{NEG}_C(X), \mathrm{BND}_C(X) \cap \mathrm{POS}_C^{\alpha,\beta}(X), \mathrm{BND}_C^{\alpha,\beta}(X), \mathrm{BND}_C(X) \cap \mathrm{NEG}_C^{\alpha,\beta}(X)$, *or only the (dependent) three regions:* $\mathrm{BND}_C(X) \cap \mathrm{POS}_C^{\alpha,\beta}(X), \mathrm{BND}_C^{\alpha,\beta}(X), \mathrm{BND}_C(X) \cap \mathrm{NEG}_C^{\alpha,\beta}(X)$, *or only the (dependent) three regions:* $\mathrm{BND}_C(X), \mathrm{POS}_C^{\alpha,\beta}(X), \mathrm{NEG}_C^{\alpha,\beta}(X)$.

The CR-Preservation Principle (Principle 5.5) is natural and scientific and defines a fundamental framework and underlies the in-depth exploration of attribute reduction. According to the two-category DTRS reduction, Theorem 5.6 provides concrete results for CR-Preservation. In the regional view, CR-Preservation is actually the (complete) five-region or the (dependent) three-region preservation. In other words, only the five or three regions can fully determine the CR-Preservation structure. Thus, Theorem 5.6 is specifically called *Five-Region Preservation*. First, Five-Region Preservation can be generalized to the usual quantitative and extended models. The classical Pawlak reduction preserves C-POS or consistency and, in fact, preserves the (complete) three regions or the (dependent) single region: C-BND, where the criterion corresponds to *Three-Region Preservation*. For the qualitative Pawlak model, Five-Region Preservation will naturally degenerate into Three-Region Preservation, whereas for the quantitative models, Three-Region Preservation is no longer applicable. Therefore, Five-Region Preservation is an extension of the classical approach and is a specific feature of the extended and quantitative models. Moreover, based on the calculation mechanism of rough set regions, Five-Region Preservation usually exhibits only a linear difference, and thus, it has the same computational feasibility level as Three-Region Preservation.

Some basic properties are provided for the next reduct construction, and certain symbols are defined. Similarly to KP/KNP, CRP is used to denote CR-Preservation, whereas CRNP denotes the opposite. Note that $CRP(C) = \{a \in C | C \xrightarrow{-a} C - \{a\} \models CRP\}$ and $CRNP(C) = \{a \in C | C \xrightarrow{-a} C - \{a\} \models CRNP\}$. $\forall A \subseteq C, CRP(A)$ and $CRNP(A)$ have a similar expression. Suppose that $C_0 \subseteq C'$ and that $. \xrightarrow{-C_0} .$ is K-Coarsening in a sequence T for $C \xrightarrow{-C'} B$.

Proposition 5.7 (*Infection Principle*).

$$\begin{aligned} &\exists T, \exists . \xrightarrow{-C_0} . \models CRNP \Longleftrightarrow C \xrightarrow{-C'} B \models CRNP, \\ &\exists C_0^* \subseteq C', C \xrightarrow{-C_0^*} C - C_0^* \models CRNP \Longleftrightarrow C \xrightarrow{-C'} C - C' \models CRNP. \end{aligned} \tag{17}$$

Proposition 5.8 (*Purity Principle*).

$$\begin{aligned} &C \xrightarrow{-C'} B \models CRP \Longleftrightarrow \forall T, \forall . \xrightarrow{-C_0} . \models CRP, \\ &C \xrightarrow{-C'} C - C' \models CRP \Longleftrightarrow \forall C_0^* \subseteq C', C \xrightarrow{-C_0^*} C - C_0^* \models CRP. \end{aligned} \tag{18}$$

Proposition 5.9 (*Core Principle*).

$$\begin{aligned} &CRNP(C) \cap C' \neq \phi \Rightarrow C \xrightarrow{-C'} B \models CRNP, \\ &C \xrightarrow{-C'} B \models CRP \Rightarrow C' \subseteq CRP(C). \end{aligned} \tag{19}$$

In the DTRS model, a change in region exhibits complexity, such as non-monotonicity. The Infection and Purity Principles ensure the stability of CR-Preservation in the complex environment and address our concern about whether the global CRP can be implemented by the local CRNP in K-Coarsening. More importantly, the Core Principle further reflects the core existence. Thus, the Infection and Purity Principles and the Core Principle harmoniously underlie CR-Preservation attribute reduction, and in turn, the reduction construction can be naturally carried out according to the Pawlak reduction approach.

5.2. CRP-Reduct

Next, we study the CR-Preservation reduct based on CR-Preservation and the relevant properties.

Definition 5.10. If $a \in CRNP(C)$, then a is independent. $\forall b \in B$, if $b \in CRNP(B)$, then B is independent. B is a CRP-Reduct of C, if $C \xrightarrow{-} B \models CRP$ and B is independent. $CRNP(C)$ is the core of C, which is denoted as $Core^{\alpha,\beta}(C)$.

Definition 5.11 (*CRP-Reduct*). B is CRP-Reduct of C, denoted as $B \in Red^{\alpha,\beta}(C)$, if (1) $C \xrightarrow{-} B \models CRP$; (2) $\forall B' \subset B, B \xrightarrow{-} B' \models CRNP$.

In the CR-Preservation Principle, CRP-Reduct is equivalently defined by Definitions 5.10 and 5.11. With respect to joint sufficiency, CRP-Reduct connotes the CR-Preservation target, which is located between K-Preservation and C-Preservation; with respect to individual necessity, set maximality and set independency become equivalent for the reduction target.

Theorem 5.12 (*CRP-Reduct Extension*). *Based on CR-Preservation, CRP-Reduct extends the qualitative Pawlak reduct to the quantitative environment.*

Theorem 5.12 reflects the CRP-Reduct extension of the Pawlak reduct. Thus, the Pawlak reduct-based reduction theory has been expanded to the quantitative models by CRP-Reduct, and the Pawlak reduct acts as only a special case of CRP-Reduct. Therefore, the novel CRP-Reduct becomes a natural and rational expansion of the Pawlak reduct and holds great significance for quantitative attribute reduction.

We finally establish the two-category DTRS-Reduction concept: CRP-Reduct. Next, we summarize some conclusions regarding generalization with respect to the category case. For simplicity, the two-category and multi-category cases are denoted as Cases I and II, respectively. Our approach concerning the preservation and reduct not only directly applies to Case I but is also suitable for Case II. C-Preservation is clear in both Cases I and II. For R-Preservation, the generalization can be implemented by preserving both the classification regions and some of their internal structures in Case II. In fact, this paper's descriptive term (in Case I) concerns only set regions rather than classification regions due to the former's sufficiency. Furthermore, CR-Preservation and CRP-Reduct can be naturally generalized by the integration of C-Preservation and R-Preservation.

Next, our approach will be compared with other existing methods. For DTRS-Reduction, there are three main reducts [8]: a positive-based reduct [61], a nonnegative-based Reduct [8] and a cost-based reduct [8], which are all based on Case II and are reviewed in Section 2. The reduction targets naturally become the subject of comparison; thus, they can be compared in Case I by degeneration (in Proposition 2.5) and in Case II by generalization (from the above analysis). (1) The positive-based reduct aims to preserve C-POS in Case II, whereas in Case I, it cannot necessarily preserve POS $\cup$ NEG and POS and NEG. The nonnegative-based reduct aims to preserve $C-\mathrm{POS}\cup C-\mathrm{BND}$ in Case II, whereas it preserves U in Case I (which is meaningless). Only R-Preservation aims to preserve POS, BND and NEG in Case I, whereas it preserves not only C-POS and C-BND but also some of their internal components in Case II. Thus, R-Preservation is rational and stable in both the degenerate and general cases. Furthermore, CRP-Reduct can also preserve the consistency/inconsistency in Cases I and II. Thus, CRP-Reduct exhibits improvements for both the positive-based and nonnegative-based reducts. (2) CRP-Reduct and the positive-based and nonnegative-based Reducts employ a structural approach because they consider classification regions or set regions (which are related to application structures), whereas the cost-based reduct employs a measurement method in which the reduction target is the minimum cost. The cost-based reduct utilizes basic cost information and obtains the final optimal measure, but it neglects the basic structure and thus lacks the appropriate link to the classical Pawlak reduct. Therefore, based on CR-Preservation, we will make some improvements by considering both structure and measurement, which constitute our work on the optimal reduct in Section 6.3 (where C-Preservation is clearly a good strategy for considering cost).

Compared to the existing methods regarding DTRS-Reduction, our approach exhibits improvements by introducing many innovative elements. In fact, our work outlined above has already fully verified the advancement and superiority of CR-Preservation and CRP-Reduct, such as their extension to the classical targets and reducts. In particular, C-Preservation, R-Preservation, CR-Preservation and CRP-Reduct are all novel concepts and provide valuable content for the exploration of PRS-Reduction. Of course, many further studies must be conducted, such as on the generalization of the implementation. Next, we discuss concrete constructions for CRP-Reduct.

Theorem 5.13 (*Core Property*).
$Core^{\alpha,\beta}(C)=\bigcap Red^{\alpha,\beta}(C)$.

Proof. (1) Suppose that $a\notin\bigcap Red^{\alpha,\beta}(C)$; therefore, $\exists B\in Red^{\alpha,\beta}(C)$ but $a\notin B$. Thus, $C\overset{-(C-B)}{\rightarrow}B\models CRP$ and $a\in C-B\subseteq C$; based on the Purity Principle (Proposition 5.8), $C\overset{-a}{\rightarrow}C-\{a\}\models CRP$, i.e., $a\in CRP(C)$ and $a\notin CRNP(C)$. Hence, $a\notin Core^{\alpha,\beta}(C)$, and $Core^{\alpha,\beta}(C)\subseteq\bigcap Red^{\alpha,\beta}(C)$. (2) Suppose that $a\notin Core^{\alpha,\beta}(C)$; therefore, $C\overset{-a}{\rightarrow}C-\{a\}\models CRP$. $\exists B\in Red^{\alpha,\beta}(C-\{a\})$; hence, $C-\{a\}\rightarrow B\models CRP$ and B is independent. Based on the natural composite property, $C\overset{-}{\rightarrow}B\models CRP$. Thus, $B\in Red^{\alpha,\beta}(C)$ but $a\notin B$. Hence, $a\notin\bigcap Red^{\alpha,\beta}(C)$, and $\bigcap Red^{\alpha,\beta}(C)\subseteq Core^{\alpha,\beta}(C)$. □

Theorem 5.13 reflects two points. First, the CRP-Reduct Core has extended the Pawlak reduct core and has a similar fundamental position, such as the construction function for reducts. Second, the Purity Principle is utilized to implement the proof, which becomes fundamental in the quantitative environment.

Definition 5.14. $S=(U,C\cup D),A\subseteq C$, and the region dependency-degrees are defined as follows:

$$\gamma_A^P(D)=\frac{|\mathrm{BND}_A(X)\cap\mathrm{POS}_A^{\alpha,\beta}(X)|}{|U|},\gamma_A^B(D)=\frac{|\mathrm{BND}_A^{\alpha,\beta}(X)|}{|U|},\gamma_A^N(D)=\frac{|\mathrm{BND}_A(X)\cap\mathrm{NEG}_A^{\alpha,\beta}(X)|}{|U|}.$$

Furthermore, $\gamma_A^{\alpha,\beta}(D)=(\gamma_A(D),\gamma_A^P(D),\gamma_A^N(D))$ is called the dependency-degree array of D on A.

In the Pawlak model, the measure $\gamma_A(D)$ is the addition fusion of the internal region measures and is monotonic such that it completely describes the Pawlak reduct. For the DTRS model, the region change becomes not only non-monotonic but also uncertain; thus, we utilize the classified description strategy rather than the fused approach. $\gamma_A(D)$ describes $\mathrm{POS}_A(X),\mathrm{NEG}_A(X)$, while $\gamma_A^P(D),\gamma_A^N(D)$ control $\mathrm{BND}_A(X)\cap\mathrm{POS}_A^{\alpha,\beta}(X),\mathrm{BND}_A^{\alpha,\beta}(X),\mathrm{BND}_A(X)\cap\mathrm{NEG}_A^{\alpha,\beta}(X)$. According to Five-Region Preservation (Theorem 5.6), the array $\gamma_A^{\alpha,\beta}(D)$ can fully reflect the dependency of knowledge D on knowledge A. In particular, the region dependency degrees $\gamma_A^P(D),\gamma_A^B(D)$ and $\gamma_A^N(D)$ describe the proportions of the objects that are classified into the three specific regions. Moreover, $\gamma_A^P(D),\gamma_A^B(D),\gamma_A^N(D),\gamma_A(D)\in[0,1]$, and $\gamma_A^P(D)+\gamma_A^B(D)+\gamma_A^N(D)=1-\gamma_A(D)$.

Theorem 5.15. $A_1 \subseteq A_2 \subseteq C$.

(1) *In the Pawlak model,* $\gamma_A^{\alpha,\beta}(D) = (\gamma_A(D), 0, 0)$, *and* $A_2 \xrightarrow{-} A_1 \models CRP \Longleftrightarrow \gamma_{A_1}^{\alpha,\beta}(D) = \gamma_{A_2}^{\alpha,\beta}(D) \Longleftrightarrow \gamma_{A_1}(D) = \gamma_{A_2}(D)$;

(2) *In the DTRS model,* $A_2 \xrightarrow{-} A_1 \models CRP \Rightarrow \gamma_{A_1}^{\alpha,\beta}(D) = \gamma_{A_2}^{\alpha,\beta}(D)$.

Theorem 5.15 describes the property of the dependency-degree array. Compared to $\gamma_A(D), \gamma_A^{\alpha,\beta}(D)$ exhibits the extension feature with dimensions and provides an equivalent description for the Pawlak model CR-Preservation. Moreover, the equality of the dependency-degree array acts as the only necessary condition for DTRS CR-Preservation due to the complexity of the region change. Thus, two interesting problems arise. What is suitable heuristic information for the attributes? How can heuristic algorithms for CRP-Reduct be constructed?

CR-Preservation is located between C-Preservation and K-Preservation; as a result, CRP-Reduct adopts an intermediate position between CP-Reduct and KP-Reduct (which are actually Pawlak reducts in D-Table and KRS, respectively). Thus, the CRP-Reduct calculation could adopt two directional strategies that are based on Pawlak reducts, i.e., CRP-Reduct can be searched from Pawlak reducts in D-Table and KRS, respectively. In this context, two algorithms that are connected to the core are provided for CRP-Reduct computation.

Algorithm 1. CP-Reduct-based algorithm for CRP-Reduct

Input:
D-Table $(U, C \cup D)$, thresholds α, β;
Output:
CRP-Reduct subset $Red_{sub}^{\alpha,\beta}(C)$;
1: Obtain CP-Reduct subset $Red_{sub}(C)$;
2: Calculate the CRP-Reduction core $Core^{\alpha,\beta}(C)$. In $Red_{sub}(C)$, obtain the subset $Red_{*}^{\alpha,\beta}(C) = \{A \in Red_{sub}(C) | A \supseteq Core^{\alpha,\beta}(C)\}$;
3: In $Red_{*}^{\alpha,\beta}(C)$, obtain subset $Red_{**}^{\alpha,\beta}(C)$, whose elements satisfy the R-Preservation condition;
4: In $Red_{**}^{\alpha,\beta}(C)$, obtain the subset $Red_{sub}^{\alpha,\beta}(C)$ by considering set independency/maximality;
5: **return** $Red_{sub}^{\alpha,\beta}(C)$.

Algorithm 2. KP-Reduct-based algorithm for CRP-Reduct

Input:
D-Table $(U, C \cup D)$, thresholds α, β;
Output:
CRP-Reduct $B \in Red^{\alpha,\beta}(C)$;
1: Obtain KP-Reduct B_K;
2: Obtain the CRP-Reduction core $Core^{\alpha,\beta}(C)$;
3: In the range $Core^{\alpha,\beta}(C) \subseteq B \subseteq B_K$, completely search B to satisfy both the CR-Preservation and independency conditions;
4: **return** B.

In Algorithm 1, Step 1 obtains CP-Reduct subset $Red_{sub}(C)$ using the D-Table Pawlak reduct. Steps 2, 3 and 4 search the hierarchical subsets $Red_{*}^{\alpha,\beta}(C), Red_{**}^{\alpha,\beta}(C)$ and $Red_{sub}^{\alpha,\beta}(C)$ by the progressive conditions for the core, R-Preservation and maximality, respectively. Step 5 yields the final CRP-Reduct subset $Red_{sub}^{\alpha,\beta}(C)$. This algorithm uses the layer-by-layer condition reinforcement, which is partially due to the stronger development of the reduction target. Thus, if more Pawlak reducts are initially provided, better results can be obtained. In Algorithm 2, Step 1 yields KP-Reduct B_K by KRS Pawlak reduct. Step 2 computes the CRP-Reduction core. Between the core and KP-Reduct, Step 3 searches the independent CR-Preservation set B, which is the final result. Based on Proposition 7.2, this algorithm is convergent and effective because it necessarily obtains all CRP-Reducts included in the given KP-Reduct. In short, both algorithms utilize the Pawlak reduct platform, but they involve the D-Table and KRS, respectively. The CP-Reduct-based algorithm can obtain a certain subset for all of the CRP-Reducts, whereas a KP-Reduct-based algorithm can obtain all of the included CRP-Reducts for the given KP-Reduct. In particular, both algorithms exploit the hierarchical reduct relationship. Section 7 provides relevant discussion in this respect and further analyzes both algorithms using a D-Table example. Moreover, both algorithms adopt a hierarchical strategy. On the other hand, an integrated technology can also be considered for the CRP-Reduct computation, where CR-Preservation is directly required, and this method is similar to the classical approach in RS theory; furthermore, heuristic attribute information is expected.

6. Structure targets and general reducts

CR-Preservation and CRP-Reduct have already provided the basic reduction results for two-category DTRS. Using granular computing, our objective is to conduct an in-depth exploration from a structural point of view and obtain further general benefits. First, attribute reduction is mainly linked to knowledge structures of condition attributes, which is partially related to K-Coarsening (Section 3). Second, concrete reduction targets (such as C-Preservation, R-Preservation and CR-Preservation) are mainly associated with qualitative classifications (such as Three-Region Preservation and Five-Region Preservation). Thus, attribute reduction with reduction targets has the following operational space – interaction within qualitative classifications. All of these concepts motivate the abstract and universal approach to structural reduction described in this section. In particular, our preliminary work is based on two-category DTRS and CR-Preservation, but the concepts and results thereof lie far beyond the concrete framework.

6.1. Structure targets (ST)

This subsection proposes the structure targets for constructing the structural basis.

Proposition 6.1. *K-Preservation deduces CR-Preservation, whereas CR-Preservation deduces C-Preservation and R-Preservation. However, neither of the opposites hold.*

Proposition 6.1 indicates the clear relationships among the relevant reduction targets. The development line, C-Preservation→CR-Preservation→ K-Preservation, mainly seeks both finer structure for condition knowledge and higher requirements for reduction targets. Here, we introduce *structure targets* to link condition knowledge and reduction targets and to further propose *general reducts*. Note that CR-Preservation and K-Preservation have formed a new infimum and supremum for reduction targets; thus, they finally establish the scientific framework with a more reasonable and necessary reduction range. Thus, our concrete discussion occurs mainly within this framework. Here, the knowledge dependency/deducibility relation "⇐"/"⇒" can be easily generalized to describe the arbitrary granular structure.

Example 4. Let $E/IND(A) = \{[x]_A^1, [x]_A^2, [x]_A^3\}$. According to the three knowledge granules, only five types of granular structures exist, i.e., $st_1 = \{[x]_A^1, [x]_A^2, [x]_A^3\}, st_2 = \{[x]_A^1 \cup [x]_A^2, [x]_A^3\}, st_3 = \{[x]_A^1, [x]_A^2 \cup [x]_A^3\}, st_4 = \{[x]_A^1 \cup [x]_A^3, [x]_A^2\}, st_5 = \{[x]_A^1 \cup [x]_A^2 \cup [x]_A^3\}$. $ST_A(E) = \{st_1, \ldots, st_5\}$ is a complete lattice with respect to "⇐", whereas st_1 and st_5 become the greatest and least elements, respectively. Moreover, $st_1 \Rightarrow st_2, st_3, st_4 \Rightarrow st_5$.

Example 4 provides a vivid example illustrating the granular structure. Based on the dependency relation, the roughest st_5 is called the least element, which corresponds to the lowest structural requirement. Moreover, "⇒" is usually used for simplicity. Next, some descriptions are provided for the one-dimensional case in RS theory. $S = (U, C \cup D)$, $A \subseteq C, E \subseteq U, \underline{apr}_A E = \overline{apr}_A E$. Let st denote the arbitrary structure on $E/IND(A)$, and $ST_A(E) = \{st\}$. $\forall st_1, st_2 \in ST_A(E), st_1 \Rightarrow st_2$ is defined as follows: $\forall g_1 \in st_1, \exists g_2 \in st_2, g_1 \subseteq g_2$ holds. Thus, $(ST_A(E), \Leftarrow)$ is a complete lattice.

Definition 6.2 (*Structure Target*). Let $st^{P_0} \in ST_C(\mathrm{POS}_C(X)), st^{N_0} \in ST_C(\mathrm{NEG}_C(X)), st^P \in ST_C(\mathrm{BND}_C(X) \cap \mathrm{POS}_C^{\alpha,\beta}(X))$, $st^B \in ST_C(\mathrm{BND}_C^{\alpha,\beta}(X))$, $st^N \in ST_C(\mathrm{BND}_C(X) \cap \mathrm{NEG}_C^{\alpha,\beta}(X))$. Thus, $st^C = (st^{P_0}, st^{N_0}, st^P, st^B, st^N)$ is a structure target (ST), and $ST^C = \{st^C\}$ is the ST set. Suppose that

$$\begin{aligned} st_1^C &= (st_1^{P_0}, st_1^{N_0}, st_1^P, st_1^B, st_1^N) \in ST^C, \\ st_2^C &= (st_2^{P_0}, st_2^{N_0}, st_2^P, st_2^B, st_2^N) \in ST^C. \end{aligned} \tag{20}$$

If $st_1^{P_0} \Rightarrow st_2^{P_0}, st_1^{N_0} \Rightarrow st_2^{N_0}, st_1^P \Rightarrow st_2^P, st_1^B \Rightarrow st_2^B, st_1^N \Rightarrow st_2^N$, then we define $st_1^C \Rightarrow st_2^C$ and conclude that st_1^C deduces st_2^C while st_2^C depends on st_1^C.

Based on the premise of CR-Preservation, Definition 6.2 formally defines ST according to the knowledge structure with respect to the five fundamental regions. CR-Preservation, i.e., Five-Region Preservation, determines the entire complete classification with respect to the five regions. Thus, ST aims to describe, in depth, the five internal structures. From another perspective, st^C is a type of specific structure that has the range ST^C. In particular, a partial order $\Rightarrow$ on ST^C is naturally defined by the Cartesian combination of the internal partial orders and reflects the five-structure dependency/deducibility. Moreover, ST – a quintuple form – corresponds to five dimensions, but only one component can be used to conduct a representative and effective analysis. Thus, $(ST_A(E), \Leftarrow)$ and Example 4 provide a simple but powerful illustration.

Theorem 6.3. *$(ST^C, \Leftarrow)$ is a complete lattice, and the greatest and least elements are $st_{KP}^C = (U/C|\mathrm{POS}_C(X),\ U/C|\mathrm{NEG}_C(X), U/C|(\mathrm{BND}_C(X) \cap \mathrm{POS}_C^{\alpha,\beta}(X)), U/C|\mathrm{BND}_C^{\alpha,\beta}(X), U/C|(\mathrm{BND}_C(X) \cap \mathrm{NEG}_C^{\alpha,\beta}(X)))$, $st_{CRP}^C = (\mathrm{POS}_C(X), \mathrm{NEG}_C(X), \mathrm{BND}_C(X) \cap \mathrm{POS}_C^{\alpha,\beta}(X), \mathrm{BND}_C^{\alpha,\beta}(X), \mathrm{BND}_C(X) \cap \mathrm{NEG}_C^{\alpha,\beta}(X))$, respectively, where $U/C|E$ indicates the restriction structure of knowledge $U/IND(C)$ with respect to set E. Thus, $\forall st^C \in ST^C, st_{KP}^C \Rightarrow st^C \Rightarrow st_{CRP}^C$.*

Theorem 6.3 describes the mathematical structural feature of $(ST^C, \Leftarrow)$. The greatest element st^C_{KP} is actually $U/IND(C)$, and it corresponds to the finest structure and the highest target. The least element st^C_{CRP} actually represents the five whole blocks, and it corresponds to the roughest structure and the lowest requirement. st^C_{KP} and st^C_{CRP} provide the ST supremum and infimum; therefore, a general ST is usually located between them and can be obtained by roughening st^C_{KP} or refining st^C_{CRP}. Next, let us analyze the $(ST^C, \Leftarrow)$ significance for attribute reduction. C is the initial complete condition attribute, and thus, arbitrary feasible knowledge (including the knowledge defined by a reduct) can be represented by $U/IND(C)$. Therefore, ST^C includes all possible knowledge structures based on the premise of CR-Preservation. In other words, $(ST^C, \Leftarrow)$ provides a fundamental platform for the CR-Preservation Principle and a relevant discussion thereof (such as CR-Reduction), where the dependency/deducibility underlies the reduction reasoning; hence, the complete lattice establishes a wide and deep background. Moreover, the knowledge also includes the realizability problem, which relies on C.

Next, we will analyze the structural essence of the reduction target. The reduction target essentially corresponds to certain structures with respect to $U/IND(C)$; thus, it can be described by $(ST^C, \Leftarrow)$. Here, ST and the complete lattice are utilized.

Definition 6.4 (*ST Reduction Target*). $\forall st^C \in ST^C, MT_C(st^C) = \{st^C_* \in ST^C | st^C_* \Rightarrow st^C\}$ is defined as the reduction target on st^C.

Proposition 6.5. *st^C is the infimum of set $MT_C(st^C)$, which is denoted as $inf(MT_C(st^C)) = st^C$. Moreover, $\forall st^C_* \in MT_C(st^C), st^C_* \Rightarrow st^C$, which is denoted as $MT_C(st^C) \Rightarrow st^C$.*

Based on the complete lattice, $MT_C(st^C)$ has a unique existence and depends on the given st^C; moreover, st^C is the infimum of $MT_C(st^C)$. By $(ST^C, \Leftarrow)$, Definition 6.4 provides the formal definition of the ST reduction target, where infimum st^C is the lowest structure requirement. $MT_C(st^C) \Rightarrow st^C$ in Proposition 6.5 shows that the ST Reduction Target is a set of structure targets that can deduce the given structure target. Moreover, attribute reduction aims to obtain structure targets in $MT_C(st^C)$ rather than only st^C. Generally, the typical reduction target can be viewed as the set of certain structures that include the infimum (i.e., the lowest structure requirement). For example, the basic CR-Preservation target aims to seek deducible structures for st^C_{CRP}, which is the lowest structure criterion.

Theorem 6.6. *$MT_C : st^C \to MT_C(st^C)$ is an injective mapping on ST^C. Thus, there is a one-to-one mapping between ST^C and $\{MT_C(st^C) | st^C \in ST^C\}$. Moreover, $inf : MT_C(st^C) \to st^C$ is the inverse mapping of MT_C.*

From a mathematical point of view, Theorem 6.6 is clear. More importantly, it exhibits a one-to-one corresponding relationship between ST and the ST Reduction Target. Thus, only the infimum can fully determine and describe the ST Reduction Target. Hence, the lowest structure requirement is important and can be extensively used to complete the description with respect to the ST Reduction Target. This concept will be extensively utilized to describe the reduction targets of C-Preservation, CR-Preservation, ST-Preservation and K-Preservation in Section 7.

6.2. ST-Preservation and STP-Reduct

Next, K-Coarsening is introduced. As in the case of the classical and above-described reduction targets and reducts, we can require preservation for a specific structure target (i.e., ST-Preservation) and develop the corresponding reduct (i.e., STP-Reduct).

Definition 6.7 (*ST-Preservation*). $\forall st^C \in ST^C$, K-Coarsening $C \bar{\to} B$ satisfies ST-Preservation if (1) $C \bar{\to} B \models CRP$; (2) $U/IND(B) \in MT_C(st^C)$, i.e., $U/IND(B) \Rightarrow st^C$.

Definition 6.7 simulates the K-Coarsening's action for preserving the ST Reduction Target, which is called ST-Preservation. Item (1) concerns the premise of CR-Preservation, whereas item (2) requires the deducible implementation for knowledge B. Note that ST-Preservation does not maintain st^C but deduces st^C, and B corresponds to only one stable structure $U/IND(B)$. Moreover, ST-Preservation can be related to the applicable requirement.

Proposition 6.8. *K-Preservation deduces ST-Preservation, whereas ST-Preservation deduces CR-Preservation. The converse does not hold in either case.*

Proposition 6.8 exhibits the basic position of ST-Preservation, which is located between CR-Preservation and K-Preservation. In other words, ST-Preservation is stronger than CR-Preservation; as a result, stronger or deeper reducts usually exist. Moreover, compared to CR-Preservation, ST-Preservation might be a more reliable criterion because it is more closely related to K-Preservation (which is the surest strategy). Furthermore, K-Preservation and ST-Preservation provide a new reduction range. Based on ST-Preservation, the corresponding reduct can be naturally defined.

Definition 6.9 (*STP-Reduct*). $\forall st^C \in ST^C, B$ is STP-Reduct on st^C, if

(1) $C \bar{\to} B \models CRP$, and $U/IND(B) \Rightarrow st^C$;
(2) $\forall B' \subset B$, if $C \bar{\to} B' \models CRP$, but $U/IND(B') \Rightarrow st^C$ does not hold.

Definition 6.9 defines STP-Reduct for ST-Preservation, where the given structure target is viewed as the lowest target. Item (1) indicates that $C \bar{\rightarrow} B$ has the ST-Preservation feature, whereas item (2) requires the set maximality with respect to ST-Preservation. Moreover, the core can be normally defined.

Theorem 6.10 (*STP-Reduct Inclusiveness*).

Based on the premise of CR-Preservation, STP-Reducts include all attribute reducts. In particular, CRP-Reduct and KP-Reduct constitute two special cases of STP-Reducts. When $st^C = st^C_{CRP}, st^C = st^C_{KP}$, *STP-Reduct degenerates into CRP-Reduct and KP-Reduct.*

Theorem 6.11 (*STP-Reduct Hierarchy*).

For a given ST, STP-Reduct is a middle-level reduct between CRP-Reduct and KP-Reduct.

As discussed in Section 5, CR-Preservation and CRP-Reduct have already established the initial reduction framework for two-category DTRS. In this section, they are further developed by STP-Reduct. In view of the characteristics of the ST parameter, two points of view are provided by Theorems 6.10 and 6.11. (1) STP-Reduct can provide all reducts with the CR-Preservation property due to the complete change in ST. Thus, STP-Reducts become the general reducts for two-category DTRS and finally establish a complete framework. (2) STP-Reduct also exhibits a hierarchy, but not inclusiveness, and is viewed more as a type of middle level between CRP-Reduct and KP-Reduct. Thus, the suitable choice for ST will become a new problem due to the variability. To address this new issue, we next provide a measurement strategy and further propose the optimal reducts.

Herein, the generalization is explained beyond two-category DTRS-Reduction. Based on the above-described studies, we actually simulate the attribute reduction by ST and its lattice structure. *A reduction target is a set whose elements can deduce the given structure target, whereas the objective of attribute reduction is to seek condition attribute subsets whose knowledge structure can deduce the lowest structure requirement given.* Thus, structural reduction theory, which includes the concepts of ST, ST Reduction Target, ST-Preservation and STP-Reduct, can be extensively utilized. Section 7 will provide a D-table example to illustrate STP-Reduct and the reduction approach based on the aforementioned structures.

6.3. Optimal reducts

In accordance with the conventional dependency/deducibility approach, STP-Reduct is established and provides general reducts. In fact, ST carries specific information, which enables the argument to reflect dependent variables, such as concrete measures. For STP-Reduct, the optimal measures can tackle the ST choice problem. Multiple concrete measures have been developed in DTRS [51], which also provide the STP-Reduction value and space. From another perspective, we can pursue optimal measures in attribute reduction, i.e., the optimal reducts can be established for the given measures. In this context, this subsection will propose several optimal reducts based on the results reported in [8,51]. Note that the optimal reducts lie within the CR-Preservation framework, i.e., the pursuit of the optimal measure has a structural premise; thus, the optimal reducts will consider both the structure and measures.

Definition 6.12 (*Optimal Reduct I*).

Given a measure $(m, \leqslant)$ on 2^C, suppose that the optimal measure value concerns the minimum, $m_{opt} = \min(A)$, where $A \subseteq C$ and $C \bar{\rightarrow} A \models CRP$. B is called an optimal reduct with respect to $(m, \leqslant)$ if

(1) $C \bar{\rightarrow} B \models CRP, B = \arg m_{opt}$;

(2) $\forall B' \subset B$, if $C \bar{\rightarrow} B' \models CRP$, but $m(B') > m(B)$.

According to one measure $(m, \leqslant)$ on knowledge structures, Definition 6.12 proposes a type of optimal reduct. Based on the premise of CR-Preservation, m_{opt} is the optimal value. Thus, the optimal reduct aims to search the independent CR-Preservation set to reach the optimal value. Next, a specific optimal reduct is proposed by a concrete measure, which is the cost, and the cost formula is discussed first.

Proposition 6.13. *Let* $\lambda_{PP} = 0 = \lambda_{NN}, p = P(D_{max}([x]_A)|[x]_A)$; *thus, the cost on knowledge A is* $COST(A) =$

$$\sum_{x \in \mathrm{POS}_A^{\alpha,\beta}(X) \cap \mathrm{BND}_A(X)} (1-p)\lambda_{PN} + \sum_{x \in \mathrm{BND}_A^{\alpha,\beta}(X)} (p\lambda_{BP} + (1-p)\lambda_{BN}) + \sum_{x \in \mathrm{NEG}_A^{\alpha,\beta}(X) \cap \mathrm{BND}_A(X)} p\lambda_{NP}.$$

$COST(A)$ mainly represents the decision cost on knowledge $A \subseteq C$. For the two-category approach, the original formula [8] in Proposition 2.8 concerns only three set regions. Furthermore, the new formula in Proposition 6.13 involves the five regions of CR-Preservation, and the cost on Pawlak-POS and Pawlak-NEG are equal to 0 because $p = 1$ and $p = 0$, respectively. Concretely, the new cost reflects the following three points. (1) The cost denotes the structural requirement for rational CR-Preservation. (2) The computational regions are scientifically reduced because the two certainty regions do not incur

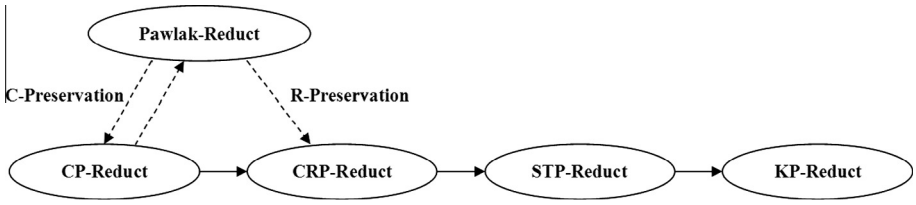

Fig. 1. Reduction hierarchy of four types of reduct.

the cost. (3) The formula originates from the two-category case, which allows it to clearly correspond to the DTRS modeling process. Thus, the new cost formula has improved upon the old one, at least at the two-category level. Moreover, the cost function can well reflect the merits of the qualitative classification with respect to certainty and uncertainty; in other words, the rationality of C-Preservation is also verified by the cost conclusion.

Definition 6.14 (*Optimal Reduct II*).
Let $COST_{opt} = \min_{A \subseteq C, C \xrightarrow{-} A \models CRP} COST(A)$. B is called an optimal reduct with respect to the cost if

(1) $C \xrightarrow{-} B \models CRP, B = \arg COST_{opt}$;
(2) $\forall B' \subset B$, if $C \xrightarrow{-} B' \models CRP$, but $COST(B') > COST(B)$.

Within the framework of Optimal Reduct I, Definition 6.14 further yields a concrete result based on the new cost formula, where $COST_{opt}$ indicates the minimum cost in the CR-Preservation range. Because of the introduction of risk information, this optimal reduct is more closely linked to the Bayesian risk decision of two-category DTRS; thus, it holds important value for semantics applications. In particular, Cost-based Reduct pursues the optimal cost regardless of the structure, whereas Optimal Reduct II considers the minimum cost based on the premise of rational structural; thus, Optimal Reduct II improves upon the cost-based reduct and is more rational, at least at the two-category level.

The above-described optimal reducts involve only a single measure. However, multiple measures can also be considered. Ref. [51] offers many results concerning this topic. Here, a type of optimal reduct is proposed based on the structures.

Definition 6.15 (*Optimal Reduct III*).
Suppose that $M = (m_1, m_2, \ldots)$ is a measure set on 2^C and each measure has the optimal minimum. B is an optimal reduct with respect to M if

(1) $C \xrightarrow{-} B \models CRP, m(B) \leqslant m(C), \forall m \in M$;
(2) $\forall B' \subset B$, if $C \xrightarrow{-} B' \models CRP$, then $m(B') \not\leqslant m(B), \forall m \in M$.

Based on the initial measure criterion with respect to C, Definition 6.15 searches the optimal reducts by considering the multiple measures given. Of course, CR-Preservation is still the structural premise.

Theorem 6.16 (*Optimal Reduct Merit*).
The optimal reducts can preserve the fundamental structure of CR-Preservation and can exhibit the optimal feature for a measure or multiple measures.

Theorem 6.16 reflects the merit of the three optimal reducts. Based on the premise of CR-Preservation, the reducts can reach the optimal measurement values. Thus, the optimal reducts enhance the basic CRP-Reduct and improve the existing reducts through their structure or measures. Hence, the reducts hold great significance for reduction theory, especially for measurement applications. Furthermore, optimal reducts can be generally explored by adopting the double-requirement concept for the structure and measures. Moreover, the measures can also be considered to be heuristic information for constructing reduction algorithms.

In summary, based on the premise of CR-Preservation, ST concerns the internal structures of the five regions. Within this structural framework, the general reducts are further explored. Based on CR-Preservation, the general reduction makes an adjustment to the feasible five internal structures; thus, it can realize reduction targets with respect to dependency/deducibility or the optimal measure value. In particular, the general reducts are proposed to simulate the classical reducts and improve the applicable reducts. Thus, the general reduction system has also been constructed for two-category DTRS. With respect to generalization, the ST and STP-Reduct effectively reflect the mathematical essence of attribute reduction and thus can be generalized. The objective of attribute reduction is to seek accessible and independent knowledge structures to deduce the lowest structural criterion, and this general approach will be used in the next section. Optimal reducts search

the optimal structures for the given measures and thus are more suitable for the DTRS measure environment; this optimal approach based on structure can be generalized as well.

7. Reduct relationships and an example

Up to this point, we have obtained the hierarchical four targets and four reducts, which are C-Preservation, CR-Preservation, ST-Preservation and K-Preservation and CP-Reduct, CRP-Reduct, STP-Reduct and KP-Reduct, respectively. The development line, C-Preservation→CR-Preservation→ST-Preservation→K-Preservation, represents the evolutionary step from a weak reduction target to a strong one. For simplicity, the reduct that is related to a strong reduction target is called *the strong reduct*, and the reduct that is associated with a weak reduction target is called *the weak reduct*. Thus, the corresponding reduct line (from the weak reduct to the strong one) becomes CP-Reduct→CRP-Reduct→ STP-Reduct →KP-Reduct. Moreover, the reduction target relationships have been previously provided. In this section, the four reduct relationships are analyzed, and a D-Table is specifically provided for illustration.

Proposition 7.1. *The core of a strong reduct includes the core of a weak reduct.*

Proposition 7.2. *If B_{strong} is an arbitrary strong reduct, then $\exists B_{weak} \subseteq B_{strong}$, where B_{weak} is a weak reduct. In contrast, if B_{weak} is an arbitrary weak reduct, then there is no necessary conclusion – B_{strong} is a strong reduct and $B_{strong} \supseteq B_{weak}$.*

Propositions 7.1 and 7.2 are easy to prove, and the next example will provide some explanation in this regard. Proposition 7.1 indicates that the core of a weak reduct is necessarily in the core of a strong reduct. Proposition 7.2 indicates that a weak reduct necessarily exists in each strong reduct but could still exist beyond the embedded relationship with respect to all of the strong reducts. Thus, the strong reducts can provide some guidance for the weak reducts. KP-Reduct is the strongest reduct and therefore has some specific functions for all of the reducts, which has been previously noted in Section 3.

The four types of reduction targets have a clear hierarchy. However, the four types of reducts cannot obtain some necessary relationships due to the reduct maximality/independency requirement. Thus, reducts mainly rely on the accessible

Table 2
A statistical decision table that is based on equivalence classes of condition attributes.

$[x]_i$	$\lvert[x]_i\rvert$	c_1	c_2	c_3	c_4	c_5	$\lvert[x]_i \cap X\rvert$
$[x]_1$	5	1	1	1	1	1	0
$[x]_2$	3	1	1	1	2	1	0
$[x]_3$	6	1	2	1	1	3	1
$[x]_4$	5	1	2	1	1	2	1
$[x]_5$	6	2	1	2	2	1	3
$[x]_6$	4	2	3	1	2	2	3
$[x]_7$	6	2	3	3	1	3	5
$[x]_8$	4	3	1	3	2	2	4
$[x]_9$	2	3	2	2	2	1	2

Table 3
Four types of reducts and their cores.

Type	Core	Reduct	Reduct number
CP-Reduct	ϕ	c_1c_2, c_1c_5, c_2c_3	3
CRP-Reduct	ϕ	$c_1c_5, c_2c_3, c_1c_2c_4$	3
STP-Reduct	$\{c_4\}$	$c_1c_2c_4, c_1c_4c_5, c_2c_3c_4$	3
KP-Reduct	$\{c_4, c_5\}$	$c_1c_4c_5, c_2c_3c_4c_5$	2

Table 4
Four types of reducts and their knowledge structures.

Reduct	Knowledge structure	CP-Reduct	CRP-Reduct	STP-Reduct	KP-Reduct
c_1c_2	$\{[x]_1 \cup [x]_2, [x]_3 \cup [x]_4, [x]_5, [x]_6 \cup [x]_7, [x]_8 \cup [x]_9\}$	✓			
c_1c_5	$\{[x]_1 \cup [x]_2, [x]_3, \ldots, [x]_9\}$	✓	✓		
c_2c_3	$\{[x]_1 \cup [x]_2, [x]_3 \cup [x]_4, [x]_5, \ldots, [x]_9\}$	✓	✓		
$c_1c_2c_4$	$\{[x]_1, [x]_2, [x]_3 \cup [x]_4, [x]_5, \ldots, [x]_9\}$		✓	✓	
$c_1c_4c_5$	$\{[x]_1, [x]_2, \ldots, [x]_9\}$			✓	✓
$c_2c_3c_4$	$\{[x]_1, [x]_2, [x]_3 \cup [x]_4, [x]_5, \ldots, [x]_9\}$			✓	
$c_2c_3c_4c_5$	$\{[x]_1, [x]_2, \ldots, [x]_9\}$				✓

substructure of the condition attribute C. Of course, if the maximality/independency condition is weakened, then the hierarchy can provide more reliable conclusions. In particular, Fig. 1 illustrates the relevant reduction hierarchy. The qualitative Pawlak reduct acts as the basis, and its equivalent C-Preservation and R-Preservation are separated into quantitative DRTS-Reduct. Thus, CP-Reduct is based on C-Preservation, whereas CRP-Reduct is based on the dual preservation, and STP-Reduct and KP-Reduct are further related to strong development with high accuracy. Note that DTRS CP-Reduct is also consistent with the Pawlak reduct.

Example 5. In D-Table $S = (U, C \cup D), U = \{x_1, x_2, \ldots, x_{41}\}, C = \{c_1, c_2, c_3, c_4, c_5\}, U/IND(D) = \{X, \neg X\}$. According to the initial D-Table, Table 2 provides statistical information that is based on the equivalence classes of C; moreover, Table 1 is actually a part of Table 2. Here, we utilize the structural approach (including the generalized structure target) for the description and computation.

At first, $U/IND(C) = \{[x]_1, \ldots, [x]_9\}$ and the complete lattice $(ST_C(U), \Leftarrow)$ underlie the entire reduction discussion. For the four types of reduction targets, only one basic computational process is required. (1) C-Preservation concerns only C and D; (2) CR-Preservation also concerns the model, and we assume that $\alpha = 0.8, \beta = 0.2$; (3) ST-Preservation further involves the given ST, and we assume that $st = \{[x]_1, [x]_2, [x]_3 \cup [x]_4, [x]_5 \cup [x]_6, [x]_7, [x]_8 \cup [x]_9\}$; (4) K-Preservation concerns only C. By calculation, $\mathrm{NEG}_C(X) = [x]_1 \cup [x]_2, \mathrm{BND}_C(X) = [x]_3 \cup \ldots \cup [x]_7, \mathrm{POS}_C(X) = [x]_8 \cup [x]_9; \mathrm{NEG}_C^{\alpha,\beta}(X) = [x]_1 \cup \ldots \cup [x]_4, \mathrm{BND}_C^{\alpha,\beta}(X) = [x]_5 \cup [x]_6$, $\mathrm{POS}_C^{\alpha,\beta}(X) = [x]_7 \cup [x]_8 \cup [x]_9$. Thus, the four reduction targets can be represented by the four target infima (i.e., the lowest structure requirements): (1) $inf(C) = \{[x]_1 \cup [x]_2, [x]_3 \cup \ldots \cup [x]_7, [x]_8 \cup [x]_9\}$, (2) $inf(CR) = \{[x]_1 \cup [x]_2, [x]_3 \cup [x]_4, [x]_5 \cup [x]_6, [x]_7, [x]_8 \cup [x]_9\}$, (3) $inf(ST) = \{[x]_1, [x]_2, [x]_3 \cup [x]_4, [x]_5 \cup [x]_6, [x]_7, [x]_8 \cup [x]_9\}$, (4) $inf(K) = U/IND(C) = \{[x]_1, \ldots, [x]_9\}$.

Based on the four target infima, the four types of reducts can be obtained by examining the deducibility and maximality, and this new approach is equivalent to classical processing. Next, the hierarchical results are presented in order from the weakest reduct to the strongest one. (1) For C-Preservation, the core is empty, and there are only three CP-Reducts: c_1c_2, c_1c_5, c_2c_3. (2) For CR-Preservation, the core is also empty, and there are only three CRP-Reducts: $c_1c_5, c_2c_3, c_1c_2c_4$. (3) For ST-Preservation, the core is $\{c_4\}$, and there are only three STP-Reducts: $c_1c_2c_4, c_1c_4c_5, c_2c_3c_4$. (4) For K-Preservation, the core is $\{c_4, c_5\}$, and there are only two KP-Reducts: $c_1c_4c_5, c_2c_3c_4c_5$. Table 3 vividly summarizes the relevant cores and reducts.

There are five condition attributes; therefore, $2^5 - 1 = 31$ non-empty subsets exist in theory. The four above-described types of reducts involve eleven reducts in total, but they concern only seven proper subsets. Table 4 provides the seven subsets and their knowledge structures and reduct distributions, where ✔ marks the matched reduct. Thus, the reduct deducibility for the reduction target infima is easily verified.

According to the comprehensive results presented in Table 3, the four cores are $\phi, \phi, \{c_4\}$ and $\{c_4, c_5\}$, which confirms Proposition 7.1. There are only two KP-Reducts. KP-Reduct $c_1c_4c_5$ is also STP-Reduct and further includes CRP-Reduct and CP-Reduct c_1c_5; KP-Reduct $c_2c_3c_4c_5$ includes STP-Reduct $c_2c_3c_4$, and the latter further includes CRP-Reduct and CP-Reduct c_2c_3. For the opposite direction, only one specific case exists, i.e., no KP-Reduct exhibits the inclusiveness for STP-Reduct $c_1c_2c_4$. Thus, Proposition 7.2 is also confirmed. Moreover, KP-Reducts $c_1c_4c_5, c_2c_3c_4c_5$ can provide some guidance in determining other reducts, especially when seeking only one weak reduct.

Here, we choose a structure target: $st = \{[x]_1, [x]_2, [x]_3 \cup [x]_4, [x]_5 \cup [x]_6, [x]_7, [x]_8 \cup [x]_9\}$. In fact, CR-Preservation corresponds to the infimum $inf(CR) = \{[x]_1 \cup [x]_2, [x]_3 \cup [x]_4, [x]_5 \cup [x]_6, [x]_7, [x]_8 \cup [x]_9\}$. Thus, $(ST^C, \Leftarrow)$ has only $2^4 = 16$ structure targets based on the premise of CR-Preservation, including the used $inf(CR), st, inf(K) = U/IND(C)$; moreover, $inf(CR) \Leftarrow st \Leftarrow U/IND(C)$. Of course, not all structure targets can be realized by the knowledge structure of some condition attributes.

Next, this example is utilized to illustrate the two algorithms proposed for CRP-Reduct in Section 5.2. (1) For the CRP-Reduct-based algorithm, suppose that Step 1 yields the usual CP-Reduct result $Red_{sub}(C) = \{c_1c_2, c_1c_5, c_2c_3\}$; Step 2 yields the empty core, which means that $Red_{*}^{\alpha,\beta}(C) = Red_{sub}(C)$ in the core condition; Step 3 checks the R-Preservation condition, and c_1c_2 is removed, whereas $Red_{**}^{\alpha,\beta}(C) = \{c_1c_5, c_2c_3\}$. Step 4 checks the independency/maximality and obtains the output result $Red_{sub}^{\alpha,\beta}(C) = \{c_1c_5, c_2c_3\}$. Thus, this algorithm provides a CRP-Reduct subset because the complete CRP-Reduct set is $Red^{\alpha,\beta}(C) = \{c_1c_5, c_2c_3, c_1c_2c_4\}$. (2) For the KP-Reduct-based algorithm, Step 2 also yields the empty core. Thus, if Step 1 provides KP-Reduct $B_k = c_1c_4c_5$, then Step 3 will yield CRP-Reduct $B = c_1c_5$; if Step 1 provides KP-Reduct $B_k = c_2c_3c_4c_5$, then Step 3 will yield CRP-Reduct $B = c_2c_3$. Hence, this algorithm provides a concrete CRP-Reduct. This example illustrates the effectiveness of both algorithms. Moreover, note that Proposition 7.2 actually underlies the effectiveness of the KP-Reduct-based algorithm.

In summary, Example 5 clearly shows that CP-Reduct is a D-Table Pawlak reduct, whereas KP-Reduct is a KRS Pawlak reduct. This example and its sub-examples (i.e., Examples 2 and 3) have demonstrated the rationality of both C-Preservation and R-Preservation; thus, CR-Preservation, CP-Reduct and CRP-Reduct are all scientific; moreover, CRP-Reduct is shown to be a novel concept that extends the Pawlak reduct and is mainly applied in quantitative models. Example 5 also demonstrates the generalized structure approach, and STP-Reduct is obtained, which can be viewed as more detailed than CP-Reduct. In particular, Example 5 demonstrates the hierarchy for the four types of targets and reducts, where $inf(K) \Rightarrow inf(ST) \Rightarrow inf(CR) \Rightarrow inf(C)$ actually acts as the hierarchy source. Therefore, Example 5 illustrates well the effectiveness of our whole approach and the relevant results obtained this paper.

Finally, we summarize the structural reduction approach following this vivid example. The reduction target is essentially a family of structure targets that has an infimum, i.e., it has certain structural requirements with the smallest criterion. Thus, attribute reduction considers only the deducibility of the sub-attributes knowledge structures for the reduction target structure. Thus, transforming the reduction target into the structural requirement and focusing on the deducibility of the knowledge structures become a structural reduction characteristic, where the complete lattice acts as the basis and, at the same time, the repeated computation on the decision attributes is avoided. Given a reduction target, attribute reduction mainly removes redundant structures of conditional knowledge and thus realizes optimization and generalization; thus, this structural approach can best reflect the essence of attribute reduction.

8. Conclusions

Attribute reduction plays a central role in RS theory. According to the attribute reduction problem from PRS quantitative extension, this paper concentrates on the hierarchical attribute reduction for two-category DTRS. By promotion and hierarchical analysis, we investigate four targets and reducts. For the reduction targets, K-Preservation implies ST-Preservation, whereas ST-Preservation implies CR-Preservation and CR-Preservation implies C-Preservation. K-Preservation is the highest target, whereas KP-Reduct promotes KRS Pawlak reduct and provides auxiliary information for all of the reducts. Both C-Preservation and R-Preservation are rational and necessary; thus, based on separability, CR-Preservation defines the basic structural premise for two-category DTRS reduction. Moreover, CP-Reduct promotes D-Table Pawlak reduct, whereas CRP-Reduct extends the qualitative Pawlak reduct to the quantitative realm. ST and general reducts (including STP-Reduct and Optimal Reduct) describe the mathematical essence of attribute reduction and provide ample space for application from a structural perspective. In particular, all of the content is effectively explained by the final D-Table example and can be naturally generalized to multiple-category or other quantitative models. Thus, this study demonstrates the promotion, rationality, structure, hierarchy and generalization of attribute reduction and establishes a fundamental reduction framework for two-category DTRS; furthermore, it provides new insights into the problems of Pawlak-Reduction, DTRS-Reduction, and PRS-Reduction. The approaches and results presented are worthy of further verification in practical applications, including experiments on a large database; thus, the structure and hierarchy of attribute reduction must be explored in greater depth based on granular computing, and efforts to generalize the results to multi-category and quantitative models must be undertaken.

Acknowledgements

The authors thank both the editors and the anonymous referees for their valuable suggestions, which substantially improved this paper.

This work was supported by the National Science Foundation of China (61203285 and 61273304), China Postdoctoral Science Foundation Funded Project (2013T60464 and 2012M520930), Shanghai Postdoctoral Scientific Program (13R21416300), Specialized Research Fund for the Doctoral Program of Higher Education of China (20130072130004), and Key Project of Sichuan Provincial Education Department of China (12ZA138).

References

[1] N. Azam, J.T. Yao, Analyzing uncertainties of probabilistic rough set regions with game-theoretic rough sets, Int. J. Approx. Reason., http://dx.doi.org/10.1016/j.ijar.2013.03.015.
[2] M. Beynon, Reducts within the variable precision rough sets model: a further investigation, Eur. J. Oper. Res. 134 (2001) 592–605.
[3] T.F. Fan, D.R. Liu, G.H. Tzeng, Rough set-based logics for multicriteria decision analysis, Eur. J. Oper. Res. 182 (1) (2007) 340–355.
[4] S. Greco, B. Matarazzo, R. Słowiński, Parameterized rough set model using rough membership and bayesian confirmation measures, Int. J. Approx. Reason. 49 (2) (2008) 285–300.
[5] J.P. Herbert, J.T. Yao, Game-theoretic rough sets, Fundam. Inform. 108 (2011) 267–286.
[6] B. Huang, H.X. Li, D.K. Wei, Dominance-based rough set model in intuitionistic fuzzy information systems, Knowl.-Based Syst. 28 (2012) 115–123.
[7] M. Inuiguchi, Y. Yoshioka, Y. Kusunoki, Variable-precision dominance-based rough set approach and attribute reduction, Int. J. Approx. Reason. 50 (8) (2009) 1199–1214.
[8] X.Y. Jia, W.H. Liao, Z.M. Tang, L. Shang, Minimum cost attribute reduction in decision-theoretic rough set models, Inform. Sci. 219 (10) (2013) 151–167.
[9] X.Y. Jia, Z.M. Tang, W.H. Liao, L. Shang, On an optimization representation of decision-theoretic rough set model, Int. J. Approx. Reason., http://dx.doi.org/10.1016/j.ijar.2013.02.010.
[10] Y.C. Jiang, J. Wang, S.Q. Tang, B. Xiao, Reasoning with rough description logics: an approximate concepts approach, Inform. Sci. 179 (5) (2009) 600–612.
[11] X.P. Kang, D.Y. Li, S.G. Wang, K.S. Qu, Rough set model based on formal concept analysis, Inform. Sci. 222 (2013) 611–625.
[12] F. Li, M. Ye, X.D. Chen, An extension to rough c-means clustering based on decision-theoretic rough set model, Int. J. Approx. Reason., http://dx.doi.org/10.1016/j.ijar.2013.05.005.
[13] H.X. Li, X.Z. Zhou, J.B. Zhao, D. Liu, Attribute reduction in decision-theoretic rough set model: a further investigation, Lecture Notes in Computer Science 6954 (2011) 466–475.
[14] M. Li, C.X. Shang, S.Z. Feng, J.P. Fan, Quick attribute reduction in inconsistent decision tables, Inform. Sci., http://dx.doi.org/10.1016/j.ins.2013.08.038.
[15] D.C. Liang, D. Liu, W. Pedrycz, P. Hu, Triangular fuzzy decision-theoretic rough sets, Int. J. Approx. Reason. 54 (8) (2013) 1087–1106.
[16] J.Y. Liang, J.R. Mi, W. Wei, F. Wang, An accelerator for attribute reduction based on perspective of objects and attributes, Knowl.-Based Syst. 44 (2013) 90–100.
[17] T.Y. Lin, Q. Liu, First-order rough logic I: approximate reasoning via rough sets, Fundam. Inform. 27 (2) (1996) 137–153.
[18] P. Lingras, M. Chen, D.Q. Miao, Rough cluster quality index based on decision theory, IEEE Trans. Knowl. Data Eng. 21 (7) (2009) 1014–1026.
[19] P. Lingras, M. Chen, D.Q. Miao, Rough multi-category decision theoretic framework, Lecture Notes in Computer Science 5009 (2008) 676–683.
[20] P. Lingras, M. Chen, D.Q. Miao, Semi-supervised rough cost/benefit decisions, Fundam. Inform. 94 (2) (2009) 233–244.

[21] C.H. Liu, D.Q. Miao, N. Zhang, C. Gao, Graded rough set model based on two universes and its properties, Knowl.-Based Syst. 33 (2012) 65–72.
[22] D. Liu, T.R. Li, H.X. Li, A multiple-category classification approach with decision-theoretic rough sets, Fundam. Inform. 115 (2–3) (2012) 173–188.
[23] D. Liu, T.R. Li, D.C. Liang, Incorporating logistic regression to decision-theoretic rough sets for classifications, Int. J. Approx. Reason., http://dx.doi.org/10.1016/j.ijar.2013.02.013.
[24] D. Liu, T.R. Li, D. Ruan, Probabilistic model criteria with decision-theoretic rough sets, Inform. Sci. 181 (2011) 3709–3722.
[25] J.S. Mi, W.Z. Wu, W.X. Zhang, Approaches to knowledge reduction based on variable precision rough set model, Inform. Sci. 159 (3–4) (2004) 255–272.
[26] D.Q. Miao, Y. Zhao, Y.Y. Yao, H.X. Li, F.F. Xu, Relative reducts in consistent and inconsistent decision tables of the Pawlak rough set model, Inform. Sci. 179 (2009) 4140–4150.
[27] A. Nakamura, A rough logic based on incomplete information and its application, Int. J. Approx. Reason. 15 (4) (1996) 367–378.
[28] Z. Pawlak, Rough sets, Int. J. Inform. Comput. Sci. 11 (5) (1982) 341–356.
[29] Z. Pawlak, Rough Sets: Theoretical Aspects of Reasoning about Data, System Theory, Knowledge Engineering and Problem Solving, vol. 9, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1991.
[30] Z. Pawlak, A. Skowron, Rough membership functions, in: Advances in the Dempster–Shafer Theory of Evidence, John Wiley and Sons, New York, 1994, pp. 251–271.
[31] Z. Pawlak, A. Skowron, Rudiments of rough sets, Inform. Sci. 177 (2007) 3–27.
[32] W. Pedrycz, Allocation of information granularity in optimization and decision-making models: towards building the foundations of granular computing, Eur. J. Oper. Res. 232 (1) (2014) 137–145.
[33] W. Pedrycz, Granular Computing: Analysis and Design of Intelligent Systems, CRC Press, Francis Taylor, Boca Raton, 2013.
[34] Y.H. Qian, H. Zhang, Y.L. Sang, J.Y. Liang, Multigranulation decision-theoretic rough sets, Int. J. Approx. Reason., http://dx.doi.org/10.1016/j.ijar.2013.03.004.
[35] K.Y. Qin, Z. Pei, J.L. Yang, Y. Xu, Approximation operators on complete completely distributive lattices, Inform. Sci. 247 (2013) 123–130.
[36] Y.H. She, X.L. He, Uncertainty measures for rough formulae in rough logic: an axiomatic approach, Comput. Math. Appl. 63 (1) (2012) 83–93.
[37] Y.H. She, L.N. Ma, On the rough consistency measures of logic theories and approximate reasoning in rough logic, Int. J. Approx. Reason., http://dx.doi.org/10.1016/j.ijar.2013.10.001.
[38] A. Skowron, J. Stepaniuk, R. Swiniarski, Modeling rough granular computing based on approximation spaces, Inform. Sci. 184 (1) (2012) 20–43.
[39] D. Slezak, W. Ziarko, Bayesian rough set model, in: Proc. of the International Workshop on Foundation of Data Mining (FDM2002), Maebashi, Japan, 2002, pp. 131–135.
[40] D. Slezak, W. Ziarko, The investigation of the bayesian rough set model, Int. J. Approx. Reason. 40 (1–2) (2005) 81–91.
[41] C.Z. Wang, Q. He, D.G. Chen, Q.H. Hu, A novel method for attribute reduction of covering decision systems, Inform. Sci., http://dx.doi.org/10.1016/j.ins.2013.08.057.
[42] F. Wang, J.Y. Liang, C.Y. Dang, Attribute reduction for dynamic data sets, Appl. Soft Comput. 13 (1) (2013) 676–689.
[43] J.Y. Wang, J. Zhou, Research of reduct features in the variable precision rough set model, Neurocomputing 72 (2009) 2643–2648.
[44] S.K.M. Wong, W. Ziarko, Comparison of the probabilistic approximate classification and the fuzzy set model, Fuzzy Sets Syst. 21 (1987) 357–362.
[45] W.H. Xu, J.Z. Pang, S.Q. Luo, A novel cognitive system model and approach to transformation of information granules, Int. J. Approx. Reason., http://dx.doi.org/10.1016/j.ijar.2013.10.002.
[46] Y.Y. Yao, The superiority of three-way decision in probabilistic rough set models, Inform. Sci. 181 (2011) 1080–1096.
[47] Y.Y. Yao, Three-way decisions with probabilistic rough sets, Inform. Sci. 180 (2010) 341–353.
[48] Y.Y. Yao, T.Y. Lin, Generalization of rough sets using modal logics, Intell. Autom. Soft Comput. 2 (2) (1996) 103–120.
[49] Y.Y. Yao, T.Y. Lin, Graded rough set approximations based on nested neighborhood systems, in: H.J. Zimmermann (Ed.), Proceedings of 5th European Congress on Intelligent Techniques and Soft Computing, EUFIT'97, vol. 1, Verlag Mainz, Aachen, 1997, pp. 196–200.
[50] Y.Y. Yao, S.K.M. Wong, P. Lingras, A decision-theoretic rough set model, in: Z.W. Ras, M. Zemankova, M.L. Emrich (Eds.), The 5th International Symposium on Methodologies for Intelligent Systems, North-Holland, New York, 1990, pp. 17–25.
[51] Y.Y. Yao, Y. Zhao, Attribute reduction in decision-theoretic rough set models, Inform. Sci. 178 (17) (2008) 3356–3373.
[52] D.Y. Ye, Z.J. Chen, S.L. Ma, A novel and better fitness evaluation for rough set based minimum attribute reduction problem, Inform. Sci. 222 (2013) 413–423.
[53] H. Yu, Z.G. Liu, G.Y. Wang, An automatic method to determine the number of clusters using decision-theoretic rough set, Int. J. Approx. Reason., http://dx.doi.org/10.1016/j.ijar.2013.03.018.
[54] H.Y. Zhang, J. Zhou, D.Q. Miao, C. Gao, Bayesian rough set model: a further investigation, Int. J. Approx. Reason. 53 (2012) 541–557.
[55] J.B. Zhang, J.S. Wong, T.R. Li, Y. Pan, A comparison of parallel large-scale knowledge acquisition using rough set theory on different MapReduce runtime systems, Int. J. Approx. Reason., http://dx.doi.org/10.1016/j.ijar.2013.08.003.
[56] X.H. Zhang, B. Zhou, P. Li, A general frame for intuitionistic fuzzy rough sets, Inform. Sci. 216 (2012) 34–49.
[57] X.Y. Zhang, D.Q. Miao, Quantitative information architecture, granular computing and rough set models in the double-quantitative approximation space on precision and grade, Inform. Sci. 268 (2014) 147–168.
[58] X.Y. Zhang, D.Q. Miao, Two basic double-quantitative rough set models of precision and grade and their investigation using granular computing, Int. J. Approx. Reason. 54 (8) (2013) 1130–1148.
[59] X.Y. Zhang, Z.W. Mo, F. Xiong, W. Cheng, Comparative study of variable precision rough set model and graded rough set model, Int. J. Approx. Reason. 53 (1) (2012) 104–116.
[60] S.Y. Zhao, X.Z. Wang, D.G. Chen, E.C.C. Tsang, Nested structure in parameterized rough reduction, Inform. Sci. 248 (2013) 130–150.
[61] Y. Zhao, S.K.M. Wong, Y.Y. Yao, A note on attribute reduction in the decision-theoretic rough set model, LNCS Transactions on Rough Sets XIII 6499 (2011) 260–275.
[62] B. Zhou, Multi-class decision-theoretic rough sets, Int. J. Approx. Reason., http://dx.doi.org/10.1016/j.ijar.2013.04.006.
[63] J. Zhou, D.Q. Miao, β-Interval attribute reduction in variable precision rough set model, Soft Comput. 15 (8) (2011) 1643–1656.
[64] W. Ziarko, Variable precision rough set model, J. Comput. Syst. Sci. 46 (1993) 39–59.

أهداف المشروع

تسلسل	الهدف

طريقة تحقيق أهداف المشروع

مسلسل	الهدف	طريقة تحقيق الهدف

خريطة أهداف - مراحل- مهام المشروع

الهدف	المرحلة	المهمة

أدوار الفريق البحثي ومدة التنفيذ لكل عضو

أعضاء الفريق	الدور	مدة التنفيذ
احمد ابراهيم عبده شرف	باحث رئيسي	0 غير مخصص

خطة العمل - الجدول الزمني

المراحل والمهام	الفريق البحثى	مدة المشروع											
		1	2	3	4	5	6	7	8	9	10	11	12

خريطة مخرجات وأهداف المشروع البحثى والأهداف الاستراتيجية للبرنامج

مخرجات المشروع المتوقعة	أهداف المشروع المطلوب تحقيقها	الأهداف الإستراتيجية للبرنامج

الأهداف الإستراتيجية للبرنامج

مسلسل	الهدف الإستراتيجي

البند	القيمة بالريال السعودي
الموارد البشرية	
الموارد البشرية	0.00
الاجمالي	0.00
materials	
equipment	0.00
basic supplies	0.00
professional works	0.00
materials	0.00
الاجمالي	0.00
Travel	
conferences	0.00
training	0.00
field trips	0.00
الاجمالي	0.00
مصروفات أخري	
publishing cost	0.00
Writhing Help	0.00
scientific editing	0.00
الاجمالي	0.00
الاجمالي	0.00

Attribute reduction for dynamic data sets

Feng Wang[a], Jiye Liang[a,*], Chuangyin Dang[b], Yuhua Qian[a]

[a]*Key Laboratory of Computational Intelligence and Chinese Information Processing of Ministry of Education, School of Computer and Information Technology, Shanxi University, Taiyuan, 030006, Shanxi, China,*
[b]*Department of System Engineering and Engineering Management, City University of Hong Kong, Hong Kong*

Abstract

Many real data sets in databases may vary dynamically. With such data sets, one has to run a knowledge acquisition algorithm repeatedly in order to acquire new knowledge. This is a very time-consuming process. To overcome this deficiency, several approaches have been developed to deal with dynamic databases. They mainly address knowledge updating from three aspects: the expansion of data, the increasing number of attributes and the variation of data values. This paper focuses on attribute reduction for data sets with dynamically varying data values. Information entropy is a common measure of uncertainty and has been widely used to construct attribute reduction algorithms. Based on three representative entropies, this paper develops an attribute reduction algorithm for data sets with dynamically varying data values. When a part of data in a given data set is replaced by some new data, compared with the classic reduction algorithms based on the three entropies, the developed algorithm can find a new reduct in a much shorter time. Experiments on six data sets downloaded from UCI show that the algorithm is effective and efficient.

Keywords: Dynamic data sets, Rough sets, Information entropy, Attribute reduction

1. Introduction

In real world databases, data sets usually vary with time. This phenomenon occurs in many fields such as economic research, social survey and medical research. As data sets change with time, especially at an unprecedented rate, it is very time-consuming or even infeasible to run repeatedly a knowledge acquisition algorithm. To overcome this deficiency, researchers have recently proposed many new analytic techniques. They usually can directly carry out the computation using the existing result from the original data set. These techniques mainly address knowledge updating from three aspects: the expansion of data [1-7], the increasing number of attributes [8-11] and the variation of data values [12-13]. For the first two aspects, a number of incremental techniques have been developed to acquire new knowledge without recomputation. However, little research has been done on the third aspect in knowledge acquisition, which motivates this study. This paper concerns attribute reduction for data sets with dynamically varying data values. For convenience of the following discussion, here are some specific explanations regarding data sets with dynamically varying data values. Generally speaking, this case usually occurs in the following several situations. One situation is that a part of the original data in a database is identified as wrong, thus needing to be corrected. Wrong data obviously lose their value to store further, and will be directly replaced by correct ones. Another familiar situation is that, initially collected data in a database may increase gradually, though it usually is not necessary to acquire knowledge from total data all the time. In other words, the sizes of interested data sets do not change. For example, in a pollution survey of X city, observed data in last few years or even decades may be stored in a database totally. However, analysis of pollution in each week (or each day, each month, etc.) does not require total observed data in the past. In this situation, because data sets of adjacent time intervals are usually similar to each other, an interested data set at one moment can be slightly amended to obtain a data set for the next moment. In addition, with

*Corresponding author. Tel./Fax: +86 0351 7018176.
Email addresses: sxuwangfeng@126.com (Feng Wang), ljy@sxu.edu.cn (Jiye Liang), mecdang@cityu.edu.hk (Chuangyin Dang), jinchengqyh@126.com (Yuhua Qian)

the rapid development of information technology, timeliness of data becomes more and more important. Thus, any out-of-date data in databases are usually useless. To improve the efficiency of knowledge acquisition, useless data should be directly updated by the latest or real-time ones. Furthermore, other similar situations occur rather often in applications such as stock analysis, tests for the disease and annual appraisal of workers.

Feature selection, a common technique for data preprocessing in many areas including pattern recognition, machine learning and data mining, has hold great significance. Among various approaches to select useful features, a special theoretical framework is Pawlak's rough set model [14,15]. One can use rough set theory to select a subset of features that is most suitable for a given recognition problem [16-21]. Rough feature selection is also called attribute reduction, which aims to select those features that keep the discerniblity ability of the original ones[22-26]. The feature subset generated by an attribute reduction algorithm is called a reduct. In the last two decades, researchers have proposed many reduction algorithms [27-32]. However, most of these algorithms can only be applicable to static data sets. Although several algorithms have been proposed for dynamic data sets [1-11], as mentioned above, they are incremental approaches only for the dynamic expansion of data or attributes.

This paper focus on attribute reduction for dynamically varying data values. To tackle this problem, this paper will exploit information entropy. The information entropy from classical thermodynamics is used to measure out-of-order degree of a system. Information entropy is introduced in rough set theory to measure uncertainty of a given data set [33-36], which have been widely applied to devise heuristic attribute reduction algorithms [37-41]. Complementary entropy [33], combination entropy [35] and Shannon's entropy [36] are three representative entropies which have been mainly used to construct reduction algorithms in rough set theory. To fully explore properties in reduct updating caused by the variation of data values, this paper develops an attribute reduction algorithm for dynamic data sets based on the three entropies. In view of that a key step of the development is the computation of entropy, this paper first introduces three updating mechanisms of the three entropies, which determine an entropy by changing one object to a new one in a decision table. When only one object is changed, instead of recomputation on the given decision table, the updating mechanisms derive new entropies by integrating the changes of conditional classes and decision classes into the existing entropies. With these mechanisms, an attribute reduction algorithm is proposed for dynamic decision tables. When a part of data in a given data set is replaced by some new data, compared with the classic reduction algorithms based on the three entropies, the developed algorithm can find a new reduct in a much shorter time. Experiments on six data sets downloaded from UCI show that the algorithm is effective and efficient.

The rest of this paper is organized as follows. Some preliminaries in rough set theory are briefly reviewed in Section 2. Traditional heuristic reduction algorithms based on three representative entropies are introduced in Section 3. Section 4 presents the updating mechanisms of the three entropies for dynamically varying data values. In Section 5, based on the updating mechanisms, a reduction algorithm is proposed to compute reducts for dynamic data sets. In Section 6, six UCI data sets are employed to demonstrate effectiveness and efficiency of the proposed algorithm. Section 7 concludes this paper with some discussions.

2. Preliminary knowledge on rough sets

2.1. Basic concepts

This section reviews several basic concepts in rough set theory. Throughout this paper, the universe U is assumed a finite nonempty set.

An information system, as a basic concept in rough set theory, provides a convenient framework for the representation of objects in terms of their attribute values. An information system is a quadruple $S = (U, A, V, f)$, where U is a finite nonempty set of objects and is called the universe and A is a finite nonempty set of attributes, $V = \bigcup_{a\in A} V_a$ with V_a being the domain of a, and $f : U \times A \to V$ is an information function with $f(x, a) \in V_a$ for each $a \in A$ and $x \in U$. The system S can often be simplified as $S = (U, A)$.

Each nonempty subset $B \subseteq A$ determines an indiscernibility relation in the following way,

$$R_B = \{(x, y) \in U \times U \mid f(x, a) = f(y, a), \forall a \in B\}.$$

The relation R_B partitions U into some equivalence classes given by

$$U/R_B = \{[x]_B \mid x \in U\}, \text{ just } U/B,$$

where $[x]_B$ denotes the equivalence class determined by x with respect to B, i.e.,

$$[x]_B = \{y \in U \mid (x, y) \in R_B\}.$$

Given an equivalence relation R on the universe U and a subset $X \subseteq U$, one can define a lower approximation of X and an upper approximation of X by

$$\underline{R}X = \bigcup\{x \in U \mid [x]_R \subseteq X\}$$

and

$$\overline{R}X = \bigcup\{x \in U \mid [x]_R \cap X \neq \emptyset\},$$

respectively [39]. The order pair $(\underline{R}X, \overline{R}X)$ is called a rough set of X with respect to R. The positive region of X is denoted by $POS_R(X) = \underline{R}X$.

A partial relation $\preceq$ on the family $\{U/B \mid B \subseteq A\}$ is defined as follows [37]: $U/P \preceq U/Q$ (or $U/Q \succeq U/P$) if and only if, for every $P_i \in U/P$, there exists $Q_j \in U/Q$ such that $P_i \subseteq Q_j$, where $U/P = \{P_1, P_2, \cdots, P_m\}$ and $U/Q = \{Q_1, Q_2, \cdots, Q_n\}$ are partitions induced by $P,\ Q \subseteq A$, respectively. In this case, we say that Q is coarser than P, or P is finer than Q. If $U/P \preceq U/Q$ and $U/P \neq U/Q$, we say Q is strictly coarser than P (or P is strictly finer than Q), denoted by $U/P \prec U/Q$ (or $U/Q \succ U/P$).

It is clear that $U/P \prec U/Q$ if and only if, for every $X \in U/P$, there exists $Y \in U/Q$ such that $X \subseteq Y$, and there exist $X_0 \in U/P$ and $Y_0 \in U/Q$ such that $X_0 \subset Y_0$.

A decision table is an information system $S = (U, C \cup D)$ with $C \cap D = \emptyset$, where an element of C is called a condition attribute, C is called a condition attribute set, an element of D is called a decision attribute, and D is called a decision attribute set. Given $P \subseteq C$ and $U/D = \{D_1, D_2, \cdots, D_r\}$, the positive region of D with respect to the condition attribute set P is defined by $POS_P(D) = \bigcup_{k=1}^{r} \underline{P}D_k$.

For a decision table S and $P \subseteq C$, $X \in U/P$ is consistent iff all its objects have the same decision value; otherwise, X is inconsistent. The decision table S is called a consistent decision table iff $\forall X \in U/C$ are consistent; and if $\exists x, y \in U$ are inconsistent, then the table is called an inconsistent decision table. One can extract certain decision rules from a consistent decision table and uncertain decision rules from an inconsistent decision table.

For a decision table S and $P \subseteq C$, when a new object x is added to S, x is indistinguishable on B iff, $\exists y \in U$, $\forall a \in P$, such that $f(x, a) = f(y, a)$; and x is distinguishable on P iff, $\forall y \in U$, $\exists a \in P$ such that $f(x, a) \neq f(y, a)$.

2.2. Attribute reduction in rough set theory

Given an information system, all the attributes are not necessarily in the same importance, and some of them are irrelevant to the learning or recognition tasks. The concept of attribute reduction was first originated by Pawlak in [14, 15], which aimed to delete the irrelevant or redundant attributes on the condition of retaining the discernible ability of original attributes (the whole attributes set). The retained attribute subset got from attribute reduction is called a reduct.

Definition 1. *Let $S = \{U, A\}$ be an information system. Then $B \subseteq A$ is a reduct of S if*
(1) $U/B = U/A$ and
(2) $\forall a \in B, U/(B - \{a\}) \neq U/B$.

There are usually multiple reducts in a given information system, and the intersection of all reducts is called core. Given a decision table, the retained attribute subset got from attribute reduction is called a relative reduct, and the intersection of all relative reducts is called relative core [14, 15].

Definition 2. *Let $S = \{U, C \cup D\}$ be a decision table. Then $B \subseteq C$ is a relative reduct of S if*
(1) $POS_B(D) = POS_C(D)$ and
(2) $\forall a \in B, POS_{B-\{a\}}(D) \neq POS_B(D)$.

For these two definitions, the first condition guarantees that the reduct has the same distinguish power as the whole attribute set, and the second condition guarantees that there is no redundant attributes in the reduct. A reduct is called an exact reduct if it satisfies both of these two constraints, otherwise, is just an approximate reduct. In [40], Skowron proposed a discernibility matrix method to find all exact reducts of an information system without decision attributes.

However, it has been proved that using this algorithm to generate reducts is an NP-hard problem. For decision tables, Kryszkiewicz proposed an approach to computing the minimal set of attributes that functionally determine a decision attribute[41]. This algorithm can find an exact reduct of a given decision table. These two approaches are both very time-consuming.

As is well known, Pawlak's rough set model is applicable for the case that only nominal attributes exist in data sets. However, many real data in applications usually come with a complicated form. To conceptualize and analyze various types of data, researchers have generalized Pawlak's classic rough set model, and attribute reduction based on these generalizations was also redefined. Reducts generated by these reduction algorithms are usually approximate reducts. Ziarko provided the concept of β-reduct based on the introduction of variable precision rough set model (VPRS)[42]. VPRS deals with partial classification by introducing a probability value β. The β value represents a bound on the conditional probability of objects in a condition class which are classified to the same decision class. Yao proposed the decision-theoretic rough set model and also defined attribute reduction based on this generalized model[43, 44]. This model with loss functions aims to obtain optimization decisions by minimizing the overall risk with Bayesian decision procedures. An extensive review of multi-criteria decision analysis based on dominance rough sets was given by Greco et al. [45]. Dominance rough set model has also been applied for ordinal attribute reduction and multi-criteria classification [46, 47]. Dubois and Prade constructed the first fuzzy rough model by extending equivalence relation to fuzzy equivalence relation [48], where fuzzy equivalence relations satisfy reflexivity, symmetry and max-min transitivity. Reduction algorithms based on above generalized rough set models usually generate one or more approximation reducts and have been applied to solve their corresponding issues. It is deserved to point out that each kind of attribute reduction tries to preserve a particular property of a given table.

In addition, to save computational time of finding reduct, researchers have also developed many heuristic attribute reduction algorithms which can generate a single reduct from a given table [33, 35, 37, 38, 49]. Most of them are greedy and forward search algorithms, keeping selecting attributes with high significance until the dependence no longer increases. The reduct generated by a heuristic reduction algorithm is usually considered as an approximation reduct. It will be an exact reduct when deleting its redundant attributes.

However, the above analysis about generating an exact reduct and an approximation reduct is not very rigorous. For example, though reducts generated by heuristic reduction algorithms are considered as approximation reducts, some of them are often exact redcuts. Because so many reduction algorithms have been proposed in the last two decades and it is very difficult to list all of them here, this section just introduces a common distinction between generating an exact reduct and generating an approximation reduct.

3. Attribute reduction based on information entropy

Among various heuristic attribute reduction algorithms, reduction based on information entropy (or its variants) is a kind of common algorithm which has attracted much attention. There are three representative entropies which are used to construct reduction algorithms. They are complementary entropy[33], combination entropy[35] and Shannons information entropy[36]. The heuristic attribute reduction algorithms based on these three entropies are reviewed in this section.

In [33], the complementary entropy was introduced to measure uncertainty in rough set theory. Liang et al. also proposed the conditional complementary entropy to measure uncertainty of a decision table in [34]. By preserving the conditional entropy unchanged, the conditional complementary entropy was applied to construct reduction algorithms and reduce the redundant features in a decision table [35]. The conditional complementary entropy used in this algorithm is defined as follows[33, 34, 35].

Definition 3. *Let $S = (U, C \cup D)$ be a decision table and $B \subseteq C$. Then, one can obtain the condition partitions $U/B = \{X_1, X_2, \cdots, X_m\}$ and $U/D = \{Y_1, Y_2, \cdots, Y_n\}$. Based on these partitions, a conditional entropy of B relative to D is defined as*

$$E(D|B) = \sum_{i=1}^{m} \sum_{j=1}^{n} \frac{|X_i \cap Y_j|}{|U|} \frac{|Y_j^c - X_i^c|}{|U|}, \tag{1}$$

where Y_i^c and X_j^c are complement set of Y_i and X_j respectively.

Another information entropy, called combination entropy, was presented in [35] to measure the uncertainty of data tables. The conditional combination entropy was also introduced and can be used to construct the heuristic reduction algorithms [35]. This reduction algorithm can find a feature subset that possesses the same number of pairs of indistinguishable elements as that of the original decision table. The definition of the conditional combination entropy is defined as follows[35].

Definition 4. *Let $S = (U, C \cup D)$ be a decision table and $B \subseteq C$. Then one can obtain the condition partitions $U/B = \{X_1, X_2, \cdots, X_m\}$ and $U/D = \{Y_1, Y_2, \cdots, Y_n\}$. Based on these partitions, a conditional entropy of B relative to D is defined as*

$$CE(D|B) = \sum_{i=1}^{m}(\frac{|X_i|}{|U|}\frac{C^2_{|X_i|}}{C^2_{|U|}} - \sum_{j=1}^{n}\frac{|X_i \cap Y_j|}{|U|}\frac{C^2_{|X_i \cap Y_j|}}{C^2_{|U|}}). \tag{2}$$

where $C^2_{|X_i|}$ denotes the number of pairs of objects which are not distinguishable from each other in the equivalence class X_i.

Based on the classical rough set model, Shannon's information entropy[36] and its conditional entropy were also introduced to find a reduct in a heuristic algorithm [38, 50]. In [38], the reduction algorithm keeps the conditional entropy of target decision unchanged, and the conditional entropy is defined as follows[38].

Definition 5. *Let $S = (U, C \cup D)$ be a decision table and $B \subseteq C$. Then, one can obtain the condition partitions $U/B = \{X_1, X_2, \cdots, X_m\}$ and $U/D = \{Y_1, Y_2, \cdots, Y_n\}$. Based on these partitions, a conditional entropy of B relative to D is defined as*

$$H(D|B) = -\sum_{i=1}^{m}\frac{|X_i|}{|U|}\sum_{j=1}^{n}\frac{|X_i \cap Y_j|}{|X_i|}\log(\frac{|X_i \cap Y_j|}{|X_i|}). \tag{3}$$

For convenience, a uniform notation $ME(D|B)$ is introduced to denote these three entropies. For example, if one adopts Shannon's conditional entropy to define the attribute significance, then $ME(D|B) = H(D|B)$. Given a decision table $S = (U, C \cup D)$ and $B_1, B_2 \subseteq C$. According to literatures [33, 35, 38], if $U/B_1 \preceq U/B_2$, one can get that $ME(D|B_1) \leq ME(D|B_2)$. This conclusion indicates that, for a given decision table, as its condition classifications become finer, its entropies (the three entropies) are monotone decreasing. In addition, as the condition classifications become finer, the classified quality (see Definition 11) of the given decision table is monotone increasing. Thus, one can get that the three entropies of a given decision table are monotone decreasing with the classified quality increasing. Especially, when the classified quality is one, the entropies are zero [33, 35, 38].

The attribute significances based on entropies in a heuristic reduction algorithm is defined as follows (See Definitions 6-7) [33, 35, 38].

Definition 6. *Let $S = (U, C \cup D)$ be a decision table and $B \subseteq C$. $\forall a \in B$, the significance measure (inner significance) of a in B is defined as*

$$Sig^{inner}(a, B, D) = ME(D|B - \{a\}) - ME(D|B). \tag{4}$$

Definition 7. *Let $S = (U, C \cup D)$ be a decision table and $B \subseteq C$. $\forall a \in C - B$, the significance measure (outer significance) of a in B is defined as*

$$Sig^{outer}(a, B, D) = ME(D|B) - ME(D|B \cup \{a\}). \tag{5}$$

Given a decision table $S = (U, C \cup D)$ and $a \in C$. From the literatures [26-29], one can get that if $Sig^{inner}(a, C, D) > 0$, then the attribute a is indispensable, i.e., a is a core attribute of S. Based on core attributes, a heuristic attribute reduction algorithm can find a reduct by gradually adding selected attributes to the core. The definition of reduct based on information entropy is defined as follows [26-29].

Definition 8. *Let $S = (U, C \cup D)$ be a decision table and $B \subseteq C$. Then the attribute set B is a relative reduct if B satisfies:*

(1) $ME(D|B) = ME(D|C)$;

(2) $\forall a \in B$, $ME(D|B) \neq ME(D|B - \{a\})$.

Formally, the searching strategies in reduction algorithms based on the three entropies are similar to each other. The specific steps are written as follows [33, 35, 38].

Algorithm 1. Classic attribute reduction algorithm based on information entropy for a decision table (CAR)

Input: A decision table $S = (U, C \cup D)$
Output: Reduct *red*

Step 1: $red \leftarrow \emptyset$;
Step 2: $for\ (j = 1; j \leq |C|; j++)$
{ if $Sig^{inner}(a_j, C, D) > 0$, then $red \leftarrow red \cup \{a_j\}$;
}
Step 4: $P \leftarrow red$, *while* $(ME(D|P) \neq ME(D|C))$ *do*
{
Compute and select sequentially $Sig^{outer}(a_0, P, D) = max\{Sig^{outer}(a_i, P, D)\}, a_i \in C - P$;
$P \leftarrow P \cup \{a_0\}$;
}
Step 5: $red \leftarrow P$, return *red* and end.

Based on Definition 6, one can get core attributes according to steps 1-2 in this algorithm. Steps 3-4 add selected attributes to the core gradually, and then one can obtain a reduct of the given table. This algorithm can be considered as the common attribute reduction algorithm based on information entropy. The time complexity of CAR given in [37] is $O(|U||C|^2)$. However, this time complexity does not include the computational time of entropies. For a given decision table, computing entropies is a key step in above reduction algorithm, which is not computationally costless. Thus, to analyze the exact time complexity of above algorithm, the time complexity of computing entropies is given as well.

Given a decision table, according to Definitions 3-5, it first needs to compute the conditional classes and decision classes, respectively, and then computes the value of entropy. Xu et al. in [51] gave a fast algorithm for partition with time complexity being $O(|U||C|)$. So, the time complexity of computing entropy is

$$O(|U||C| + |U| + \sum_{i=1}^{m} |X_i| \cdot \sum_{j=1}^{n} |Y_j|) = O(|U||C| + |U| + |U||U|) = O(|U|^2),$$

where the specific introduction of m, n, X_i and Y_j is shown in Definitions 3-5. Thus, the time complexity of computing core (steps 1-2) is $O(|C||U|^2)$, and the time complexity of computing reduct according to CAR is

$$O(|C||U|^2 + |C|(|U||C| + |U|^2)) = O(|C|^2|U| + |C||U|^2).$$

4. Updating mechanism of information entropy

Given a dynamic decision table, based on the three representative entropies, this section presents the updating mechanisms of the three entropies for dynamically varying data values. As data values in a decision table vary with time, recomputing entropy is obviously time-consuming. To overcome this deficiency, the updating mechanisms derive new entropies by integrating the changes of conditional classes and decision classes into existing entropies. When data values of a single object vary, Theorems 1-4 introduce the updating mechanisms for the three entropies respectively.

For convenience, here are some explanations which will be used in the following theorems. Let $S = (U, C \cup D)$ be a decision table, $B \subseteq C$ and $x \in U$. $U/B = \{X_1, X_2, \cdots, X_m\}$, $U/D = \{Y_1, Y_2, \cdots, Y_n\}$, $x \in X_{p_1}$ and $x \in Y_{q_1}$

($p_1 \in \{1, 2, \cdots, m\}$ and $q_1 \in \{1, 2, \cdots, n\}$). If attribute values of x are varied, and here assumes that x is changed to x'. Let $U_{x'}$ denotes the new universe, one has $x' \in X'_{p_2}$ and $x' \in Y'_{q_2}$ ($X'_{p_2} \in U_{x'}/B$ and $Y'_{q_2} \in U_{x'}/D$). Obviously, one can get that $X'_{p_2} - \{x\} \in U/B$, $Y'_{q_2} - \{x\} \in U/D$, $X_{p_1} - \{x\} \in U_{x'}/B$ and $Y_{q_1} - \{x\} \in U_{x'}/D$. In addition, X'_{p_1}, Y'_{q_1}, X_{p_2} and Y_{q_2} denote $X_{p_1} - \{x\}$, $Y_{q_1} - \{x\}$, $X'_{p_2} - \{x\}$ and $Y'_{q_2} - \{x\}$, respectively.

Theorem 1. *Let $S = (U, C \cup D)$ be a decision table and $B \subseteq C$. The complementary conditional entropy of D with respect to B is $E_U(D|B)$. Then, one can obtain the partitions $U/B = \{X_1, X_2, \cdots, X_m\}$ and $U/D = \{Y_1, Y_2, \cdots, Y_n\}$. $x \in X_{p_1}$ and $x \in Y_{q_1}$. If one and only object $x \in U$ is changed to x', then $x' \in X'_{p_2}$ and $x' \in Y'_{q_2}$ ($X'_{p_2} \in U_{x'}/B$ and $Y'_{q_2} \in U_{x'}/D$). The new conditional complementary entropy becomes*

$$E_{U_{x'}}(D|B) = E_U(D|B) + \frac{2|X'_{p_2} - Y'_{q_2}| - 2|X'_{p_1} - Y'_{q_1}|}{|U|^2},$$

where $X'_{p_1} = X_{p_1} - \{x\}$ and $Y'_{q_1} = Y_{q_1} - \{x\}$.

Proof. For the decision table S, when x is changed to x', there are four situations about the changes of conditional classes and decision classes, which are as follows:

(a) $U_{x'}/B = \{X_1, X_2, \cdots, X'_{p_1}, \cdots, X_m, \{x'\}\}$ and $U_{x'}/D = \{Y_1, Y_2, \cdots, Y'_{q_1}, \cdots, Y_n, \{x'\}\}$;
(b) $U_{x'}/B = \{X_1, X_2, \cdots, X'_{p_1}, \cdots, X'_{p_2}, \cdots, X_m, \}$, $x' \in X'_{p_2}$, and $U_{x'}/D = \{Y_1, Y_2, \cdots, Y'_{q_1}, \cdots, Y_n, \{x'\}\}$;
(c) $U_{x'}/B = \{X_1, X_2, \cdots, X'_{p_1}, \cdots, X_m, \{x'\}\}$, $U_{x'}/D = \{Y_1, Y_2, \cdots, Y'_{q_1}, \cdots, Y'_{q_2}, \cdots, Y_n\}$ and $x' \in Y'_{q_2}$;
(d) $U_{x'}/B = \{X_1, X_2, \cdots, X'_{p_1}, \cdots, X'_{p_2}, \cdots, X_m, \}$, $x' \in X'_{p_2}$ and $U_{x'}/D = \{Y_1, Y_2, \cdots, Y'_{q_1}, \cdots, Y'_{q_2}, \cdots, Y_n\}$, $x' \in Y'_{q_2}$.

For convenience, here introduces a uniform notation about these four situations.

Let $U/B = \{X_1, X_2, \cdots, X_m, X_{m+1}\}$ and $X_{m+1} = \emptyset$. Then, we have $U_{x'}/B = \{X_1, X_2, \cdots, X'_{p_1}, \cdots, X'_{p_2}, \cdots, X_{m+1}, \}$, $X'_{p_1} = X_{p_1} - \{x'\}$ and $X'_{p_2} = \{x'\} \cup X_{p_2}$. Similarly, let $Y_{n+1} = \emptyset$, we can get that $U/D = \{Y_1, Y_2, \cdots, Y_{n+1}\}$, $U_{x'}/D = \{Y_1, Y_2, \cdots, Y'_{q_1}, \cdots, Y'_{q_2}, \cdots, Y_{n+1}\}$ and $Y'_{q_2} = \{x'\} \cup Y_{q_2}$. Obviously, for situation (a), we have $X'_{p_2} = X_{m+1} \cup \{x'\} = \emptyset \cup \{x'\} = \{x'\}$ and $Y'_{q_2} = Y_{n+1} \cup \{x'\} = \{x'\}$. Similarly, for situation (b), we have $Y'_{q_2} = Y_{n+1} \cup \{x'\} = \{x'\}$. And for situation (c), we have $X'_{p_2} = \{x'\}$.

According to the uniform notation, the updating mechanism of complementary conditional entropy is as follows.

$$\begin{aligned}
&E(D|B)\\
&= \sum_{i=1}^{m}\sum_{j=1}^{n} \frac{|X_i \cap Y_j|}{|U|}\frac{|Y_j^c - X_i^c|}{|U|}\\
&= \sum_{i=1}^{m}\sum_{j=1}^{n} \frac{|X_i \cap Y_j|}{|U|}\frac{|X_i - Y_j|}{|U|}.
\end{aligned}$$

And

$$\begin{aligned}
&E_{U_{x'}}(D|B)\\
&= \sum_{i=1}^{m-1}(\sum_{j=1}^{n-1} \frac{|X_i \cap Y_j||X_i - Y_j|}{|U|^2} + \frac{|X_i \cap Y'_{q_1}||X_i - Y'_{q_1}|}{|U|^2} + \frac{|X_i \cap Y'_{q_2}||X_i - Y'_{q_2}|}{|U|^2}) + \sum_{j=1}^{n-1} \frac{|X'_{p_1} \cap Y_j||X'_{p_1} - Y_j|}{|U|^2} + \frac{|X'_{p_1} \cap Y'_{q_1}||X'_{p_1} - Y'_{q_1}|}{|U|^2} + \frac{|X'_{p_1} \cap Y'_{q_2}||X'_{p_1} - Y'_{q_2}|}{|U|^2} +\\
&\quad \sum_{j=1}^{n-1} \frac{|X'_{p_2} \cap Y_j||X'_{p_2} - Y_j|}{|U|^2} + \frac{|X'_{p_2} \cap Y'_{q_1}||X'_{p_2} - Y'_{q_1}|}{|U|^2} + \frac{|X'_{p_2} \cap Y'_{q_2}||X'_{p_2} - Y'_{q_2}|}{|U|^2}\\
&= \sum_{i=1}^{m-1}(\sum_{j=1}^{n-1} \frac{|X_i \cap Y_j||X_i - Y_j|}{|U|^2} + \frac{|X_i \cap Y_{q_1}||X_i - Y_{q_1}|}{|U|^2} + \frac{|X_i \cap Y_{q_2}||X_i - Y_{q_2}|}{|U|^2}) + \sum_{j=1}^{n-1} \frac{|X_{p_1} \cap Y_j| \cdot (|X_{p_1} - Y_j| - 1)}{|U|^2} + \frac{(|X_{p_1} \cap Y_{q_1}| - 1) \cdot |X_{p_1} - Y_{q_1}|}{|U|^2} + \frac{|X_{p_1} \cap Y_{q_2}| \cdot (|X_{p_1} - Y_{q_2}| - 1)}{|U|^2}\\
&\quad + \sum_{j=1}^{n-1} \frac{|X_{p_2} \cap Y_j| \cdot (|X_{p_2} - Y_j| + 1)}{|U|^2} + \frac{|X_{p_2} \cap Y_{q_1}| \cdot (|X_{p_2} - Y_{q_1}| + 1)}{|U|^2} + \frac{(|X_{p_2} \cap Y_{q_2}| + 1) \cdot |X_{p_2} - Y_{q_2}|}{|U|^2}\\
&= \sum_{i=1}^{m+1}\sum_{j=1}^{n+1} \frac{|X_i \cap Y_j||X_i - Y_j|}{|U|^2} + \frac{|X_{p_2} - Y_{q_2}| - |X_{p_1} - Y_{q_1}|}{|U|^2} + \sum_{j=1, j\neq q_2}^{n+1} \frac{|X_{p_2} \cap Y_j|}{|U|^2} - \sum_{j=1, j\neq q_1}^{n+1} \frac{|X_{p_1} \cap Y_j|}{|U|^2}\\
&= E_U(D|B) + \frac{|X'_{p_2} - Y'_{q_2}| - |X'_{p_1} - Y'_{q_1}|}{|U|^2} + \sum_{j=1, j\neq q_2}^{n+1} \frac{|X'_{p_2} \cap Y_j|}{|U|^2} - \sum_{j=1, j\neq q_1}^{n+1} \frac{|X'_{p_1} \cap Y_j|}{|U|^2}\\
&= E_U(D|B) + \frac{|X'_{p_2} - Y'_{q_2}| - |X'_{p_1} - Y'_{q_1}|}{|U|^2} + \sum_{j=1, j\neq q_2}^{n+1} \frac{|X'_{p_2} \cap Y_j|}{|U|^2} + \frac{|X'_{p_2} \cap Y'_{q_2}|}{|U|^2} - \frac{|X'_{p_2} \cap Y'_{q_2}|}{|U|^2} - \sum_{j=1, j\neq q_1}^{n+1} \frac{|X'_{p_1} \cap Y_j|}{|U|^2} - \frac{|X'_{p_1} \cap Y'_{q_1}|}{|U|^2} + \frac{|X'_{p_1} \cap Y'_{q_1}|}{|U|^2}\\
&= E_U(D|B) + \frac{|X'_{p_2} - Y'_{q_2}| - |X'_{p_1} - Y'_{q_1}|}{|U|^2} + \frac{|X'_{p_2}| - |X'_{p_2} \cap Y'_{q_2}|}{|U|^2} - \frac{|X'_{p_1}| - |X'_{p_1} \cap Y'_{q_1}|}{|U|^2}\\
&= E_U(D|B) + \frac{2|X'_{p_2} - Y'_{q_2}| - 2|X'_{p_1} - Y'_{q_1}|}{|U|^2}. \text{ This completes the proof. } \square
\end{aligned}$$

For convenience of introducing combination entropy, here gives a deformation of the definition of combination entropy (see Definition 2). According to $C_N^2 = \frac{N(N-1)}{2}$, the following deformation can be got. Then the updating mechanism of combination conditional entropy is introduced on the basis of this deformation.

Definition 9. *Let $S = (U, C \cup D)$ be a decision table and $B \subseteq C$. One can obtain the condition partition $U/B = \{X_1, X_2, \cdots, X_m\}$ and $U/D = \{Y_1, Y_2, \cdots, Y_n\}$. The combination conditional entropy of B relative to D is defined as*

$$CE(D|B) = \sum_{i=1}^{m}(\frac{|X_i|^2(|X_i|-1)}{|U|^2(|U|-1)} - \sum_{j=1}^{n}\frac{|X_i \cap Y_j|^2(|X_i \cap Y_j|-1)}{|U|^2(|U|-1)}). \tag{6}$$

Theorem 2. *Let $S = (U, C \cup D)$ be a decision table and $B \subseteq C$. The combination conditional entropy of D with respect to B is $E_U(D|B)$. Then, one can obtain the partitions $U/B = \{X_1, X_2, \cdots, X_m\}$ and $U/D = \{Y_1, Y_2, \cdots, Y_n\}$. $x \in X_{p_1}$ and $x \in Y_{q_1}$. If one and only object $x \in U$ is changed to x', then $x' \in X'_{p_2}$ and $x' \in Y'_{q_2}$ ($X'_{p_2} \in U_{x'}/B$ and $Y'_{q_2} \in U_{x'}/D$). The new combination conditional entropy becomes*

$$CE_{U_{x'}}(D|B) = CE_U(D|B) + \Delta,$$

where $\Delta = \frac{|X'_{p_2} - Y'_{q_2}|(3|X'_{p_2}|+3|X'_{p_2}\cap Y'_{q_2}|-5) - |X'_{p_1} - Y'_{q_1}|(3|X'_{p_1}|+3|X'_{p_1}\cap Y'_{q_1}|+1)}{|U|^2}$, $X'_{p_1} = X_{p_1} - \{x\}$ *and* $Y'_{q_1} = Y_{q_1} - \{x\}$.

Proof. Similarly, when x is added to S, there are four same situations as the proof in Theorem 1. Then, the combination conditional entropy is

$$\begin{aligned}
&CE_{U_{x'}}(D|B)\\
&= \sum_{i=1}^{m-1}(\frac{|X_i|^2(|X_i|-1)}{|U|^2(|U|-1)} - \sum_{j=1}^{n-1}\frac{|X_i\cap Y_j|^2(|X_i\cap Y_j|-1)}{|U|^2(|U|-1)} - \frac{|X_i\cap Y'_{q_1}|^2(|X_i\cap Y'_{q_1}|-1)}{|U|^2(|U|-1)} - \frac{|X_i\cap Y'_{q_2}|^2(|X_i\cap Y'_{q_2}|-1)}{|U|^2(|U|-1)}) + \frac{|X'_{p_1}|^2(|X'_{p_1}|-1)}{|U|^2(|U|-1)} - \sum_{j=1}^{n-1}\frac{|X'_{p_1}\cap Y_j|^2(|X'_{p_1}\cap Y_j|-1)}{|U|^2(|U|-1)}\\
&\quad - \frac{|X'_{p_1}\cap Y'_{q_1}|^2(|X'_{p_1}\cap Y'_{q_1}|-1)}{|U|^2(|U|-1)} - \frac{|X'_{p_1}\cap Y'_{q_2}|^2(|X'_{p_1}\cap Y'_{q_2}|-1)}{|U|^2(|U|-1)} + \frac{|X'_{p_2}|^2(|X'_{p_2}|-1)}{|U|^2(|U|-1)} - \sum_{j=1}^{n-1}\frac{|X'_{p_2}\cap Y_j|^2(|X'_{p_2}\cap Y_j|-1)}{|U|^2(|U|-1)} - \frac{|X'_{p_2}\cap Y'_{q_1}|^2(|X'_{p_2}\cap Y'_{q_1}|-1)}{|U|^2(|U|-1)}\\
&\quad - \frac{|X'_{p_2}\cap Y'_{q_2}|^2(|X'_{p_2}\cap Y'_{q_2}|-1)}{|U|^2(|U|-1)}\\
&= \sum_{i=1}^{m-1}(\frac{|X_i|^2(|X_i|-1)}{|U|^2(|U|-1)} - \sum_{j=1}^{n+1}\frac{|X_i\cap Y_j|^2(|X_i\cap Y_j|-1)}{|U|^2(|U|-1)}) + \frac{(|X_{p_1}|-1)^2(|X_{p_1}|-2)}{|U|^2(|U|-1)} - \sum_{j=1}^{n-1}\frac{|X_{p_1}\cap Y_j|^2(|X_{p_1}\cap Y_j|-1)}{|U|^2(|U|-1)} - \frac{(|X_{p_1}\cap Y_{q_1}|-1)^2(|X_{p_1}\cap Y_{q_1}|-2)}{|U|^2(|U|-1)} -\\
&\quad \frac{|X_{p_1}\cap Y_{q_2}|^2(|X_{p_1}\cap Y_{q_2}|-1)}{|U|^2(|U|-1)} + \frac{(|X_{p_2}|+1)^2|X_{p_2}|}{|U|^2(|U|-1)} - \sum_{j=1}^{n-1}\frac{|X_{p_2}\cap Y_j|^2(|X_{p_2}\cap Y_j|-1)}{|U|^2(|U|-1)} - \frac{|X_{p_2}\cap Y_{q_1}|^2(|X_{p_2}\cap Y_{q_1}|-1)}{|U|^2(|U|-1)} - \frac{(|X_{p_2}\cap Y_{q_2}|+1)^2|X_{p_2}\cap Y_{q_2}|}{|U|^2(|U|-1)}\\
&= \sum_{i=1}^{m+1}(\frac{|X_i|^2(|X_i|-1)}{|U|^2(|U|-1)} - \sum_{j=1}^{n+1}\frac{|X_i\cap Y_j|^2(|X_i\cap Y_j|-1)}{|U|^2(|U|-1)}) + \frac{-3|X_{p_1}|^2+5|X_{p_1}|+3|X_{p_1}\cap Y_{q_1}|^2-5|X_{p_1}\cap Y_{q_1}|}{|U|^2(|U|-1)} + \frac{3|X_{p_2}|^2+|X_{p_2}|-3|X_{p_2}\cap Y_{q_2}|^2-|X_{p_2}\cap Y_{q_2}|}{|U|^2(|U|-1)}\\
&= CE_U(D|B) + \frac{|X_{p_2}-Y_{q_2}|(3|X_{p_2}|+3|X_{p_2}\cap Y_{q_2}|+1)}{|U|^2(|U|-1)} - \frac{|X_{p_1}-Y_{q_1}|(3|X_{p_1}|+3|X_{p_1}\cap Y_{q_1}|-5)}{|U|^2(|U|-1)}.
\end{aligned}$$

And because $X'_{p_2} = X_{p_2} \cup \{x\}$ and $Y'_{q_2} = Y_{q_2} \cup \{x\}$, from Theorem 2, one can also get that

$CE_{U_{x'}}(D|B)$

$= CE_U(D|B) + \frac{|X_{p_2}-Y_{q_2}|(3|X_{p_2}|+3|X_{p_2}\cap Y_{q_2}|+1)}{|U|^2(|U|-1)} - \frac{|X_{p_1}-Y_{q_1}|(3|X_{p_1}|+3|X_{p_1}\cap Y_{q_1}|-5)}{|U|^2(|U|-1)}$. This completes the proof. □

The following two theorems are the updating mechanisms of Shannon's conditional entropy shown in Definition 3.

Theorem 3. *Let $S = (U, C \cup D)$ be a decision table and $B \subseteq C$. The Shannon's conditional entropy of D with respect to B is $E_U(D|B)$. Then, one can obtain the partitions $U/B = \{X_1, X_2, \cdots, X_m\}$ and $U/D = \{Y_1, Y_2, \cdots, Y_n\}$. $x \in X_{p_1}$ and $x \in Y_{q_1}$. If one and only object $x \in U$ is changed to x', then $x' \in X'_{p_2}$ and $x' \in Y'_{q_2}$ ($X'_{p_2} \in U_{x'}/B$ and $Y'_{q_2} \in U_{x'}/D$). The new Shannon's conditional entropy becomes*

$$H_{U_{x'}}(D|B) = H_U(D|B) - \Delta,$$

where $\Delta = \sum_{j=1,j\neq q_1}^{n}\frac{|X_{p_1}\cap Y_j|}{|U|}log\frac{|X_{p_1}|}{|X'_{p_1}|} + \sum_{j=1,j\neq q_2}^{n}\frac{|X_{p_2}\cap Y_j|}{|U|}log\frac{|X_{p_2}|}{|X'_{p_2}|} + \frac{|X_{p_1}\cap Y_{q_1}|}{|U|}\cdot log\frac{|X'_{p_1}\cap Y'_{q_1}||X_{p_1}|}{|X_{p_1}\cap Y_{q_1}||X'_{p_1}|} + \frac{|X_{p_2}\cap Y_{q_2}|}{|U|}log\frac{|X'_{p_2}\cap Y'_{q_2}||X_{p_2}|}{|X_{p_2}\cap Y_{q_2}||X'_{p_2}|} + \frac{1}{|U|}log\frac{|X'_{p_1}||X'_{p_2}\cap Y'_{q_2}|}{|X'_{p_2}||X'_{p_1}\cap Y'_{q_1}|}$, $X'_{p_1} = X_{p_1} - \{x\}$ *and* $Y'_{q_1} = Y_{q_1} - \{x\}$.

Proof. Similar to the proof in Theorem 4, it can be easily proved. □

In view of that the formula of Δ is complicated, thus, for the large-scale data tables, an approximate computational formula is proposed in the following theorem.

Theorem 4. *Let $S = (U, C \cup D)$ be a large-scale decision table and $B \subseteq C$. The conditional Shannon's entropy of D with respect to B is $E_U(D|B)$. Then, one can obtain the partitions $U/B = \{X_1, X_2, \cdots, X_m\}$ and $U/D = \{Y_1, Y_2, \cdots, Y_n\}$. $x \in X_{p_1}$ and $x \in Y_{q_1}$. If one and only object $x \in U$ is changed to x', then $x' \in X'_{p_2}$ and $x' \in Y'_{q_2}$ ($X'_{p_2} \in U_{x'}/B$ and $Y'_{q_2} \in U_{x'}/D$). The new Shannon's conditional entropy becomes*

$$H_{U_{x'}}(D|B) \approx H_U(D|B) - \frac{1}{|U|} log \frac{|X'_{p_1}||X'_{p_2} \cap Y'_{q_2}|}{|X'_{p_2}||X'_{p_1} \cap Y'_{q_1}|},$$

where $X'_{p_1} = X_{p_1} - \{x\}$ and $Y'_{q_1} = Y_{q_1} - \{x\}$.

Proof. Similarly, when x is added to S, there are four same situations as the proof in Theorem 1. Then, the Shannon's conditional entropy is

$$\begin{aligned}
&H_{U_{x'}}(D|B)\\
&= -(\sum_{i=1}^{m-1} \frac{|X_i|}{|U|}(\sum_{j=1}^{n-1} \frac{|X_i \cap Y_j|}{|X_i|} log \frac{|X_i \cap Y_j|}{|X_i|} + \frac{|X_i \cap Y'_{q_1}|}{|X_i|} log \frac{|X_i \cap Y'_{q_1}|}{|X_i|} + \frac{|X_i \cap Y'_{q_2}|}{|X_i|} log \frac{|X_i \cap Y'_{q_2}|}{|X_i|}) + \frac{|X'_{p_1}|}{|U|}(\sum_{j=1}^{n-1} \frac{|X'_{p_1} \cap Y_j|}{|X'_{p_1}|} log \frac{|X'_{p_1} \cap Y_j|}{|X'_{p_1}|} + \frac{|X'_{p_1} \cap Y'_{q_1}|}{|X'_{p_1}|} \cdot\\
&\quad log \frac{|X'_{p_1} \cap Y'_{q_1}|}{|X'_{p_1}|} + \frac{|X'_{p_1} \cap Y'_{q_2}|}{|X'_{p_1}|} \cdot log \frac{|X'_{p_1} \cap Y'_{q_2}|}{|X'_{p_1}|}) + \frac{|X'_{p_2}|}{|U|}(\sum_{j=1}^{n-1} \frac{|X'_{p_2} \cap Y_j|}{|X'_{p_2}|} log \frac{|X'_{p_2} \cap Y_j|}{|X'_{p_2}|} + \frac{|X'_{p_2} \cap Y'_{q_1}|}{|X'_{p_2}|} log \frac{|X'_{p_2} \cap Y'_{q_1}|}{|X'_{p_2}|} + \frac{|X'_{p_2} \cap Y'_{q_2}|}{|X'_{p_2}|} \cdot log \frac{|X'_{p_2} \cap Y'_{q_2}|}{|X'_{p_2}|}))\\
&= -(\sum_{i=1}^{m-1} \frac{|X_i|}{|U|}(\sum_{j=1}^{n+1} \frac{|X_i \cap Y_j|}{|X_i|} log \frac{|X_i \cap Y_j|}{|X_i|}) + \frac{|X_{p_1}|}{|U|} \sum_{j=1}^{n-1} \frac{|X_{p_1} \cap Y_j|}{|X_{p_1}|} log \frac{|X_{p_1} \cap Y_j|}{|X'_{p_1}|} + \frac{|X_{p_1}|}{|U|} \cdot \frac{|X_{p_1} \cap Y_{q_1}|-1}{|X_{p_1}|} log \frac{|X'_{p_1} \cap Y'_{q_1}|}{|X'_{p_1}|} + \frac{|X_{p_1}|}{|U|} \cdot \frac{|X_{p_1} \cap Y_{q_2}|}{|X_{p_1}|} \cdot log \frac{|X'_{p_1} \cap Y'_{q_2}|}{|X'_{p_1}|}\\
&\quad + \frac{|X_{p_2}|}{|U|} \sum_{j=1}^{n-1} \frac{|X_{p_2} \cap Y_j|}{|X_{p_2}|} \cdot log \frac{|X_{p_2} \cap Y_j|}{|X'_{p_2}|} + \frac{|X_{p_2}|}{|U|} \cdot \frac{|X_{p_2} \cap Y_{q_1}|}{|X_{p_2}|} log \frac{|X_{p_2} \cap Y_{q_1}|}{|X'_{p_2}|} + \frac{|X_{p_2}|}{|U|} \cdot \frac{|X_{p_2} \cap Y_{q_2}|+1}{|X_{p_2}|} \cdot log \frac{|X'_{p_2} \cap Y'_{q_2}|}{|X'_{p_2}|})\\
&= -(\sum_{i=1}^{m-1} \frac{|X_i|}{|U|} \sum_{j=1}^{n+1} \frac{|X_i \cap Y_j|}{|X_i|} log \frac{|X_i \cap Y_j|}{|X_i|} + \frac{|X_{p_1}|}{|U|} \sum_{j=1}^{n-1} \frac{|X_{p_1} \cap Y_j|}{|X_{p_1}|}(log \frac{|X_{p_1} \cap Y_j|}{|X_{p_1}|} + log \frac{|X_{p_1}|}{|X'_{p_1}|}) + \frac{|X_{p_1}|}{|U|} \cdot \frac{|X_{p_1} \cap Y_{q_1}|}{|X_{p_1}|}(log \frac{|X_{p_1} \cap Y_{q_1}|}{|X_{p_1}|} + log \frac{|X'_{p_1} \cap Y'_{q_1}||X_{p_1}|}{|X_{p_1} \cap Y_{q_1}||X'_{p_1}|})\\
&\quad - \frac{1}{|U|} log \frac{|X'_{p_1} \cap Y'_{q_1}|}{|X'_{p_1}|} + \frac{|X_{p_1}|}{|U|} \cdot \frac{|X_{p_1} \cap Y_{q_2}|}{|X_{p_1}|} \cdot (log \frac{|X_{p_1} \cap Y_{q_2}|}{|X_{p_1}|} + log \frac{|X_{p_1}|}{|X'_{p_1}|}) + \frac{|X_{p_2}|}{|U|} \sum_{j=1}^{n-1} \frac{|X_{p_2} \cap Y_j|}{|X_{p_2}|} \cdot (log \frac{|X_{p_2} \cap Y_j|}{|X_{p_2}|} + log \frac{|X_{p_2}|}{|X'_{p_2}|}) + \frac{|X_{p_2}|}{|U|} \cdot \frac{|X_{p_2} \cap Y_{q_1}|}{|X_{p_2}|}(log \frac{|X_{p_2} \cap Y_{q_1}|}{|X_{p_2}|}\\
&\quad + log \frac{|X_{p_2}|}{|X'_{p_2}|}) + \frac{|X_{p_2}|}{|U|} \cdot \frac{|X_{p_2} \cap Y_{q_2}|}{|X_{p_2}|} \cdot (log \frac{|X_{p_2} \cap Y_{q_2}|}{|X_{p_2}|} + log \frac{|X'_{p_2} \cap Y'_{q_2}||X_{p_2}|}{|X_{p_2} \cap Y_{q_2}||X'_{p_2}|}) + \frac{1}{|U|} log \frac{|X'_{p_2} \cap Y'_{q_2}|}{|X'_{p_2}|}).
\end{aligned}$$

To simplify the calculations of above formula, for the large-scale decision tables, here are some approximated expressions. In view of that $|X_{p_1}|$ and $|X_{p_2}|$ based on the large-scale decision tables are relatively large, respectively, one can get that $|X_{p_1}| \approx |X'_{p_1}|$ and $|X_{p_2}| \approx |X'_{p_2}|$ ($X'_{p_1} = X_{p_1} - \{x\}$ and $X'_{p_2} = X_{p_2} \cup \{x\}$), e.t. $log \frac{|X_{p_1}|}{|X'_{p_1}|} \approx 0$ and $log \frac{|X_{p_2}|}{|X'_{p_2}|} \approx 0$. Similarly, one can also get from $|X_{p_1} \cap Y_{q_1}| \approx |X'_{p_1} \cap Y'_{q_1}|$ and $|X_{p_2} \cap Y_{q_2}| \approx |X'_{p_2} \cap Y'_{q_2}|$ that $log \frac{|X'_{p_1} \cap Y'_{q_1}||X_{p_1}|}{|X_{p_1} \cap Y_{q_1}||X'_{p_1}|} \approx 0$ and $log \frac{|X'_{p_2} \cap Y'_{q_2}||X_{p_2}|}{|X_{p_2} \cap Y_{q_2}||X'_{p_2}|} \approx 0$. Hence, the above formula can be simplified to

$$\begin{aligned}
&H_{U_{x'}}(D|B)\\
&\approx -(\sum_{i=1}^{m} \frac{|X_i|}{|U|} \sum_{j=1}^{n} \frac{|X_i \cap Y_j|}{|X_i|} log \frac{|X_i \cap Y_j|}{|X_i|} - \frac{1}{|U|} log \frac{|X'_{p_1} \cap Y'_{q_1}|}{|X'_{p_1}|} + \frac{1}{|U|} \cdot log \frac{|X'_{p_2} \cap Y'_{q_2}|}{|X'_{p_2}|})\\
&= H_U(D|B) - \frac{1}{|U|} log \frac{|X'_{p_2} \cap Y'_{q_2}||X'_{p_1}|}{|X'_{p_2}||X'_{p_1} \cap Y'_{q_1}|}.
\end{aligned}$$

Thus, this completes the proof. □

5. Attribute reduction algorithm for decision tables with dynamically varying attribute values

Based on the updating mechanisms of the three entropies, this section introduces an attribute reduction algorithm based on information entropy for decision tables with dynamically varying attribute values. In view of that core is another key concept besides redcut in rough set theory [14, 15], this section also gives an algorithm for core computation. In rough set theory, core is the intersection of all reducts of a given table, and core attributes are considered as the indispensable attributes in a reduct. Note that, for the three entropies, the following algorithms are commonly used to update core and reduct.

Algorithm 2. An algorithm to core computation for a dynamic decision table ($ACORE_{x'}$)

Input: A decision table $S = (U, C \cup D)$ and object $x \in U$ is changed to x'
Output: Core attribute $CORE_{x'}$ on $U_{x'}$

Step 1 : Find X'_{p_1} and X'_{p_2}: in $U/C = \{X_1, X_2, \cdots, X_m\}$ and $x \in X_{p_1}$. If x is changed to x', and $x' \in X'_{p_2}$. One have $X'_{p_1} = X_{p_1} - \{x'\}$ and $U_{x'}/C = \{X_1, X_2, \cdots, X'_{p_1}, \cdots, X'_{p_2}, \cdots, X_m\}$.

Step 2 : Find Y'_{q_1} and Y'_{q_2}: in $U/D = \{Y_1, Y_2, \cdots, Y_n\}$ and $x \in Y_{q_1}$. If x is changed to x', and $x' \in Y'_{q_2}$. We have $Y'_{q_1} = Y_{q_1} - \{x'\}$ and $U_{x'}/D = \{Y_1, Y_2, \cdots, Y'_{q_1}, \cdots, Y'_{q_2}, \cdots, Y_n\}$.

Step 3 : Compute $ME_{U_{x'}}(D|C)$ (according to Theorems 1, 2 or 4);

Step 4 : $CORE_{U_{x'}} \leftarrow \emptyset$, for each $a \in C$

1) In $U/(C - \{a\}) = \{M_1, M_2, \cdots, M_{m'}\}$ ($m' \leq m$) and $x \in M_{t_1}$. If x is changed to x', one have $x' \in M'_{t_2}$, $M'_{t_1} = M_{t_1} - \{x'\}$ and $U_{x'}/(C - \{a\}) = \{M_1, M_2, \cdots, M'_{t_1}, \cdots, M'_{t_2}, \cdots, M_{m'}\}$.

2) Compute $ME_{U_{x'}}(D|C - \{a\})$ (according to Theorems 1, 2 or 4).

3) If $ME_{U_{x'}}(D|C - \{a\}) \neq ME_{U_{x'}}(D|C)$, then $CORE_{U_{x'}} = CORE_{U_{x'}} \cup \{a\}$.

Step 5 : Return $CORE_{U_{x'}}$ and end.

Based on updating mechanisms of the three entropies, an attribute reduction algorithm for decision tables with dynamically varying attribute values is introduced in the following. In this algorithm, the existing reduction result is one of inputs, which is used to find its new reduct after data changes.

Algorithm 3. An algorithm to reduct computation for a dynamic decision table ($ARED_{x'}$)

Input: A decision table $S = (U, C \cup D)$, reduct RED_U on U, and the changed object x which is changed to x'

Output: Attribute reduct $RED_{U_{x'}}$ on $U_{x'}$

Step 1 : Find M'_{t_1} and M'_{t_2}: in $U/B = \{M_1, M_2, \cdots, M_{m'}\}$ and $x \in M_{t_1}$. If x is changed to x', and $x' \in M'_{t_2}$. One have $M'_{t_1} = M_{t_1} - \{x'\}$ and $U_{x'}/B = \{M_1, M_2, \cdots, M'_{t_1}, \cdots, M'_{t_2}, \cdots, M_{m'}\}$.

Step 2 : Find Y'_{q_1} and Y'_{q_2}: in $U/D = \{Y_1, Y_2, \cdots, Y_n\}$ and $x \in Y_{q_1}$. If x is changed to x', and $x' \in Y'_{q_2}$. One have $Y'_{q_1} = Y_{q_1} - \{x'\}$ and $U_{x'}/D = \{Y_1, Y_2, \cdots, Y'_{q_1}, \cdots, Y'_{q_2}, \cdots, Y_n\}$.

Step 3 : Compute $ME_{U_{x'}}(D|B)$ (according to Theorems 1, 2 or 4);

Step 4 : Find X'_{p_1} and X'_{p_2}: in $U/C = \{X_1, X_2, \cdots, X_m\}$ and $x \in X_{p_1}$. If x is changed to x', and $x' \in X'_{p_2}$. One have $X'_{p_1} = X_{p_1} - \{x'\}$ and $U_{x'}/C = \{X_1, X_2, \cdots, X'_{p_1}, \cdots, X'_{p_2}, \cdots, X_m\}$.

Step 5 : Compute $ME_{U_{x'}}(D|C)$ (according to Theorems 1, 2 or 4);

Step 6 : If $ME_{U_{x'}}(D|B) = ME_{U_{x'}}(D|C)$, then $RED_{U_{x'}} \leftarrow RED_U$, turn to *Step 8*; else turn to *Step 7*.

Step 7 : $B \leftarrow RED_U$, *while* $ME_{U_{x'}}(D|B) \neq ME_{U_{x'}}(D|C)$ *do*

{ For each $a \in C - B$, compute $Sig^{outer}_{U_{x'}}(a, B, D)$ (according to Theorems 1, 2 or 4 and Definition 6);

Select $a_0 = max\{Sig^{outer}_{U_{x'}}(a, B, D)\}$, $a \in C - B$;

$B \leftarrow B \cup \{a_0\}$.

}

Step 8 : For each $a \in B$ *do*

{ Compute $Sig^{inner}_{U_{x'}}(a, B, D)$;

If $Sig^{inner}_{U_{x'}}(a, B, D) = 0$, then $B \leftarrow B - \{a\}$. }

Step 9 : $RED_{U_{x'}} \leftarrow B$, return $RED_{U_{x'}}$ and end.

The following is time complexities of above two algorithms. First is the time complexity of computing entropy according to Theorems 1, 2, and 4, which is $O(m|C| + n + |X'_{p_1}||Y'_{q_1}| + |X'_{p_2}||Y'_{q_2}|) = O(max(|X'_{p_1}||Y'_{q_1}|, |X'_{p_2}||Y'_{q_2}|))$ (the explanations of $m, n, X'_{p_1}, Y'_{q_1}, X'_{p_2}$ and Y'_{q_2} are shown in Theorems 1, 2, and 4). For convenience, Θ' is used to denote the above time complexity, i.e., $\Theta' = O(max(|X'_{p_1}||Y'_{q_1}|, |X'_{p_2}||Y'_{q_2}|))$.

In the algorithm $ACORE_{x'}$, the time complexity of *Steps 1-3* is Θ'; in *Step 4*, the time complexity is $|C|\Theta'$. Hence, the time complexity of algorithm $ACORE_{x'}$ is

$$O(\Theta' + |C|\Theta') = O(|C|(max(|X'_{p_1}||Y'_{q_1}|, |X'_{p_2}||Y'_{q_2}|))) = O(max(|C||X'_{p_1}||Y'_{q_1}|, |C||X'_{p_2}||Y'_{q_2}|)).$$

In algorithm $ARED_{x'}$, the time complexity of *Steps 1-3* is Θ'; the time complexity of *Steps 4-5* is also $O(\Theta')$; in *Step 7*, the time complexity of adding attributes is $O(|C|\Theta')$; in *Step 8*, the time complexity of deleting redundant attributes is $O(|B|\Theta')$. Hence, the total time complexity of algorithm IA_RED_x is

$$O(\Theta' + |C|\Theta' + |B|\Theta') = O(|C|\Theta') = O(|C|^2|U| + max(|C||X'_{p_1}||Y'_{q_1}|, |C||X'_{p_2}||Y'_{q_2}|)).$$

To stress above findings, the time complexities of computing core and reduct are shown in Table 1. $CA\_CORE_{x'}$ and $CA\_RED_{x'}$ denote classic algorithms based on information entropy for computing core and reduct, respectively.

Table 1: Comparison of time complexity

Entropy	Classic	Incremental
	$O(\|U\|^2)$	$O(\|U\|\|C\| + max(\|X'_{p_1}\|\|Y'_{q_1}\|, \|X'_{p_2}\|\|Y'_{q_2}\|))$
Core	$CA\_CORE_{x'}$	$ACORE_{x'}$
	$O(\|C\|\|U\|^2)$	$O(max(\|C\|\|X'_{p_1}\|\|Y'_{q_1}\|, \|C\|\|X'_{p_2}\|\|Y'_{q_2}\|))$
Reduct	$CA\_RED_{x'}$	$ARED_{x'}$
	$O(\|C\|^2\|U\| + \|C\|\|U\|^2)$	$O(\|C\|^2\|U\| + max(\|C\|\|X'_{p_1}\|\|Y'_{q_1}\|, \|C\|\|X'_{p_2}\|\|Y'_{q_2}\|))$

Table 2: Description of data sets

	Data sets	Samples	Attributes	Classes
1	Backup-large	307	35	19
2	Dermatology	366	33	6
3	Breast-cancer-wisconsin(Cancer)	683	9	2
4	Mushroom	5644	22	2
5	Letter-recognition(Letter)	20000	16	26
6	Shuttle	58000	9	7

In Table 1, $|X'_{p_1}||Y'_{q_1}|$ (or $|X'_{p_2}||Y'_{q_2}|$) is usually much smaller than $|U|^2$. Hence, based on the three entropies, the calculation of proposed algorithms ($ACORE_{x'}$ and $ARED_{x'}$) are usually much smaller than that of the classic algorithms for reduct (or core).

6. Experimental analysis

The objective of the following experiments is to show effectiveness and efficiency of the proposed reduction algorithm $ARED_{x'}$. Due to that the core is a subset of a reduct, we only run the reduction algorithms in the experiments. Data sets used in the experiments are outlined in Table 2, which were all downloaded from UCI repository of machine learning databases. All the experiments have been carried out on a personal computer with Windows XP and Inter(R) Core(TM) 2 Quad CPU Q9400, 2.66 GHz and 3.37 GB memory. The software being used is Microsoft Visual Studio 2005 and programming language is C#.

There are two objectives to conduct the experiments. The first one is to show whether the reduct found by $ARED_{x'}$ is feasible by comparing with that of *CAR* (see in section 6.1). The second one is to compare the efficiency of $ARED_{x'}$ and *CAR* (see in section 6.2). Six UCI data sets are employed to test the two algorithms. *Mushroom* and *Breast-cancer-wisconsin* are data sets with missing values, and for a uniform treatment of all data sets, the objects with missing values have been removed. Moreover, *Shuttle* is preprocessed using the data tool Rosetta.

6.1. Effectiveness analysis

In this subsection, to test the effectiveness of $ARED_{x'}$, four common evaluation measures in rough set theory are employed to evaluate the decision performance of the reducts found by *CAR* and $ARED_{x'}$. The four evaluation measures are approximate classified precision, approximate classified quality, certainty measure and consistency measure.

In [14], Pawlak defined the approximate classified precision and approximate classified quality to describe the precision of approximate classification in rough set theory.

Definition 10. *Let $S = (U, C \cup D)$ be a decision table and $U/D = \{X_1, X_2, \cdots, X_r\}$. The approximate classified precision of C with respect to D is defined as*

$$AP_C(D) = \frac{|POS_C(D)|}{\sum_{i=1}^{r} |\overline{C}X_i|}. \tag{7}$$

Definition 11. *Let $S = (U, C \cup D)$ be a decision table. The approximate classified quality of C with respect to D is defined as*

$$AQ_C(D) = \frac{|POS_C(D)|}{|U|}. \tag{8}$$

In rough set theory, by adopting reduction algorithms, one can get reducts for a given decision table. Then, based on a reduct, a set of decision rules can be generated from a decision table. Here briefly recalls the notions of decision rules [14, 52], which will be used in the following development.

Definition 12. *Let $S = (U, C \cup D)$ be a decision table. $U/C = \{X_1, X_2, \cdots, X_m\}$, $U/D = \{Y_1, Y_2, \cdots, Y_n\}$ and $\cap Y_j \neq \emptyset$. $des(X_i)$ and $des(Y_j)$ are denoted the descriptions of the equivalence classes X_i and Y_j, respectively. A decision rule induced by C is formally defined as*

$$Z_{ij} : des(X_i) \rightarrow des(Y_j), X_i \in U/C, Y_j \in U/D. \tag{9}$$

In [53, 54], certainty measure and support measure were introduced to evaluate a single decision rule. For a rule set, two measures were introduced to measure the certainty and consistency in [15]. However, it has been pointed out that those two measures cannot give elaborate depictions of the certainty and consistency for a rule set in [52]. To address this issue, Qian et al. in [52] defined certainty measure and consistency measure to evaluate the certainty and consistency of a set of decision rules, which has attracted considerable attention [55].

Definition 13. *Let $S = (U, C \cup D)$ be a decision table, $U/C = \{X_1, X_2, \cdots, X_m\}$, $U/D = \{Y_1, Y_2, \cdots, Y_n\}$, and $RULE = \{Z_{ij} | Z_{ij} : des(X_i) \rightarrow des(Y_j), X_i \in U/C, Y_j \in U/D\}$. The certainty measure α of the decision rules on S is defined as*

$$\alpha(S) = \sum_{i=1}^{m} \sum_{j=1}^{n} \frac{|X_i \cap Y_j|^2}{|U||X_i|}. \tag{10}$$

Definition 14. *Let $S = (U, C \cup D)$ be a decision table, $U/C = \{X_1, X_2, \cdots, X_m\}$, $U/D = \{Y_1, Y_2, \cdots, Y_n\}$, and $RULE = \{Z_{ij} | Z_{ij} : des(X_i) \rightarrow des(Y_j), X_i \in U/C, Y_j \in U/D\}$. The consistency measure β of the decision rules on S is defined as*

$$\beta(S) = \sum_{i=1}^{m} \frac{|X_i|}{|U|} [1 - \frac{4}{|X_i|} \sum_{j=1}^{n} \frac{|X_i \cap Y_j|^2}{|X_i|} (1 - \frac{|X_i \cap Y_j|}{|X_i|})]. \tag{11}$$

For each data set in Table 2, 50% objects are selected randomly and replaced by new ones. Then, algorithms *CAR* and $ARED_{x'}$ are employed to update reduct of each varying data set. The generated reducts are shown in Tables 3, 5 and 7, and the evaluation results of reducts based on the four evaluation measures are shown in Tables 4, 6 and 8.

• Comparison of *CAR* and $ARED_{x'}$ based on complementary entropy

It is easy to note from Table 4 the values of the four evaluation measures of the generated reducts by using the two algorithms are very close, and even identical on some data sets. But, according to Table 3, the computational time of $ARED_{x'}$ is much smaller than that of *CAR*. In other words, the performance and decision making of the reduct found by $ARED_{x'}$ are very close to that of *CAR*, but $ARED_{x'}$ is more efficient. Hence, the experimental results indicate that, compared with the classic reduction algorithm *CAR* based on complementary entropy, the algorithm $ARED_{x'}$ can find a feasible reduct in a much shorter time.

• Comparison of *CAR* and $ARED_{x'}$ based on combination entropy

From Tables 5 and 6, it is easy to get that algorithm $ARED_{x'}$ can find a reduct which has same performance and decision making as those of reduct generated by *CAR* in a much shorter time. Thus, compared with *CAR* based on combination entropy, the algorithm $ARED_{x'}$ is more efficient to deal with dynamic data sets.

• Comparison of *CAR* and $ARED_{x'}$ based on Shannon's entropy

According to experimental results in Tables 7 and 8, it is easy to see that performance and decision making of reducts generated by $ARED_{x'}$ and *CAR* are relatively close. But, $ARED_{x'}$ is more efficient than *CAR*. Hence, one can observe that algorithm $ARED_{x'}$ can find a feasible reduct, and save lots of computational time.

Table 3: Comparison of reducts based on complementary entropy

	CAR		$ARED_{x'}$	
Data sets	Reduct	Time/s	Reduct	Time/s
Backup-large	1,4,7,8,10,13,16,22	4.5937	1,4,7,8,10,13,22	0.1406
Dermatology	1,2,3,4,5,14,16,18,19	5.7812	1,2,3,4,5,14,18,19	0.2031
Cancer	1,2,4,6	2.0000	1,2,4,6	0.5000
Mushroom	1,2,3,5,7,8,9,20	486.26	1,2,3,5,7,9,20	37.312
Letter	1,2,3,4,5,8,9,10,11,12,13,15,16	9602.6	1,2,3,4,5,8,9,10,11,12,15,16	358.81
Shuttle	1,2,3,5	17010.1	1,2,3,5	5122.1

Table 4: Comparison of evaluation measures based on complementary entropy

	CAR				$ARED_{x'}$			
Data sets	AQ	AP	α	β	AQ	AP	α	β
Backup-large	1.0000	0.9999	1.0000	1.0000	1.0000	0.9999	1.0000	1.0000
Dermatology	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Cancer	0.9999	1.0000	1.0000	1.0000	0.9999	1.0000	1.0000	1.0000
Mushroom	1.0000	1.0000	1.0000	1.0000	0.9996	0.9993	0.9993	0.9986
Letter	0.9999	1.0000	1.0000	1.0000	0.9999	1.0000	1.0000	1.0000
Shuttle	0.9988	0.9977	0.9976	0.9953	0.9988	0.9977	0.9976	0.9953

Table 5: Comparison of reducts based on combination entropy

	CAR		$ARED_{x'}$	
Data sets	Reduct	Time/s	Reduct	Time/s
Backup-large	1,4,7,8,10,13,16,22	4.5312	1,4,7,8,10,13,22	0.1366
Dermatology	1,2,3,4,5,14,16,18,19	5.7500	1,2,3,4,5,14,18,19	0.2030
Cancer	1,2,4,6	1.9843	1,2,4,6	0.4987
Mushroom	1,2,3,4,7,8,9,20	478.37	1,2,3,4,7,8,9,20	37.301
Letter	1,2,3,4,5,8,9,10,11,12,13,15,16	8825.6	1,2,3,4,5,8,9,11,12,13,15,16	358.81
Shuttle	1,2,3,5	19935.7	1,2,3,5	5089.1

Table 6: Comparison of evaluation measures based on combination entropy

	CAR				$ARED_{x'}$			
Data sets	AQ	AP	α	β	AQ	AP	α	β
Backup-large	1.0000	0.9999	1.0000	1.0000	1.0000	0.9999	1.0000	1.0000
Dermatology	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Cancer	0.9999	1.0000	1.0000	1.0000	0.9999	1.0000	1.0000	1.0000
Mushroom	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Letter	0.9999	1.0000	1.0000	1.0000	0.9999	1.0000	1.0000	1.0000
Shuttle	0.9988	0.9977	0.9976	0.9953	0.9988	0.9977	0.9976	0.9953

Table 7: Comparison of reducts based on Shannon's entropy

	CAR		$ARED_{x'}$	
Data sets	Reduct	Time/s	Reduct	Time/s
Backup-large	1,4,5,6,7,8,10,16,22	5.3281	1,4,5,6,7,8,10,22	0.1368
Dermatology	1,4,5,9,12,14,17,18,21,22,26	5.7500	1,4,5,9,12,14,17,18,26	0.2030
Cancer	2,3,5,6	1.9843	2,3,5,6	0.5125
Mushroom	1,2,3,4,5,9,20,22	482.75	1,2,3,5,9,20,22	37.751
Letter	1,2,3,4,5,8,9,10,11,12,13,15,16	8389.8	1,2,3,4,5,8,9,10,11,13,16	358.87
Shuttle	1,2,3,5	23698.5	1,2,3,5	5130.6

Table 8: Comparison of evaluation measures based on Shannon's entropy

	CAR				$ARED_{x'}$			
Data sets	AQ	AP	α	β	AQ	AP	α	β
Backup-large	0.9999	1.0000	1.0000	1.0000	0.9999	1.0000	1.0000	1.0000
Dermatology	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Cancer	0.9999	1.0000	1.0000	1.0000	0.9999	1.0000	1.0000	1.0000
Mushroom	1.0000	1.0000	1.0000	1.0000	0.9996	0.9993	0.9993	0.9986
Letter	0.9999	1.0000	1.0000	1.0000	0.9999	1.0000	1.0000	1.0000
Shuttle	0.9988	0.9977	0.9976	0.9953	0.9988	0.9977	0.9976	0.9953

Table 9: Comparison of computational time based on LE

Data sets	CAR					$ARED_{x'}$				
	10%	20%	30%	40%	50%	10%	20%	30%	40%	50%
Backup-large	4.6875	4.8594	5.0625	4.4688	4.5937	0.0312	0.0468	0.0781	0.1250	0.1406
Dermatology	5.3906	5.5781	5.6093	6.6250	5.7812	0.0468	0.0781	0.1250	0.1562	0.2031
Cancer	1.4843	1.5937	1.7187	1.9062	2.0000	0.0781	0.1718	0.2656	0.4062	0.5000
Mushroom	221.01	283.78	368.71	411.28	468.26	4.3125	9.9218	17.906	29.000	37.312
Letter	8133.2	8283.2	8630.1	9260.2	9602.6	83.625	140.25	220.48	292.68	358.81
Shuttle	12909.8	13905.3	14523.9	16100.1	17010.1	776.10	1707.3	2979.3	4098.7	5122.1

Table 10: Comparison of computational time based on CE

Data sets	CAR					$ARED_{x'}$				
	10%	20%	30%	40%	50%	10%	20%	30%	40%	50%
Backup-large	4.8125	4.8750	5.0468	4.4531	4.5312	0.0412	0.0568	0.0781	0.1193	0.1366
Dermatology	5.3906	5.4687	5.5156	5.7500	5.7500	0.0478	0.0781	0.1250	0.1563	0.2030
Cancer	1.4843	1.5781	1.7187	1.9062	1.9843	0.0781	0.1718	0.2656	0.4063	0.4987
Mushroom	222.75	253.15	371.06	424.51	478.37	4.3712	9.9118	17.860	28.937	37.301
Letter	7201.4	7669.8	8011.8	8485.2	8825.6	83.505	142.05	200.45	289.66	358.81
Shuttle	14122.1	15468.9	16236.1	18896.9	19935.7	758.11	1668.3	3005.3	4120.8	5089.1

6.2. Efficiency analysis

The objective of experiments in this subsection is to further illustrate efficiency of algorithm $ARED_{x'}$. For each data set in Table 2, 10%, 20%, $\cdots$, 50% objects are selected, in order, and are replaced by new ones. For each data set after each variation (from 10% to 50%), algorithms CAR and $ARED_{x'}$ are used to update reducts, respectively. The efficiency of the two algorithms are demonstrated by comparing their computational time. Experimental results are shown in Tables 9-11. 10%, 20%, $\cdots$, 50% in the tables mean 10%, 20%, $\cdots$, 50% objects with data values being varied, respectively.

Based on the three entropies, it is easy to see from the Tables 9-11 that, for each data set after each variation, the computational time of algorithm $ARED_{x'}$ is much smaller than that of the classic reduction algorithm CAR, especially for the larger data sets *Mushroom* and *Letter*. In addition, with the number of varying objects increasing (from 10% to 50%), the computational time of $ARED_{x'}$ is always much smaller than that of CAR. Hence, the experimental results show that algorithm $ARED_{x'}$ is efficient to solving data sets with dynamically varying data values.

6.3. Related discussion

This subsection summarizes the advantages of algorithm $ARED_{x'}$ for generating reduct and offers explanatory comments. Obviously, in above two subsections, the experimental results better illustrate effectiveness and efficiency of $ARED_{x'}$.

• Algorithm $ARED_{x'}$ based on each of the three entropies can find a feasible reduct of a given dynamic decision table.

According to experimental results in Section 6.1, it is easy to get that the decision performance of reducts generated by CAR and $ARED_{x'}$ are very close, and even identical on some data sets. Hence, compared with the classic reduction algorithms based on the three entropies, the reduct generated by $ARED_{x'}$ can be considered as a feasible reduct.

• Compared with the classic reduction algorithms (CAR) based the three entropies, $ARED_{x'}$ finds a reduct in a very efficient manner.

Table 11: Comparison of computational time based on SE

Data sets	CAR					$ARED_{x'}$				
	10%	20%	30%	40%	50%	10%	20%	30%	40%	50%
Backup-large	4.5937	4.7343	5.0000	5.2187	5.3281	0.0402	0.0568	0.0781	0.1194	0.1368
Dermatology	5.1875	5.5781	4.2500	6.3125	6.6562	0.0478	0.0750	0.1250	0.1565	0.2036
Cancer	1.4843	1.6093	1.7500	1.8750	1.9843	0.0781	0.1709	0.2625	0.4063	0.5125
Mushroom	221.84	256.36	330.31	370.59	482.75	4.3825	9.9288	17.876	28.937	37.751
Letter	7189.7	7502.8	7686.7	8015.2	8389.8	83.625	141.25	223.48	295.48	358.87
Shuttle	12968.3	15126.7	17356.1	20123.3	23698.5	775.10	1705.3	2985.3	4101.5	5130.6

Experimental results in Section 6.2 show that, based on the three entropies, the computational time of generating reduct by using $ARED_{x'}$ is much shorter than that of CAR.

• The development in the paper may make an important contribution to deal with large-scale dynamic data sets in applications.

The experimental results show that the efficiency of $ARED_{x'}$ is obvious in solving large-scale dynamic data sets. In reality, acquiring knowledge from large-scale complicated data sets is still a challenging issue. It is our wish that this paper provides new techniques for dealing with large-scale dynamic data sets.

7. Conclusions and future work

Feature selection for dynamic data sets is still a challenging issue in the field of artificial intelligence. In this paper, based on three representative entropies, an attribute reduction algorithm is proposed to update reduct of data sets with dynamically varying data values. The experimental results show that, compared with the classic reduction algorithms based on the three entropies, this algorithm can generate a feasible reduct in a much shorter time. It is our wish that this study provides new views and thoughts on dealing with large-scale and complicated dynamic data sets in applications.

It should be pointed out that updating mechanisms of the three entropies introduced in this paper are only applicable when data are varied one by one, whereas many real data may vary in groups in application. This gives rise to many difficulties for the proposed feature selection algorithm to deal with. Hence, it is expected to carry out the following work to improve efficiency of selecting useful features in dynamic data sets in the future:

• Developing group updating mechanisms of entropies and relative feature selection algorithms.

• Discernibility matrix is one of key concepts in rough set. Future work may include analyzing discernibility matrix for data sets with dynamically varying data values.

• Designing efficient feature selection algorithms based on generalized rough set models such as incomplete rough set model, dominance rough set model and multi-granulation rough set model.

Acknowledgment

This work was supported by National Natural Science Fund of China (Nos. 71031006, 60903110, 70971080), Initial Special Research for 973 Program of China(973)(No. 2011CB311805), The Research Fund for the Doctoral Program of Higher Education (20101401110002).

References

[1] F. Hu, G. Y. Wang, H. Huang, Y. Wu, Incremental attribute reduction based on elementary sets // Proceedings of the 10th International Conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing, Regina, Canada (2005) 185-193.

[2] J.Y.Liang, W.Wei, Y.H.Qian, An incremental approach to computation of a core based on conditional entropy, Chinese Journal of System Engineering Theory and Practice 4 (2008) 81-89.

[3] D. Liu, T. R. Li, D. Ruan, W. L. Zou, An incremental approach for inducing knowledge from dynamic information systems, Fundamenta Informaticae 94 (2009) 245-260.
[4] M. Orlowska, Maintenance of knowledge in dynamic information systems//R. Slowinski ed. Proceeding of the intelligent decision support, Handbook of Applications and Advances of the Rough Set Theory, Dordrecht: Kluwer Academic Publishers (1992) 315-330.
[5] L. Shan, W. Ziarko, Data-based acquisition and incremental modification of classification rules, Computational Intelligence 11 (1995) 357-370.
[6] M. Yang, An incremental updating algorithm for attributes reduction based on the improved discernibility matrix, Chinese Journal of Computers 30(5) (2007) 815-822.
[7] Z. Zheng, G. Wang G, RRIA: a rough set and rule tree based incremental knowledge acquisition algorithm, Fundamenta Informaticae 59 (2004) 299-313.
[8] C. C. Chan, A rough set approach to attribute generalization in data mining, Information Science 107 (1998) 169-176.
[9] T. R. Li, D. Ruan, W. Geert, J. Song, Y. Xu, A rough sets based characteristic relation approach for dynamic attribute generalization in data mining, Knowledge-Based System 20 (2007) 485-494.
[10] Y. Cheng, The incremental method for fast computing the rough fuzzy approximations, Data & Knowledge Engineering 70 (2011) 84-100.
[11] D. Liu, J. B. Zhang, T. R. Li, A probabilistic rough set approach for incremental learning knowledge on the change of attribute. In: Proceedings 2010 International Conference on Foundations and Applications of Computational Intelligence (2010) 722-727.
[12] H. M. Chen, T. R. Li, S. J. Qiao, D. Ruan, A rough set based dynamic maintenance approach for approximations in coarsening and refining attribute values. International Journal of Intellegent Systems 25 (2010) 1005-1026.
[13] D. Liu, T. R. Li, G. R. Liu, P. Hu, An incremental approach for inducing interesting knowledge based on the change of attribute values. In: Proceedings 2009 IEEE International Conference on Granular Computing, Nanchang, China (2009) 415-418.
[14] Z. Pawlak, Rough Sets: Theoretical Aspects of Reasoning about Data, Kluwer Academic Publishers, Boston, 1991.
[15] Z. Pawlak, A. Skowron, Rudiments of rough sets, Information Sciences 177(1) (2007) 3-27.
[16] H. S. Own, A. Abraham, A new weighted rough set framework based classification for Egyptian NeoNatal Jaundice, Applied Soft Computing 12 (3) (2012) 999-1005.
[17] K. Kaneiwa, A rough set approach to multiple dataset analysis, Applied Soft Computing 11 (2) (2011) 2538-2547.
[18] A. K. Das, J. Sil, An efficient classifier design integrating rough set and set oriented database operations, Applied Soft Computing 11 (2) (2011) 2279-2285.
[19] P. Dey, S. Dey, S. Datta, J. Sil, Dynamic discreduction using Rough Sets, Applied Soft Computing 11 (5) (2011) 3887-3897.
[20] R. R. Chen, Y. I. Chiang, P. P. Chong, Y. H. Lin, H. K Chang, Rough set analysis on call center metrics, Applied Soft Computing 11 (4) (2011) 3804-3811.
[21] W. Wei, J. Y. Liang, Y. H. Qian, A comparative study of rough sets for hybrid data, Information Sciences 190 (1) (2012) 1-16.
[22] D. G. Chen, Q. H. Hu, Y. P. Yang, Parameterized attribute reduction with Gaussian kernel based fuzzy rough sets, Information Sciences 181 (23) (2011) 5169-5179.
[23] Q.H. Hu, Z.X. Xie, D.R. Yu, Hybrid attribute reduction based on a novel fuzzy-rough model and information granulation, Pattern Recognition 40(2007) : 3509 - 3521.
[24] Q.H. Hu, D.R. Yu, Z.X. Xie, Information-preserving hybrid data reduction based on fuzzy-rough techniques, Pattern Recognition Letters 27(5) (2006) 414 - 423.
[25] Y. Y. Yao, The superiority of three-way decisions in probabilistic rough set models, Information Sciences 181 (6) (2011) 1080-1096.
[26] G. Gediga, I. Düntsch, Rough approximation quality revisited, Artificial Intelligence 132 (2001) 219 - 234.
[27] R. Jensen, Q. Shen, Fuzzy-rough sets assisted attribute selection, IEEE Transactions on Fuzzy Systems 15 (1) (2007) 73 - 89.
[28] J.Y. Liang, Y.H. Qian, Information granules and entropy theory in information systems, Science in China(Series F) 51(10) (2008) 1427 - 1444.
[29] J.S. Mi, Y. Leung, W.Z. Wu, An uncertainty measure in partition-based fuzzy rough sets, International Journal General Systems 34 (2005) 77 - 90.
[30] W. Pedrycz, G. Vukovich, Feature analysis through information granulation and fuzzy sets, Pattern Recognition 35 (2002) 825 - 834.
[31] Y.H. Qian, C.Y. Dang, J.Y. Liang, D.W. Tang, Set-valued ordered information systems, Information Sciences, 179 (2009) 2809 - 2832.
[32] Z.B. Xu, J.Y. Liang, C.Y. Dang, K.S. Chin, Inclusion degree: a perspective on measures for rough set data analysis, Information Sciences 141 (2002) 227 - 236.
[33] J. Y. Liang, K. S. Chin, C. Y. Dang, C.M. Yam Richid, A new method for measuring uncertainty and fuzziness in rough set theory, International Journal of General Systems 31(4) (2002) 331 - 342.
[34] J.Y. Liang, Z.Z. Shi, The information entropy, rough entropy and knowledge granulation in rough set theory, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 12(1) (2004) 37 - 46.
[35] Y.H. Qian, J.Y. Liang, Combination entropy and combination granulation in rough set theory, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 16 (2) (2008) 179 - 193.
[36] C.E.Shannon, The mathematical theory of communication, The Bell System Technical Journal, 27(3,4) (1948) 373 - 423.
[37] Y.H. Qian , J.Y. Liang, W. Pedrycz, C.Y. Dang, Positive approximation: an accelerator for attribute reduction in rough set theory, Artificial Intelligence 174 (2010) 597 - 618.
[38] G. Y. Wang, H. Yu, D.C. Yang, Decision table reduction based on conditional information entropy, Chinese Journal of Computer 25 (7) (2002) 759 - 766.
[39] M. Dash, H. Liu, Consistency-based search in feature selection, Artificial Intelligence 151 (2003) 155 - 176.
[40] A. Skowron, C. Rauszer, The discernibility matrices and functions in information systems, In: R Slowiński(Eds), Intelligent Decision Support, Handbook of Applications and Advances of the Rough Sets Theory, Kluwer Academic Publisher, Dordrecht, 1992.
[41] M. Kryszkiewicz, P. Lasek, FUN: fast discovery of minimal sets of attributes functionally determining a decision attribute, Transactions on Rough Sets 9 (2008) 76 - 95.
[42] W. Ziarko, Variable precision rough set model, Journal of Computer and System Science 46(1) (1993) 39 - 59.

[43] Y. Y. Yao, Decision-theoretic rough set models, Lecture Notes in Artificial Intelligence 4481 (2007) 1 - 12.
[44] Y. Y. Yao, Y. Zhao, Attribute reduction in decisiontheoretic rough set models, Information Sciences 178 (2008) 3356 - 3373.
[45] S. Greco, B. Matarazzo, R. Slowinski, Rough approximation by dominance relations, International Journal of Intelligent Systems 17 (2002) 153-171.
[46] Y. H. Qian, C. Y. Dang, J. Y. Liang, D. W. Tang, Set-valued ordered information systems, Information Sciences, 179 (2009) 2809-2832.
[47] M. W. Shao, W. X. Zhang, Dominance relation and rules in an incomplete ordered information system, International Journal of Intelligent Systems 20 (2005) 13-27.
[48] D. Dubois, H. Prade, Rough fuzzy sets and fuuzy rough sets, International Journal of General Systems 17 (1990) 191 - 209.
[49] X. H. Hu, N. Cercone, Learning in relational databases: a rough set approach, International Journal of Computational Intelligence 11(2) (1995) 323 - 338.
[50] D. Slezak, Approximate entropy reducts, Fundamenta Informaticae 53 (3-4) (2002) 365 - 390.
[51] Z.Y. Xu, Z.P. Liu, B.R. Yang, W. Song, A quick attribute reduction algorithm with complexity of $\max(O(|C||U|), O(|C|^2|U/C|))$, Chinese Journal of Computer 29 (3) (2006) 391 - 398.
[52] Y. H. Qian, J.Y. Liang, D.Y. Li, H.Y. Zhang, C.Y. Dang, Measures for evaluating the decision performance of a decision table in rough set theory, Information Sciences 178 (2008) 181 - 202.
[53] I. Düntsch, G. Gediga, Uncertainty measures of rough set prediction, Artificial Intelligence 106 (1998) 109 - 137.
[54] V.N. Huynh, Y. Nakamori, A roughness measure for fuzzy sets, Information Sciences 173 (2005) 255 - 275.
[55] W. Wei, J.Y. Liang, Y.H. Qian, F. Wang, C.Y. Dang, Comparative study of decision performance of decision tables induced by attribute reductions, International Journal of General Systems, 39 (8) (2010) 813 - 838.