Information Sciences 211 (2012) 48–67

Contents lists available at SciVerse ScienceDirect

Information Sciences

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

Attribute reduction of data with error ranges and test costs

Fan Min *, William Zhu

Lab of Granular Computing, Zhangzhou Normal University, Zhangzhou 363000, China

ARTICLE INFO

Article history:
Received 25 August 2011
Received in revised form 29 February 2012
Accepted 18 April 2012
Available online 5 May 2012

Keywords:
Cost-sensitive learning
Test cost
Error range
Neighborhood
Covering rough set

ABSTRACT

In data mining applications, we have a number of measurement methods to obtain a data item with different test costs and different error ranges. Test costs refer to time, money, or other resources spent in obtaining data items related to some object; observational errors correspond to differences in measured and true value of a data item. In supervised learning, we need to decide which data items to obtain and which measurement methods to employ, so as to minimize the total test cost and help in constructing classifiers. This paper studies this problem in four steps. First, data models are built to address error ranges and test costs. Second, error-range-based covering rough set is constructed to define lower and upper approximations, positive regions, and relative reducts. A closely related theory deals with neighborhood rough set, which has been successfully applied to heterogeneous attribute reduction. The major difference between the two theories is the definition of neighborhood. Third, the minimal test cost attribute reduction problem is redefined in the new theory. Fourth, both backtrack and heuristic algorithms are proposed to deal with the new problem. The algorithms are tested on ten UCI (University of California – Irvine) datasets. Experimental results show that the backtrack algorithm is efficient on rational-sized datasets, the weighting mechanism for the heuristic information is effective, and the competition approach can improve the quality of the result significantly. This study suggests new research trends concerning attribute reduction and covering rough set.

1. Introduction

Cost-sensitive learning is among the most challenging problems in data mining [62]. It has attracted much research interest from different data-mining societies working on topics such as decision trees (see, e.g., [13,30,55]), artificial neural networks (see, e.g., [24,68]), rough set (see, e.g., [37,38,65,66]), and Bayes networks (see, e.g., [6]). Most work addresses misclassification costs [6,24,32,65,66,68], but few address test costs [37,38,55], and even fewer address both [30,55].

Test cost is the time, money, or other resources one pays for obtaining a data item of an object. The topic has drawn our attention recently due to its broad applications. Based on data models discussed in [38], a number of situations have been identified, and respective problems have been defined. Specifically, test-cost-sensitive reduct problems were defined in [16,17,37]. These problems aim at finding a test set minimizing the test cost, while preserving the discernibility of the original decision system. The test cost constraint issue was introduced in [39,41] to address the situation where the total test cost one can afford is limited. The optimal sub-reduct problem was defined for this situation. Moreover, the problem was reconsidered in the dynamic environment, where both test costs and the constraint can change over time [40]. In these works, backtrack and heuristic algorithms were implemented and compared through an open source software Coser [42].

* Corresponding author. Tel.: +86 133 7690 8359.
E-mail addresses: minfanphd@163.com (F. Min), williamfengzhu@gmail.com (W. Zhu).

0020-0255/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ins.2012.04.031

Observational error is the difference between a measured value of the data item and its true value. In applications, we do not know exactly the observation error; however, the error range is usually known. For example, we often use 37.7 ± 0.1 °C to indicate the body temperature of a person, where 37.7 °C is the observed value, and ±0.1 °C is the error range. Other forms of observational error exist for more complicated data. For example, we use 256×256 images to store the CT (computed tomography) information. We could use $\pm\frac{1}{256}$ to represent the error range; however, the physical meaning is quite different from the body temperature. Data with observational error is a form of uncertain data that has become a hot topic in data mining (see, e.g., [1,7]).

A number of measurement methods could exist to obtain a data item. For example, one can use a mercury thermometer, an electronic thermometer, or an infrared thermometer to test the body temperature. These methods require different test costs; specifically, the respective time requirements are 5 min, 1 min, and 2 s. As a result, the error ranges are also different. The infrared thermometer has the widest error range. Throughout this paper, we assume that the test cost of each data item, and the error range of each measurement method are known. We use the term "test" to indicate the process of obtaining a data item using one particular measurement method. It also refers to an attribute in an information system, or a conditional attribute in a decision system. Different measurement methods for the same data are viewed as different tests, and correspond to different attributes.

For classification purposes, we do not have to undertake all tests, nor do we have to choose the most sophisticated measurement method for each data item. Instead, we would like to find out the most economical test set which is sufficient to make the decision. This motivates us to define a new problem for this kind of test sets. In fact, the motivation of the new problem is similar to the work in [37]. However, the data under consideration are nominal ones from [37]. In contrast, this work deals with numerical data; hence, the new problem has different application areas.

We build a new model, called an error-range-based covering rough set, to address the observational error issue and formalize our problem. Note that the term "covering" instead of "neighborhood" is employed because a neighborhood system always generates a covering of the universe. The concept of reducts is employed to describe information preservation. Consequently, the new problem is referred to as the *test-cost-sensitive attribute reduction through error-range-based covering rough set*, or TARER for brevity. In fact, there exist some neighborhood systems [11,26,29], covering rough set frameworks [5,69,70,72], neighborhood rough set frameworks [18–20,25,60], near set [47], vague set [48], and generalized rough set over fuzzy lattices [10,12,21,33], that have gained much success in both theory and application. Specifically, attribute reduction in neighborhood rough set [14,18–20] can help to increase the prediction accuracy of the classifier. Although showing similarities, the existing work in [18–20] are essentially different from ours in three ways. First, the models constructed require a user-chosen distance function and a user-specified distance threshold δ. In contrast, our model is based on error ranges that are intrinsic to data. Second, the objective of reduct algorithms for these models is to improve the classification accuracy. In contrast, our objective is to minimize the total test cost. Third, δ is used in these models to adjust the neighborhood size. In contrast, in our model different tests for the same data item produce not only different error ranges, but also different observed values on one data item. For example, with one thermometer we obtained the value 37.7 ± 0.1 °C, whereas with another we obtain 37.71 ± 0.03 °C.

The main contributions of the work are fourfold. First, we build the data model to formalize the situation mentioned above. Only numerical data are considered to simplify the discussion. Second, we build the computational model, namely the error-range-based covering rough set, and define key concepts such as lower and upper approximations, positive regions, and relative reducts. Third, we redefine the attribute reduction problem under the new model. Similar to the problem defined in [37], the objective is to minimize the total test cost. Fourth, we propose a backtrack algorithm to find an optimal reduct, a heuristic algorithm to find a sub-optimal reduct, and a competition approach to improve the performance of the heuristic algorithm. Experiments on ten datasets from the UCI library undertaken with open source software Coser [42] validate the effectiveness of these algorithms.

The rest of the paper is organized as follows: Section 2 presents the data models. Error ranges and test costs are considered one after another. Section 3 presents the computational model, namely error-range-based covering rough set model. The test-cost-sensitive attribute reduction problem under the new model is also defined in this section. Next, Section 4 presents two algorithms and the competition approach. Experiments settings and results are discussed in Section 5. Finally, Section 6 presents concluding remarks and points out further research trends.

2. Data models

This section studies data models. We start from basic information systems and decision systems. Next, we introduce observational errors to tests, and propose information and decision systems with error ranges. Finally we introduce test costs and define test-cost-sensitive decision systems with error ranges.

2.1. Information systems and decision systems

Information systems and decision systems are fundamental in machine learning and data mining. These are often stored in relational databases. For completeness, these are defined below.

Definition 1 [64]. An *information system* (IS) S is the 4-tuple:

$$S = (U, A, V = \{V_a | a \in A\}, I = \{I_a | a \in A\}), \tag{1}$$

where U is a finite set of objects called the universe, A is the set of attributes (tests), V_a is the set of values for each $a \in A$, and $I_a: U \to V_a$ is an information function for each $a \in A$.

In many applications, we have a number of decision attributes. Therefore we can construct decision trees or generate decision rule to classify unseen objects.

Definition 2 [64]. A *decision system* (DS) S is the 5-tuple:

$$S = (U, C, D, V = \{V_a | a \in C \cup D\}, I = \{I_a | a \in C \cup D\}), \tag{2}$$

where U is a finite set of objects called the universe, C is the set of conditional attributes, D is the set of decision attributes with only discrete values, V_a is the set of values for each $a \in C \cup D$, $I_a: U \to V_a$ is an information function for each $a \in C \cup D$.

For brevity, V and I are sometimes dropped and these systems denoted by $S = (U, A)$ and $S = (U, C, D)$ [45].

In most applications, $D = \{d\}$; that is, we are given only one decision attribute. If $|D| > 1$, we can construct $|D|$ decision systems, each with exactly one decision attribute. An example decision system is listed in Table 1, where $D = \{\text{class}\}$. It is a subtable of the Iris dataset. Conditional attributes are normalized to help processing and comparison. One possible normalization approach is to employ the linear function $y = (x - min)/(max - min)$, where x is the initial value, y is the normalized value, min and max are the minimal and maximal value of the attribute domain, respectively. Note that normalization is neither essential to our model nor critical to our algorithm. Table 1 can be also viewed as an information system where $A = \{\text{sepal-length, sepal-width, petal-length, petal-width, class}\}$.

2.2. Information systems and decision systems with error ranges

Attribute values of information systems and conditional attribute values of decision systems are often obtained through certain tests. Therefore, throughout this paper, attributes of information systems and conditional attributes of decision systems will also be called *tests*. Observational errors are intrinsic to tests especially when values are numerical. We include this issue in our model to make it applicable to more real data.

Definition 3. An *information system with error ranges* (IS-ER) is the 5-tuple:

$$S = (U, A, V, I, e), \tag{3}$$

where U, A, V, and I have the same meaning as Definition 1, $e: A \to \mathbb{R}^+ \cup \{0\}$ is the maximal observational error of $a \in A$, and $\pm e(a)$ is the error range of a.

Error free tests are tests with $e(a) \equiv 0$. In most cases, these are tests producing nominal outputs, which are transferred to numerical ones. If all tests are error free, an IS-ER degrades to an IS. Therefore IS-ER is a generalization of IS. Note that values of decision attributes are often specified by experts. Hence decision attributes are also prone to error. This issue involves the cost of teacher [56] and will not be discussed further in this paper.

An example observation error vector is listed in Table 2. We deliberately assign large values to the maximal observation errors to help explain some concepts through examples, where the number of objects is small. In applications these values should be small enough to make the data useful.

Similarly, we extend the concept of decision systems to consider observational errors.

Definition 4. A *decision system with error ranges* (DS-ER) S is the 6-tuple:

$$S = (U, C, D, V, I, e), \tag{4}$$

Table 1
An example decision system.

Plant	Sepal-length	Sepal-width	Petal-length	Petal-width	Class
x_1	0.24	0.64	0.10	0.05	Setosa
x_2	0.18	0.41	0.10	0.05	Setosa
x_3	0.12	0.50	0.07	0.05	Setosa
x_4	0.76	0.45	0.60	0.52	Versicolor
x_5	0.35	0.09	0.38	0.43	Versicolor
x_6	0.65	0.32	0.52	0.52	Versicolor
x_7	0.59	0.55	0.86	1.00	Virginica
x_8	0.44	0.27	0.64	0.71	Virginica
x_9	0.82	0.41	0.83	0.81	Virginica

Table 2
An example observational error vector.

a	Sepal length	Sepal width	Petal length	Petal width
$e(a)$	0.08	0.09	0.05	0.07

where U, C, D, V and I have the same meaning as Definition 2, $e : C \to \mathbb{R}^+ \cup \{0\}$ is the maximal observational error of $a \in C$, and $\pm e(a)$ is the error range of a.

Note that the concept of observational error is not applicable to decision attributes which are discrete. Tables 1 and 2 represent a DS-ER. Obviously, DS-ER is a generalization of DS.

In some applications, error ranges can be asymmetric. For example, let the true value be v, the observed value be v', we may require $v' \in [v, v + e]$. In other words, the value should never be underestimated. Because it is straightforward to adjust the model to suit such kind of conditions, we only consider the one mentioned in Definition 4.

Another observation is that error ranges at the boundaries are different. For example, let the upper bound of a thermometer be 100 °C and the error range be ±0.1 °C. When the true value is 100 °C, the observed value is in the range [99.9 °C, 100 °C] instead of [99.9 °C, 100.1 °C]. Fortunately, the boundaries influence the data generation, but not the data model, nor the algorithms processing the data. Therefore this issue will not be addressed further.

2.3. Test-cost-independent decision system with error ranges

Test costs are also intrinsic to data. We introduce that next into the data model. For brevity, we only discuss models based on decision systems. One can follow the discussion for information systems.

Definition 5. A *test-cost-independent decision system with error ranges* (TCI-DS-ER) S is the 7-tuple:

$$S = (U, C, D, V, I, e, c), \tag{5}$$

where U, C, D, V, I, and e have the same meanings as in a DS-ER, $c : C \to \mathbb{R}^+$ is the test cost function. Test costs are independent of one another, that is, $c(B) = \sum_{a \in B} c(a)$ for any $B \subseteq C$.

Again we only consider the simplest case where test costs are independent. More complicated situations can be found in [38], and one can build more sophisticated models on these. An example TCI-DS-ER is given by Tables 1–3. Naturally, DS-ER can be viewed as a special case of TCI-DS-ER where $c(a) = c$ for all $a \in C$ where c is a constant.

Note that in this model, tests costs are not applicable to decision attributes. In some applications one could consider the cost of teachers [56], and a decision attribute will be involved. However, both the motivation and the model are different.

3. Error-range-based covering rough set

In information systems, neighborhood [19,23,26,63] is an important concept when identifying a set of objects centered on a given one. From the viewpoint of granular computing [2,27,28], a neighborhood is also called an information granule. From the viewpoint of covering rough set [70,72], a neighborhood is also called a covering element, or a block.

In most existing work, two issues are employed to identify the neighborhood. The first is the distance function, which computes how far away two objects are. Well-known distance functions include the Manhattan distance, the Euclidean distance, and the Chebychev distance; these correspond to 1-norm, 2-norm, and ∞-norm, respectively. Fig. 1 [19,20] illustrates the neighborhoods of x in a two-dimension real space. A neighborhood around x based on the Manhattan distance is a square with diagonals parallel to the coordinate axes; one based on the Euclidean distance is a circle, whereas one based on the Chebychev distance is a square with edges parallel to the coordinate axes.

The second is a user-specified parameter. For the k-nearest neighbor algorithm (k-NN) [9], k is the parameter and the top k nearest objects will be viewed as neighbors. For the neighborhood rough set model [19,20], δ is a distance parameter and objects with a distance no further than δ are viewed as neighbors. All these approaches require the user to make certain choices.

In this section, we study error-range-based covering rough set. As mentioned earlier, error ranges are intrinsic to data rather than specified by users. The new model requires neither the distance function nor the distance upper bound δ, hence the user is less involved in the model building process. Note that properties shared by neighborhood systems still hold for the new model. For example, the neighborhood relation induced by the new model can also be considered as a kind of

Table 3
An example test cost vector.

a	Sepal length	Sepal width	Petal length	Petal width
$c(a)$	\$5	\$5	\$4	\$4

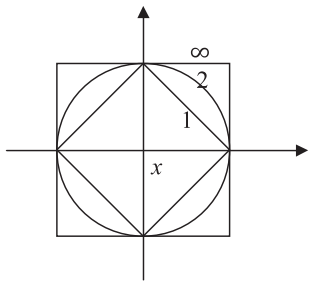

Fig. 1. Conventional neighborhoods.

tolerance relations. Therefore both covering rough set [70,72] and tolerance-relation-based rough set [3,22,35,53] can be established on the new model.

3.1. Error-range-based covering

With Definition 3, a new type of neighborhood can be defined as follows.

Definition 6. Let $S = (U, A, V, I, e)$ be an IS-ER. Given $x_i \in U$ and $B \subseteq C$, the neighborhood of x_i with respect to error ranges on test set B is defined as

$$e_B(x_i) = \{x_j \in U | \forall a \in B, |a(x_i) - a(x_j)| \leqslant 2e(a)\}. \tag{6}$$

Naturally, $e_\emptyset(x_i) = U$ for any $x_i \in U$. We explain why $2e(a)$ instead of $e(a)$ was employed in Eq. (6) as the maximal distance. Let the true value of $x \in U$ be $a'(x)$ for $a \in B$. Due to the observational error, $a'(x) - e(a) \leqslant a(x_i) \leqslant a'(x) + e(a)$. In extreme cases, $a'(x) - e(a)$ and $a'(x) + e(a)$ can be the observed value of the same object. We have $(a'(x) + e(a)) - (a'(x) - e(a)) = 2e(a)$; therefore, observed values with no more than a difference of $2e(a)$ should be viewed the same.

An observed value $a(x)$ should have its real value $a'(x)$ in the range $[a(x) - e(a), a(x) + e(a)]$. Suppose that x_j falls in the neighborhood of x_i, from Definition 6, we obtain

$$|a'(x_i) - a'(x_j)| \leqslant 4e(a). \tag{7}$$

In other words, more objects than we expect are included in the error range. The following example gives an intuitive understanding.

Example 7. Suppose there are two plants with normalized sepal lengths 0.68 and 0.72, and the error range is ±0.01. In the worst case, observed values could be 0.69 and 0.71, respectively. Since a true sepal length of 0.70 can be read as either 0.69 or 0.71, the sepal lengths of these two plants have to be viewed the same. Here $0.72 - 0.68 = 0.04 = 4 \times 0.01$. Fig. 2 gives an illustration, where neighborhoods of x_2 have observed sepal lengths in [0.69,0.73]. This example indicates that the observational error is amplified.

From Definition 6 we also know that

$$e_B(x_i) = \bigcap_{a \in B} e_{\{a\}}(x_i). \tag{8}$$

That is, the neighborhood $e_B(x_i)$ is the intersection of a number of basic neighborhoods.

Sometimes we have a number of tests to obtain the same data item. Suppose some error ranges are known and others are unknown. The following proposition provides an estimation.

Proposition 8. *Let a_i and a_j be the tests for the same data item, $|a_j(x) - a_i(x)| \leqslant e'$ for any $x \in U$. We have*

$$e(a_j) \leqslant e(a_i) + e'. \tag{9}$$

Proof. Let $a'(x)$ be the true value of x on the test. $a_j(x) \leqslant a_i(x) + e' \leqslant a'(x) + (e(a_i) + e')$; $a_j(x) \geqslant a_i(x) - e' \geqslant a'(x) - (e(a_i) + e')$. Therefore $e(a_j) \leqslant e(a_i) + e'$. □

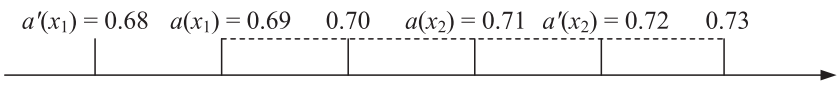

Fig. 2. Observational error amplification.

Unfortunately, even if we know that a_j is more precise than a_i, we cannot construct an error range narrower than $e(a_i)$. Therefore the error range is essentially amplified. In fact, this proposition is more often used to generate data. Most datasets from the UCI library provide only one test for each data item. For the purpose of experimentation, we are able to construct some new tests. Proposition 8 gives an approach.

One cannot decide the error range solely from the observed data. Moreover, given two tests for the same data item, one cannot tell which test has a narrower error range.

Remark 9. Let $a_i, a_j \in A$ be two tests for the same data item. For any $x \in U$, the relationship between $e(a_i)$ and $e(a_j)$ cannot be deduced from the test results.

Nevertheless, two tests for the same data item still meet certain constraints. One such constraint is given by the following proposition.

Proposition 10. *Let $a_i, a_j \in A$ be two tests for the same data item, for any $x \in U$,*

$$|a_i(x) - a_j(x)| \leqslant e(a_i) + e(a_j) \tag{10}$$

Proof. Let $a'(x)$ be the true value of x. $a'(x) - e(a_i) \leqslant a_i(x) \leqslant a'(x) + e(a_i)$, $a'(x) - e(a_j) \leqslant a_j(x) \leqslant a'(x) + e(a_j)$. Therefore Eq. (10) holds. □

However, the reverse of Proposition 10 does not hold. According to Proposition 8, even if two tests satisfy Eq. (10), they cannot be viewed as tests for the same data item.

The shapes of the neighborhoods are a line segment, a rectangle, and a cuboid for one-, two-, and three-dimensional spaces, respectively. One- and two-dimensional blocks are depicted in Figs. 3 and 4. Naturally, the size of the neighborhood depends on error ranges of tests, and more objects fall into the neighborhood of x_i for wider error ranges.

According to Definition 3, every object belongs to its own neighborhood. Consequently, for any $B \subseteq U$, we have

1. $\forall x \in U$, $e_B(x) \neq \emptyset$;
2. $\cup_{x \in U} e_B(x) = U$.

In other words, the family of neighborhoods $\{e_B(x_i) | x_i \in U\}$ forms a covering of the universe. This is formally given by the following theorem.

Theorem 11. *Let $S = (U, A, V, I, e)$ be an IS-ER and $B \subseteq A$. The set $\{e_B(x_i) | x_i \in U\}$ is a covering of U.*

This is why we call the model error-range-based covering rough set.

A neighborhood relation N_B induced by $B \subseteq A$ on the universe U can be written as a relation matrix $M(N_B) = (r_{ij})n \times n$, where

$$r_{ij} = \begin{cases} 1, & x_i \in e_B(x_j); \\ 0, & \text{otherwise}. \end{cases} \tag{11}$$

From Definition 6, we know that neighborhood relations are a kind of tolerance relations, which satisfy the properties of reflexivity and symmetry. Neighborhood relations draw the objects together for tolerance or indistinguishability from the viewpoint of observational error. Correlated test sets can induce correlated neighborhood relations, the following theorem indicates such a relationship.

Theorem 12. *Let $S = (U, A, V, I, e)$ be an IS-ER, N_B be a neighborhood relation induced by $B \subseteq A$. Given $B_1, B_2 \subseteq A$, we have*

$$N_{B_1 \cup B_2} = N_{B_1} \cap N_{B_2}. \tag{12}$$

Proof. Given $x_1, x_2 \in U$,

$N_{B_1 \cup B_2}(x_1, x_2) = 1$
$\Leftrightarrow \forall a \in B_1 \cup B_2$, $|a(x_i) - a(x_j)| \leqslant 2\ e(a)$.
$\Leftrightarrow \forall a \in B_1$, $|a(x_i) - a(x_j)| \leqslant 2e(a)$, and $\forall a \in B_2$, $|a(x_i) - a(x_j)| \leqslant 2e(a)$.
$\Leftrightarrow N_{B_1}(x_1, x_2) = 1$ and $N_{B_2}(x_1, x_2) = 1$.
This completes the proof. □

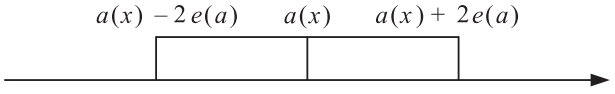

Fig. 3. One-dimensional neighborhood.

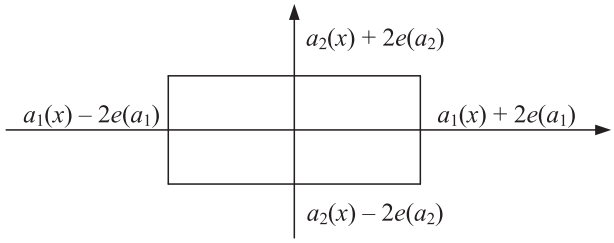

Fig. 4. Two-dimensional neighborhood.

3.2. Error-range-based covering rough set

In the following context, an error-range-based neighborhood will be called a *neighborhood* for simplicity. If all tests are error free, namely, $e(a) = 0$ for any $a \in A$, a neighborhood block degrades to an equivalence class. In this case, the objects in a neighborhood are equivalent to each other and the covering rough set model degenerates to that described by Pawlak. Therefore, the error-range-based covering rough set is a natural generalization of Pawlak's rough set.

Note that some properties of this model look the same as those appearing in [19,20]. This is due to the fact that both models deal with neighborhood. However, since the data models are different, these properties should be restudied. As will be pointed out later, some properties in [19] are no longer valid in the new model. Now we discuss some fundamental issues of rough set in the new model.

Definition 13. Let $S = (U, A, V, I, e)$ be an IS-ER, N_B be a neighborhood relation induced by $B \subseteq A$. We call $\langle U, N_B \rangle$ a neighborhood approximation space. For any $X \subseteq U$, two subsets of objects, called lower and upper approximations of X in $\langle U, N_B \rangle$, are defined as

$$\underline{N_B}X = \{x_i \in U | e_B(x_i) \subseteq X\}; \tag{13}$$

$$\overline{N_B}X = \{x_i \in U | e_B(x_i) \cap X \neq \emptyset\}. \tag{14}$$

Obviously, $\underline{N_B}X \subseteq X \subseteq \overline{N_B}X$. The boundary region of X in the approximation space is defined as $BN_BX = \overline{N_B}X - \underline{N_B}X$. Note that the neighborhood relation is always pertinent to the test set B. It cannot be independent as the one discussed in [19].

Generally, a finer covering (i.e., a covering with smaller blocks) is produced by a narrower error range, but a narrower error range does not necessarily produce a finer covering. Here, the corresponding statement similar to Theorem 2 in [19] does not hold. A counterexample is given below.

Example 14. Let $a_1, a_2 \in A$ be two tests for sepal length, $e(a_1) = 0.01$, and $e(a_2) = 0.02$, therefore a_1 has a narrower error range; a be an error-free test for sepal length, its existence is simply to indicate the true value of the data item; $x_1, x_2 \in U$ be two plants of interest. $a(x_1) = 0.70$, $a_1(x_1) = 0.69$, $a_2(x_1) = 0.72$, $a(x_2) = a_1(x_2) = a_2(x_2) = 0.67$.

According to Definition 6, $e_{\{a_1\}}(x_1) = \{x \in U | a_1(x) \in [0.67, 0.71]\}$, $e_{\{a_2\}}(x_1) = \{x \in U | a_2(x) \in [0.68, 0.76]\}$; $x_2 \in e_{\{a_1\}}(x_1)$ but $x_2 \notin e_{\{a_2\}}(x_1)$. In other words, with test a_1, x_2 is a neighbor of x_1; while with test a_2 it is not.

To produce a finer covering, we need a much narrower error range. We have the following theorem instead of Theorem 2 in [19].

Theorem 15. *Let $S = (U, A, V, I, e)$ be an information system with error ranges, $a_i, a_j \in A$ be two tests for the same data item where $e(a_i) \leqslant \frac{1}{3} e(a_j)$. We have*

1. $\forall x \in U$, $e_{\{a_i\}}(x) \subseteq e_{\{a_j\}}(x)$.
2. $\forall X \subseteq U$, $\underline{N_{\{a_i\}}}(X) \supseteq \underline{N_{\{a_j\}}}(X)$, $\overline{N_{\{a_i\}}}(X) \subseteq \overline{N_{\{a_j\}}}(X)$.

Proof. (1) Let the true value of x on the test be $a(x)$. $e_{\{a_i\}}(x) = \{x' \in U | a_i(x') \in [a_i(x) - 2e(a_i), a_i(x) + 2e(a_i)]$. Since $a(x) - e(a_i) \leqslant a_i(x) \leqslant a(x) + e(a_i)$,

$$[a_i(x) - 2e(a_i), a_i(x) + 2e(a_i)] \subseteq [a(x) - 3e(a_i), a(x) + 3e(a_i)]. \tag{15}$$

Alternatively, $e_{\{a_j\}}(x) = \{x' \in U | a_i(x') \in [a_j(x) - 2e(a_j), a_j(x) + 2e(a_j)]\}$. Since $a(x) - e(a_j) \leqslant a_j(x) \leqslant a(x) + e(a_j)$,

$$[a(x) - e(a_j), a(x) + e(a_j)] \subseteq [a_j(x) - 2e(a_j), a_j(x) + 2e(a_j)]. \tag{16}$$

Because $e(a_i) \leqslant \frac{1}{3} e(a_j)$, from Eqs. (15) and (16) we know that

$$[a_i(x) - 2e(a_i), a_i(x) + 2e(a_i)] \subseteq [a_j(x) - 2e(a_j), a_j(x) + 2e(a_j)]. \tag{17}$$

According to Definition 6, $e_{\{a_i\}}(x) \subseteq e_{\{a_j\}}(x)$.

(2) Assuming $e_{\{a_j\}}(x) \subseteq X$, we have $e_{\{a_i\}}(x) \subset X$. Therefore we have $x \in \underline{N_{\{a_i\}}}(X)$ if $x \in \underline{N_{\{a_j\}}}(X)$. However x is not necessarily in $\underline{N_{\{a_j\}}}(X)$ if we have $x \in \underline{N_{\{a_i\}}}(X)$. Hence $\underline{N_{\{a_i\}}}(X) \supseteq \underline{N_{\{a_j\}}}(X)$. Similarly, we have $\overline{N_{\{a_i\}}}(X) \subseteq \overline{N_{\{a_j\}}}(X)$. □

For the purpose of decision making, we are often interested in each class of the decision system.

Definition 16. Let $S = (U, C, D, V, I, e)$ be a DS-ER, and $X_1, X_2, \ldots, X_K$ be the object subsets with decisions 1 through K. The lower and upper approximations of decision D with respect to $B \subseteq C$ are defined as

$$\underline{N_B}D = \bigcup_{i=1}^{K} \underline{N_B}(X_i); \tag{18}$$

$$\overline{N_B}D = \bigcup_{i=1}^{K} \overline{N_B}(X_i). \tag{19}$$

The decision boundary region of D with respect to attributes B is defined as

$$BN_B D = \overline{N_B}D - \underline{N_B}D. \tag{20}$$

The lower approximation $\underline{N_B}D$ is also denoted by $POS_B(D)$. One can prove that

1. $\overline{N_B}D = U$;
2. $POS_B(D) \cup BN_B(D) = U$;
3. $POS_B(D) \cap BN_B(D) = \emptyset$.

In other words, the positive region contains objects that can be certainly classified into one class. The boundary region contains objects that can be classified into two or more classes. Given a test set B, the size of the positive region reflects the recognition power or characterizing power in the classification task. Therefore, the power, or more often called the significance, of B to approximate D is defined as follows:

Definition 17. Let $S = (U, C, D, V, I, e)$ be a DS-ER and $B \subseteq C$. The dependency degree of D to B is defined as

$$\gamma_B(D) = \frac{|POS_B(D)|}{|U|}, \tag{21}$$

where $|\bullet|$ is the cardinality of a set.

We say D completely depends on C and the DS-ER is consistent if $\gamma_C(D) = 1$; otherwise, we say D depends on C in the degree of $\gamma_C(D)$. Next, we discuss the issue in a finer granule. In one particular neighborhood of an object, there may exist some objects with different decision from the object. We propose the following definition to address this issue.

Definition 18. Let $S = (U, C, D, V, I, e)$ be a DS-ER, $B \subseteq C$ and $x \in U$. Any $y \in e_B(x)$ is called an inconsistent object in $e_B(x)$ if $D(y) \neq D(x)$. The set of inconsistent objects in $e_B(x)$ is

$$ic_B(x) = \{y \in e_B(x) | D(y) \neq D(x)\}. \tag{22}$$

The number of inconsistent objects, namely $|ic_B(x)|$, is important in evaluating the characteristics of the neighborhood block. It also influences the quality of rule induced by the block. From Definition 18 and Eq. (8) we know that

$$ic_B(x_i) = \bigcap_{a \in B} ic_{\{a\}}(x_i). \tag{23}$$

Now that we have presented key concepts of the new model, the following example can help to explain these.

Table 4
The neighborhood of objects on different test sets.

x	$\{a_1\}$	$\{a_1, a_2\}$	$\{a_1, a_2, a_3\}$	$\{a_1, a_2, a_3, a_4\}$
x_1	$\{x_1, x_2, x_3, x_5\}$	$\{x_1, x_3\}$	$\{x_1, x_3\}$	$\{x_1, x_3\}$
x_2	$\{x_1, x_2, x_3\}$	$\{x_2, x_3\}$	$\{x_2, x_3\}$	$\{x_2, x_3\}$
x_3	$\{x_1, x_2, x_3\}$	$\{x_1, x_2, x_3\}$	$\{x_1, x_2, x_3\}$	$\{x_1, x_2, x_3\}$
x_4	$\{x_4, x_6, x_9\}$	$\{x_4, x_6, x_9\}$	$\{x_4, x_6\}$	$\{x_4, x_6\}$
x_5	$\{x_1, x_5, x_8\}$	$\{x_5, x_8\}$	$\{x_5\}$	$\{x_5\}$
x_6	$\{x_4, x_6, x_7\}$	$\{x_4, x_6\}$	$\{x_4, x_6\}$	$\{x_4, x_6\}$
x_7	$\{x_6, x_7, x_8\}$	$\{x_7\}$	$\{x_7\}$	$\{x_7\}$
x_8	$\{x_5, x_7, x_8\}$	$\{x_5, x_8\}$	$\{x_8\}$	$\{x_8\}$
x_9	$\{x_4, x_9\}$	$\{x_4, x_9\}$	$\{x_9\}$	$\{x_9\}$

Example 19. Given the decision system with error ranges as indicated by Tables 1 and 2. Let a_1 = sepal-length, a_2 = sepal-width, a_3 = petal-length, a_4 = petal-width, and $D = \{d\} = \{\text{class}\}$. $e_B(x)$ is listed in Table 4, where B takes values listed as column headers, and x takes values listed in each row.

Furthermore, U is divided into a set of equivalence classes by d. $U/\{d\} = \{\{x_1,x_2,x_3\}, \{x_4,x_5,x_6\}, \{x_7,x_8,x_9\}\}$. In other words, based on the decision attribute, the objects are grouped into three subsets: $X_1 = \{x_1,x_2,x_3\}$, $X_2 = \{x_4,x_5,x_6\}$, and $X_3 = \{x_7,x_8,x_9\}$. $\underline{N_B}(X)$ is listed in the first part of Table 5, where B takes values listed as column headers, and X takes values listed in each row. Similarly, $\overline{N_B}(X)$ is listed in the second part of Table 5.

From Table 5, the positive regions and boundary regions of U on different test sets are $POS_{\{a_1\}}(\{d\}) = \{x_2,x_3\}$, $BN_{\{a_1\}}(\{d\}) = \{x_1,x_4,x_5,x_6,x_7,x_8,x_9\}$, $POS_{\{a_1,a_2\}}(\{d\}) = \{x_1,x_2,x_3,x_6,x_7\}$, $BN_{\{a_1,a_2\}}(\{d\}) = \{x_4,x_5,x_8,x_9\}$, $POS_{\{a_1,a_2,a_3\}}(\{d\}) = \{x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8,x_9\}$, $BN_{\{a_1,a_2,a_3\}}(\{d\}) = \emptyset$.

$\gamma_{\{a_1\}}(\{d\}) = \frac{2}{9}$, $\gamma_{\{a_1,a_2\}}(\{d\}) = \frac{5}{9}$, $\gamma_{\{a_1,a_2,a_3\}}(\{d\}) = 1$. Therefore $\{a_1,a_2,a_3\}$ has the same approximating power as C, and $\{d\}$ depends on $\{a_1,a_2,a_3\}$ completely.

3.3. The minimal-test-cost reduct problem

Attribute reduction has been intensively studied by the rough set society. There are many extensions of the classical rough set model [46], such as covering [71,72], decision-theoretical [32,66], variable-precision [73], and dominance-based [15,21] rough set models. In [59], to preserve more information of the decision system, a few fuzzy reducts are also required to produce fuzzy decision trees. A number of definitions of relative reducts exist [4,19,44,49] for different rough set models. Their relationships on consistent and inconsistent decision systems are closely studied [36]. Similar to the classical definition [44], we propose a new one based on the positive region in the new model.

Definition 20. Let $S = (U,C,D,V,I,e)$ be a DS-ER. Any $R \subseteq C$ is a decision-relative reduct iff:

1. $POS_R(D) = POS_C(D)$, and
2. $\forall a \in R, POS_{R-\{a\}} \subset POS_R(D)$.

The first condition guarantees that the information in terms of the positive region is preserved, and the second condition guarantees that no superfluous test is included. Therefore, a decision-relative reduct is a minimal test subset with the same approximating power as the whole test set. With this concept, decision-relative core is naturally defined as follows.

Definition 21. Let $Red(S)$ denote the set of all decision-relative reducts of a DS-ER S. $Core(S) = \cap Red(S)$ is called the core of S.

In other words, $Core(S)$ contains those tests appearing in all decision-relative reducts. A decision-relative reduct (core) is also called a reduct (core) for brevity. Definitions 20 and 21 have the same form as the classical one [43,44]. However, the data model has changed and the definition of positive region is different; hence, the computation is totally different.

Super-reducts [41,67] are important in the reduct constructing process. These are also useful in the new model and are therefore defined as follows.

Definition 22. Any $R^u \subseteq C$ is a decision-relative super-reduct iff $POS_{R^u}(D) = POS_C(D)$.

In other words, we can remove the second condition in Definition 20 to obtain this concept. We also have the following proposition, which could serve as an alternative definition of super-reducts.

Proposition 23. *Any $R^u \subseteq C$ is a decision-relative super-reduct iff $\exists R \in Red(S)$ st. $R \subseteq R^u$.*

From Definition 18 we know immediately that given $B \subseteq C$, $x \in POS_C(D)$ if and only if $ic_R(x) = \emptyset$. Consequently, we have the following proposition, which can be employed as an alternative definition of a reduct.

Proposition 24. *Let $S = (U,C,D,V,I,e)$ be a DS-ER. Any $R \subseteq C$ is a decision-relative reduct iff:*

1. *$\forall x \in POS_C(D)$, $ic_R(x) = \emptyset$, and*
2. *$\forall a \in R$, $\exists x \in POS_C(D)$, st. $ic_{R-\{a\}}(x) \neq \emptyset$.*

Table 5
Approximations of object subsets on different test sets.

	X	$\{a_1\}$	$\{a_1,a_2\}$	$\{a_1,a_2,a_3\}$	$\{a_1,a_2,a_3,a_4\}$
$\underline{N_B}(X)$	X_1	$\{x_2,x_3\}$	$\{x_1,x_2,x_3\}$	$\{x_1,x_2,x_3\}$	$\{x_1,x_2,x_3\}$
	X_2	$\emptyset$	$\{x_6\}$	$\{x_4,x_5,x_6\}$	$\{x_4,x_5,x_6\}$
	X_3	$\emptyset$	$\{x_7\}$	$\{x_7,x_8,x_9\}$	$\{x_7,x_8,x_9\}$
$\overline{N_B}(X)$	X_1	$\{x_1,x_2,x_3\}$	$\{x_1,x_2,x_3\}$	$\{x_1,x_2,x_3\}$	$\{x_1,x_2,x_3\}$
	X_2	$\{x_1,x_4,x_5,x_6,x_7,x_8,x_9\}$	$\{x_4,x_5,x_6,x_8,x_9\}$	$\{x_4,x_5,x_6\}$	$\{x_4,x_5,x_6\}$
	X_3	$\{x_4,x_5,x_6,x_7,x_8,x_9\}$	$\{x_4,x_5,x_7,x_8,x_9\}$	$\{x_7,x_8,x_9\}$	$\{x_7,x_8,x_9\}$

This proposition will help us in reduction algorithm designing, as will be discussed in Section 4. Sometimes we are interested in those reducts with minimal number of tests.

Definition 25. Let $Red(S)$ denote the set of all reducts of a DS-ER S. Any $R \in Red(S)$ where $|R| = \min\{|R'||R' \in Red(S)\}$ is called a *minimal reduct*.

The problem of finding a minimal reduct is called the minimal reduct problem, or the reduct problem [52] for brevity. In this work, we are interested in reducts with minimal test cost. Since TCI-DS-ER is a generalization of DS-ER, concepts in the latter model are also applicable to the former one. We propose the following concept.

Definition 26. Let $Red(S)$ denote the set of all reducts of a TCI-DS-ER $S = (U, C, D, V, I, e, c)$. Any $R \in Red(S)$ where $c(R) = \min\{c(R')|R' \in Red(S)\}$ is called a *minimal test cost reduct*.

The problem of finding such a reduct is called the *minimal-test-cost reduct problem*. Minimal-test-cost reducts are also called *optimal reducts* throughout the paper. Accordingly, reducts with test cost no much higher than the optimal one will be called *sub-optimal reducts*. Similar to the cases under other decision systems or neighborhood decision systems, the number of reducts is exponential with respect to the number of tests. The minimal-test-cost reduct problem under our new model is easier than the one studied in [37]. Therefore we need heuristic algorithms to deal with such problems. Before designing these algorithms, we should study the monotonicity of the dependency function.

Theorem 27 (*Type-1 monotonicity*). *Let $S = (U, C, D, V, I, e)$ be a DS-ER, $B_1 \subset B_2 \subseteq C$. We have*

1. $\forall x \in U$, $e_{B_1}(x) \supseteq e_{B_2}(x)$, $ic_{B_1}(x) \supseteq ic_{B_2}(x)$;
2. $N_{B_1} \supseteq N_{B_2}$;
3. $\forall X \subseteq U$, $\underline{N_{B_1}}(X) \supseteq \underline{N_{B_2}}(X)$;
4. $POS_{B_1}(D) \subseteq POS_{B_2}(D)$, $\gamma_{B_1}(D) \leqslant \gamma_{B_2}(D)$.

Proof

(1) is known immediately from Definitions 6 and 18.
(2) follows from Eq. (11); we know that if $r_{ij} = 1$ for $M(N_{B_2})$ then $r_{ij} = 1$ for $M(N_{B_1})$, although the reverse does not hold. Consequently, $N_{B_1} \supseteq N_{B_2}$.
(3) and (4) can be proved similarly. □

Theorem 27 shows that dependency increases monotonically with tests, which means that by adding new tests the dependency never decreases. Therefore we can employ the "depth-first-like" exhaustive algorithm [51] to construct optimal reducts. We can also adopt the addition, deletion, or addition-deletion strategies [67] to design our heuristic reduction algorithms.

3.4. Evaluation metrics

For brevity, the minimal-test-cost reduct will be called from hereon the *optimal reduct*. We can design many algorithms to deal with the minimal-test-cost reduct problem. Consequently, we need a number of metrics to evaluate the performance of these algorithms. We adopt the four metrics proposed in [37] for this purpose. These are *finding optimal* factor, *exceeding* factor, *maximal exceeding* factor, and *average exceeding* factor.

Given K datasets (TCS-DS-ERs), an attribute reduction algorithm A produces exactly one reduct for each dataset. Suppose that k out of K reducts are optimal for respective datasets. The *finding optimal factor* (FOF) is defined as

$$op = \frac{k}{K}. \tag{24}$$

This metric is both qualitative and quantitative. First, it only counts optimal solutions. Second, it is computed statistically on K datasets. In our experiments, we generated different test costs for the same DS-ER to produce different TCS-DS-ERs. Therefore we have enough data to obtain the finding optimal factor for statistics purposes.

For a TCS-DS-ER, let R' be an optimal reduct. The exceeding factor of a reduct R is

$$ef(R) = \frac{c(R) - c(R')}{c(R')}. \tag{25}$$

The exceeding factor provides a quantitative metric to evaluate the performance of a reduct. It indicates the badness of a reduct when it is not optimal. Naturally, if R is an optimal reduct, the exceeding factor is 0.

Suppose again the algorithm A is run on K datasets. On the ith dataset ($1 \leqslant i \leqslant K$), the reduct produced by the algorithm is denoted R_i. The *maximal exceeding factor* (MEF) is defined as

$$\max_{1 \leqslant i \leqslant K} ef(R_i). \tag{26}$$

This shows the worst case of the algorithm given some data set. Although it relates to the performance of one particular reduct, it should be viewed as a statistical rather than an individual metric.

The *average exceeding factor* (AEF) is defined as

$$\frac{\sum_{i=1}^{K} ef(R_i)}{K}. \tag{27}$$

Since it is averaged on K different test-cost-sensitive decision systems, it shows the overall performance of the algorithm from solely a statistical perspective.

4. Algorithms

As mentioned in the last section, the minimal-test-cost reduct problem is more complex than the traditional reduct problem [43,52]. Hence heuristic algorithms are needed to find sub-optimal reducts for large datasets. To evaluate the performance of a heuristic algorithm in terms of the quality of the solution, we should find an optimal reduct first. Consequently, exhaustive algorithms are also needed.

This section presents for the new problem both exhaustive and heuristic algorithms: the exhaustive is based on backtracking where pruning techniques are crucial in reducing computation, whereas the heuristic has a framework similar to that proposed in [37], though the algorithm is totally different due to the new data model. We also present the competition approach [37], which is still valid for the new environment to enhance heuristic algorithms.

4.1. The backtrack reduction algorithm

A general algorithm for finding all (or some) solutions to some computational problem is backtracking, which we employ to deal with minimal-test-cost reduct problems. As a general approach, we start from the empty set, and add tests one by one until a super-reduct is obtained. Then we backtrack to a former step and obtain other super-reducts. This process can be illustrated through a state space tree with exactly $2^{|C|}$ nodes, each corresponding to a test subset. Therefore one can traverse the tree in preorder and find the optimal solution in $2^{|C|}$ steps. Example 28 illustrates the case for 4 tests.

Example 28. Let $C = \{a_1, a_2, a_3, a_4\}$. The state space tree is depicted in Fig. 5. The traversal of the tree produces test subsets in the following sequence: $\emptyset$, $\{a_1\}$, $\{a_1, a_2\}$, $\{a_1, a_2, a_3\}$, $\{a_1, a_2, a_3, a_4\}$, $\{a_1, a_2, a_4\}$, $\{a_1, a_3\}$, $\{a_1, a_3, a_4\}$, $\{a_1, a_4\}$, $\{a_2\}$, $\{a_2, a_3\}$, $\{a_2, a_3, a_4\}$, $\{a_2, a_4\}$, $\{a_3\}$, $\{a_3, a_4\}$, $\{a_4\}$.

This approach is, however, rather time consuming and unacceptable in most applications. The key issue in performing this algorithm is how to prune the state space tree, or in other words, knowing when to stop searching a subtree if no better solution exists inside. This section discusses this issue and presents three techniques for pruning.

If a test is redundant, we need not choose it in the process of reduct constructing. This consideration alludes to the first technique in pruning the state space tree. Formally, given a test set $B \subset C$ and a test $a \in C - B$, if a is redundant with respect to B, there does not exist a reduct R such that $B \cup \{a\} \subseteq R$. It is, however, not straightforward to tell whether or not a test is *redundant*. Intuitively, one may expect that if $POS_{B \cup \{a\}}(D) = POS_B(D)$, a is redundant with respect to B. The following example shows that this condition is insufficient.

Example 29. A DS-ER is given by Table 6 and $e(a_1) = e(a_2) = 0.1$. $POS_{\emptyset \cup \{a_1\}}(D) = POS_{\emptyset}(D) = \emptyset$. However, a_1 should not be viewed redundant with respect to $\emptyset$. This is because $POS_{\{a_1, a_2\}}(D) = U$, and $\{a_1, a_2\}$ is a reduct.

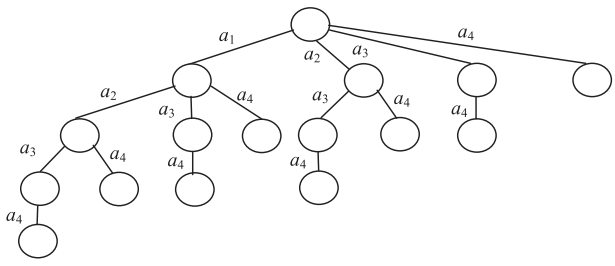

Fig. 5. The state space tree for $C = \{a_1, a_2, a_3, a_4\}$.

From Example 29, we observe the following. When the test set is $\emptyset$, all objects should be viewed neighbors. Therefore $e_\emptyset(x_1) = U$. However, when a_1 is added to the test set, we have $e_{\{a_1\}}(x_1) = \{x_1, x_2\} \subset U$. In other words, a_1 is useful in decreasing the neighborhood. In fact, if a test does not help decrease any neighborhood, it is definitely redundant. This is given by the following lemma.

Lemma 30. *Let $S = (U, C, D, V, I, e)$ be a DS-ER, $B \subset C$ and $a \in C - B$. If $\forall x \in POS_C(D) - POS_B(D)$,*

$$e_B(x) = e_{B\cup\{a\}}(x), \tag{28}$$

any B' where $B \cup \{a\} \subseteq B' \subseteq C$ is not a reduct.

In the lemma, we only consider neighborhoods of $x \in POS_C(D) - POS_B(D)$. This is because, by Definition 20, we only need to preserve the positive region, and by Theorem 27 objects already in the positive region $POS_B(D)$ do not need to be reconsidered. From the viewpoint of classification, the aim of choosing more tests is to remove inconsistent objects from neighborhood blocks. Therefore it is important whether or not inconsistent objects are removed. We then have the following theorem.

Theorem 31. *Let $S = (U, C, D, V, I, e)$ be a DS-ER, $B \subset C$ and $a \in C - B$. If $\forall x \in POS_C(D)$,*

$$ic_B(x) = ic_{B\cup\{a\}}(x), \tag{29}$$

any B' where $B \cup \{a\} \subseteq B' \subseteq C$ is not a reduct.

Proof. Given B' where $B \cup \{a\} \subseteq B' \subseteq C$. If $POS_{B'}(D) \neq POS_C(D)$, B' is not a reduct. Now suppose $POS_{B'}(D) = POS_C(D)$ and let $x \in POS_{B'}(D)$. From Eqs. (23) and (29) we know that $ic_{B'-\{a\}}(x) = ic_{B'-\{a\}-B}(x) \cap ic_B(x) = ic_{B'-\{a\}-B}(x) \cap ic_{B\cup\{a\}}(x) = ic_{B'}(x) = \emptyset$. According to Proposition 24, B' is not a reduct. □

Note that, unlike Lemma 30, we do not require $x \in POS_C(D) - POS_B(D)$. This is because $\forall x \in POS_B(D)$, $ic_B(x) = ic_{B\cup\{a\}}(x) = \emptyset$. Since $e_B(x) = e_{B\cup\{a\}}(x)$ gives $ic_B(x) = ic_{B\cup\{a\}}(x)$, Lemma 30 is also proved.

Theorem 31 indicates which kind of tests are definitely redundant and therefore can be ignored in the process of reduct constructing. With Theorem 31, we propose a backtrack algorithm to obtain one optimal reduct of a TCI-DS-ER. The algorithm is listed in Algorithm 1. Before running the algorithm, we set $B = \emptyset$, $U' = POS_C(D)$, $R = C$, and $l = 0$.

Algorithm 1. A backtrack algorithm to one optimal reduct

Input: $S = (U, C, D, V, I, e, c)$, selected tests B, last level boundary objects U', current level test index lower bound l.
Output: An optimal reduct is stored in the global variable R.
Method: backtrack

```
1: for (i = l; i < |C|; i ++) do
2:   if (c(B ∪ {a_i}) ⩾ c(R)) then
3:     continue;//Not cheaper, no need to check, prune
4:   end if
5:   lessInconsistent = false;//Less inconsistent objects induced by a_i?
6:   U'' = ∅;//Current level boundary objects
7:   for (each x ∈ U') do
8:     if (ic_{B∪{a_i}}(x) ⊂ ic_B(x)) then
9:       lessInconsistent = true;
10:      if (ic_{B∪{a_i}}(x) ≠ ∅) then
11:        U'' = U'' ∪ {x};
12:      end if
13:    end if
14:  end for
15:  if (lessInconsistent == false) then
16:    continue;//a_i is not useful, prune
17:  else
18:    if (U'' = ∅) then
19:      R = B ∪ {a_i};//A better super-reduct
20:      continue;
21:    else
22:      backtrack (U'', B ∪ {a_i}, i + 1);//Next level
23:    end if
24:  end if
25: end for
```

Table 6
The decision system for the counter-example.

U	a_1	a_2	d
x_1	0.41	0.81	Y
x_2	0.42	0.42	N
x_3	0.72	0.82	N
x_4	0.73	0.41	Y

The first prune technique is implemented through the variable 'lessInconsistent' in the algorithm. Now we discuss the other two techniques to prune the state space tree. R is employed to record currently best super-reduct. If another set $R' \subseteq C$ does not have less test cost, there is no need to check R'. This technique is implemented in lines 2 through 4 of the algorithm.

Finally, tests are chosen in a bottom-up manner. That is, after including a_i in B, the algorithm will not include any a_j, where $j < i$, in a deeper level. This technique cannot only prune the tree, but also guarantees the correctness of the algorithm. In the process of running the algorithm, R might not be a reduct. Because $U'' = \emptyset$, R must be a super-reduct. That is, it could be a reduct, or a superset of a reduct. However, when the algorithm finishes R must be a reduct. This is because the cost of each test is non-negative, hence a super-reduct will generally be replaced by a reduct. The reduct might be a subset of the super-reduct, or another reduct with lower test cost.

4.2. The λ-weighted heuristic reduction algorithm

To design a heuristic algorithm, we employ an algorithm framework very similar to the one proposed in [37]. The algorithm is listed in Algorithm 2. It follows the typical addition-deletion strategies [67]. It constructs a super-reduct, then reduces it to obtain a reduct. Core attributes are not computed since it is time consuming to obtain them in the new model. The algorithm is essentially different from the one in [37] for the following reasons. First, the input S is a TCI-DS-ER instead of a TCI-DS, and test results are numerical rather than nominal. Second, the computation of positive regions is totally different.

Algorithm 2. A general test-cost-sensitive reduction algorithm

Input: $S = (U, C, D, V, I, e, c)$
Output: A reduct with sub-minimal test cost
Method: tcs-reduction
1: $B = \emptyset$;
  //Addition
2: $CA = C$;
3: **while** $(POS_B(D) \neq POS_C(D))$ **do**
4: For any $a \in CA$, compute $f(B, a, c)$;
5: Select a' with the maximal $f(B, a', c)$;
6: $B = B \cup \{a'\}$; $CA = CA - \{a'\}$;
7: **end while**
  //Deletion
8: $CD = B$; sort attributes in CD according to respective test cost in a descending order;
9: **while** $(CD \neq \emptyset)$ **do**
10: $CD = CD - \{a'\}$, where a' is the first element of CD;
11: **if** $(POS_{B-\{a'\}}(D) = POS_B(D))$ **them**
12: $B = B - \{a'\}$;
13: **end if**
14: **end while**
15: return B;

Lines 4 and 5 contain the key code of this framework. One can design different attribute significance functions to obtain respective algorithms. As discussed through Example 29, a positive region is not good heuristic information to the new problem; nor is information gain [50], which has been often employed as the heuristic information for attribute reduction (see, e.g., [37,58]), directly applicable.

According to Proposition 24 and Theorem 31, we know that $|ic_B(x)|$ is useful in evaluating the quality of a neighborhood block. Therefore, we propose the following concepts.

Definition 32. Let $S = (U, C, D, V, I, e)$ be a DS-ER, $B \subseteq C$ and $x \in U$. The number of inconsistent objects in neighborhood $e_B(x)$ is $|ic_B(x)|$. The total number of such objects with respect to U is

$$nc_B(S) = \sum_{x \in U} |ic_B(x)|, \tag{30}$$

and with respect to the positive region is

$$pc_B(S) = \sum_{x \in POS_C(D)} |ic_B(x)|. \tag{31}$$

According to Definitions 22 and 32, we know that B is a super-reduct if and only if $pc_B(S) = 0$. Therefore we can replace the condition $POS_B(D) \neq POS_C(D)$ in Line 3 of the algorithm with $pc_B(S) > 0$. This is easier to implement from the algorithmic viewpoint. Finally, we propose the following λ-weighted attribute significance function:

$$f(B, a_i, c(a_i)) = (|pc_B(S)| - |pc_{B \cup \{a_i\}}(S)|) \times c(a_i)^{\lambda}, \tag{32}$$

where c_i is the test cost of a_i, and $\lambda \leqslant 0$ is a user-specified parameter. Basically, the idea of introducing an exponential λ is the same as [37]. If $\lambda = 0$, test costs are essentially not considered; whereas if $\lambda < 0$, tests with lower cost have bigger significance.

Note that this attribute significance function fails if there are free tests, namely, tests with $c(a) = 0$. We can set the cost to a small value (e.g., 0.01) to fix this problem. This issue seldom arises in applications because data collection and storing always introduces certain costs. Therefore it is also reasonable to assume that free tests do not exist.

4.3. The competition approach

This approach has been discussed in [37] to obtain better results with more run-time. It is still valid in the new environment because there is no universally optimal λ. Formally, let R_λ be the reduct constructed by Algorithm 2 using the exponential λ, and L be the set of user-specified λ values. The minimal test cost

$$c_L = \min_{\lambda \in L} c(R_\lambda) \tag{33}$$

can be obtained using all λ values in L.

This approach requires the algorithm to run $|L|$ times with different λ values. Since the heuristic algorithm is fast, it is acceptable for relatively small $|L|$. One can also run the program on $|L|$ different computers in parallel. As will be shown in Section 5.4, this simple approach can enhance the quality of the result significantly.

5. Experiments

In this section, we try by experimentation to answer the following questions; the first concerns the backtrack algorithm, and three others the heuristic algorithm.

1. Is the backtrack algorithm efficient?
2. Is the heuristic algorithm appropriate for the problem?
3. Is there an optimal setting of λ for any dataset?
4. Can the competition approach improve the quality of the result?

5.1. Data generation

We are interested in datasets where test costs and error ranges are available. These do exist in applications such as clinic systems and hydrology systems, among others. Unfortunately, such datasets are not represented in the UCI library. Since the main objective of this work is to study the performance of the reduction algorithm, rather than analyze the data for one particular application, we will create some data for experimentation. In this way, different parameters can be specified and different data distributions can be employed. Consequently, the reduction algorithm can be extensively studied. Unlike in simpler models, data should not be randomly generated, but meet certain constraints. For example, for the same data item, tests with narrower error ranges should be more expensive. In this subsection, we will discuss both the process and substantial settings the generation of the data. Constraints mentioned above are met in this process.

First, we choose ten datasets from the UCI library, as listed in Table 7. In this way, the datasets, are not totally artificial, having certain application domains. Each dataset should contain exactly one decision attribute, and no missing value should exist. To make the data easier to handle, data items are normalized onto [0,1]. Missing values are directly set to 0.5.

Second, we produce the number of additional tests for one particular data item. We use the uniform distribution generator [37] to produce integers between 0 and k. That is, for each data item, we have 1 to $(k + 1)$ measurement methods to obtain it; k is set to less than 5 in our experiments. In real applications, one may find data that can be obtained through more than 5 different approaches. However, commonly used ones are seldom more than 5. The number of tests for our experiments is $|C'|$ in Table 7.

Table 7
Dataset information.

No.	Name	Domain	$\|U\|$	$\|C\|$	$\|C'\|$	$D = \{d\}$
1	Iris	Zoology	150	4	10	Class
2	Glass	Manufacture	214	9	29	Type
3	Wine	Agriculture	178	13	36	Class
4	Wpbc	Clinic	198	33	50	Outcome
5	Wdbc	Clinic	569	30	56	Diagnosis
6	Credit	Commerce	690	15	26	Class
7	Image	Graphics	210	19	34	Class
8	Iono	Physics	351	34	62	Class
9	Liver	Clinic	345	6	17	Selector
10	Diab	Clinic	768	8	15	Class

Third, we produce the error ranges for each test. Since data are normalized, we set the error ranges to be $\pm 0.005i$ for the ith method. That is, we assume the error range of the original data is ± 0.005. This assumption may not hold for the data. However, unless specified by the original creator, the true error range is never known. The specified error ranges are reasonable in applications, a data with an error more than 0.025 (the maximal observational error of the 5th method) seems not quite useful. For such data, from Example 7, we know that two real values with 0.1 difference could be viewed the same.

Fourth, we produce "new" data subject to error ranges. Let a_1 be the original (first) test, according to Proposition 8, we can add a random number in $[-0.005(i-1), [-0.005(i-1)]$ to $a_1(x)$ to produce $a_i(x)$, where $x \in U$. The number is generated by the uniform distribution generator [37]. In this way, a_i is the new test with error rage $\pm 0.005i$. Note that if a datum smaller than 0 (or larger than 1) is produced, it is set to 0 (or 1) directly so as not to exceed the boundary.

Fifth, we produce test costs, which are always represented by positive integers. Let a_1 be the original (first) test and a_l is the last test for one particular data item. $c(a_l)$ is set to a random number in [1,100] subject to the uniform distribution. $c(a_i)$ where $1 \leqslant i < l$ is set to $2 \times c(a_{i+1})$. This setting guarantees that tests with narrower error ranges are more expensive.

An example dataset generated by this approach is listed in Tables 8 and 9. We generate one DS-ER from each dataset, and many TCS-DS-ERs from each DS-ER by setting different test costs. Note that by employing this approach we can generate as many datasets as we need.

5.2. Efficiency of the backtrack algorithm

We study the efficiency of Algorithm 1 using two metrics. One is the number of steps the program performs on the state space tree, i.e., the number of times the backtrack method is invoked. This metric is used to study the effectiveness of the pruning techniques. The other is the run-time compared with heuristic algorithms. Specifically, we employ Algorithm 2 for the comparison, where λ is set to -1.

For each dataset listed in Table 7, experiments are undertaken with 100 different test cost settings. The state space tree size and the number of steps for Algorithm 1 are listed in Table 10. The average and maximal run time for both algorithms are depicted in Fig. 6, where the unit of run-time is 1 ms.

From the results we note the following:

1. With the pruning techniques, the number of steps is very small compared with the state space tree size. Therefore the pruning techniques are very effective.

Table 8
A decision system generated from Iris.

Plant	SL	SL-1	SL-2	SW	PL	PL-1	PL-2	PW	PW-1	PW-2	Class
x_0	0.23529	0.23274	0.22699	0.63636	0.09524	0.09969	0.10144	0.04762	0.04447	0.04852	Setosa
x_1	0.17647	0.17657	0.17267	0.40909	0.09524	0.09599	0.10214	0.04762	0.04782	0.04662	Setosa
x_2	0.11765	0.11945	0.12345	0.50000	0.07143	0.06763	0.07303	0.04762	0.04327	0.05402	Setosa
...											
x_{50}	0.79412	0.79747	0.79382	0.50000	0.54762	0.54777	0.55512	0.47619	0.48054	0.46699	Versicolor
x_{51}	0.61765	0.61405	0.61985	0.50000	0.50000	0.50350	0.49920	0.52381	0.52566	0.52851	Versicolor
x_{52}	0.76471	0.76861	0.76181	0.45455	0.59524	0.59209	0.60464	0.52381	0.52301	0.53351	Versicolor
...											
x_{147}	0.64706	0.64616	0.65396	0.40909	0.66667	0.66607	0.66427	0.76190	0.76645	0.7583	Virginica
x_{148}	0.55882	0.56102	0.56172	0.59091	0.71429	0.71079	0.71709	0.90476	0.90316	0.91366	Virginica
x_{149}	0.47059	0.47314	0.46729	0.40909	0.64286	0.64006	0.64186	0.66667	0.67037	0.66647	Virginica

SL stands for sepal length, SW stands for sepal width, PL stands for petal length, and PW stands for petal width.
1 and 2 after SL, PL, and PW indicate different revision of the original data.
There is only one method to obtain SW.

Table 9
A generated observation error vector and a generated test cost vector.

a	SL	SL-1	SL-2	SW	PL	PL-1	PL-2	PW	PW-1	PW-2
$e(a)$	0.005	0.010	0.015	0.005	0.005	0.010	0.015	0.005	0.010	0.015
$c(a)$	52	26	13	80	372	186	93	360	180	90

Table 10
Number of steps for Algorithm 1.

Dataset	Tree size	Minimal steps	Maximal steps	Average steps
Iris	2^{10}	16	58	34
Glass	2^{29}	7702	218,972	56,591
Wine	2^{36}	10	446	82
Wpbc	2^{50}	19	345	106
Wdbc	2^{56}	63	3604	484
Credit	2^{26}	1004	141,868	16,274
Image	2^{34}	44	7551	907
Iono	2^{62}	129	6471	1023
Liver	2^{17}	194	1965	855
Diab	2^{15}	129	1146	462

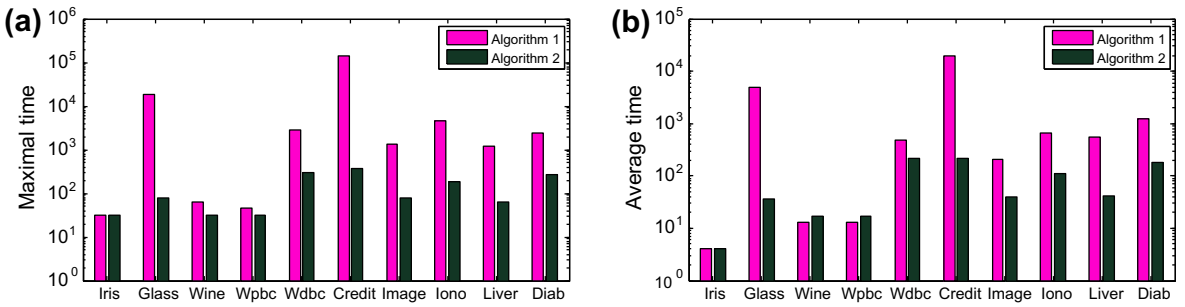

Fig. 6. Run time comparison: (a) maximal time, (b) average time.

2. In about half the datasets tested, the backtrack algorithm is comparable to, or at least not significantly worse than, the heuristic algorithm. This is partly due to the fact that these datasets are not big.
3. The time expenditure does not simply rely on the dataset size. Glass and Credit are much smaller than Wdbc; however, the run-times are longer.
4. he heuristic algorithm is more stable in terms of run-time. For example, on the Credit dataset, the maximal run-time is only 1.7 times of the average one for Algorithm 2. In contrast, it is 7.3 times for Algorithm 1.

In summary, for small or medium sized datasets, the backtrack algorithm is a good choice to obtain the optimal reduct. For large datasets, however, the heuristic algorithm must be employed.

5.3. Effectiveness of the heuristic algorithm

We let $\lambda = 0, -0.25, -0.5, \ldots, -2$. The algorithm runs 4000 times with different test cost settings with each λ setting on all datasets except Glass. The backtrack algorithm takes rather a long time on Glass; hence, we only perform it 500 times. Results are depicted in Figs. 7–9. Data for $\lambda = 0$ are not included in figures because respective results are incomparable to others. We will discuss this issue in more detail in Section 5.4.

From the results we observe the following:

1. The quality of the results varies for different datasets. It is not simply related to the size of the dataset.
2. The quality of the results are not satisfactory in terms of the finding optimal factor. However, the average exceeding factor is less than 0.1 in most cases. In other words, the results are acceptable.
3. There is no universally optimal setting of λ. $\lambda = -0.5$ might be a rational setting if no further information is available.

Although the results are generally acceptable, the performance of the algorithm should be improved. This issue will be discussed further in Section 5.4.

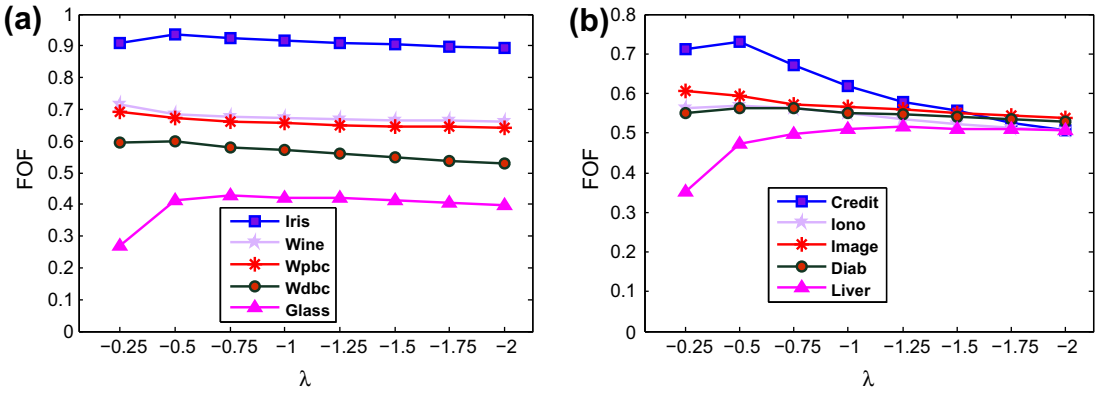

Fig. 7. Finding optimal factor: (a) datasets 1–5, (b) datasets 6–10.

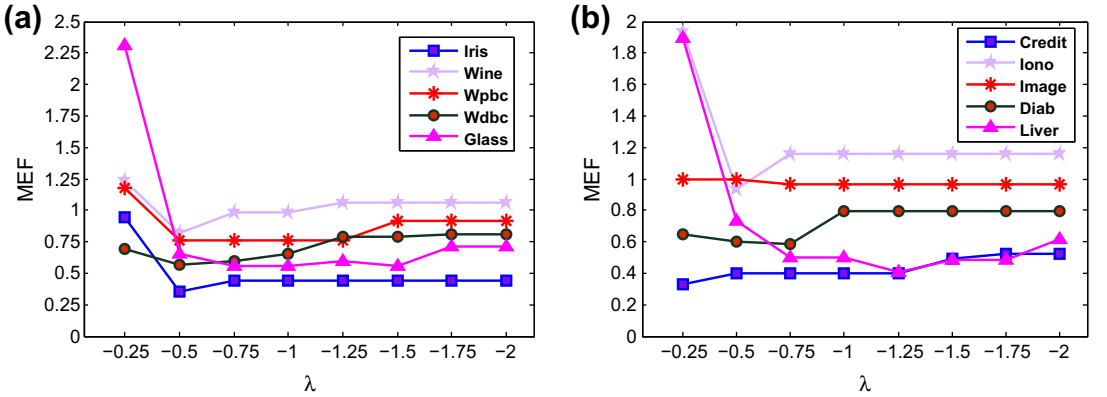

Fig. 8. Maximal exceeding factor: (a) datasets 1–5, (b) datasets 6–10.

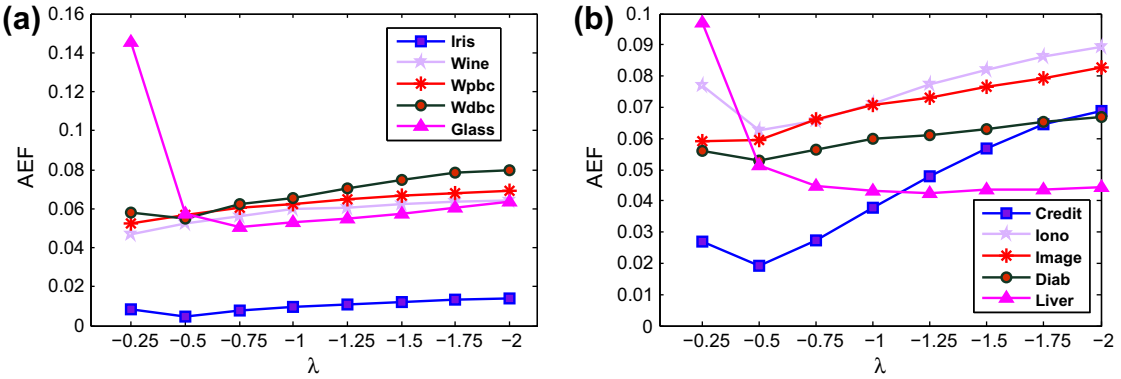

Fig. 9. Average exceeding factor: (a) datasets 1–5, (b) datasets 6–10.

5.4. Comparison of three approaches

Here we compare the performance of the three approaches mentioned in Section 4. All three are based on Algorithm 2. The first approach, called the non-weighting approach, is implemented by setting $\lambda = 0$. The second approach, called the best λ approach, chooses the best λ value as depicted in Figs. 7–9. The third approach is the competition approach discussed in Section 4.3. $L = \{0, -0.25, -0.5, \ldots, -2\}$.

Table 11 lists results for all three approaches. We observe the following:

Table 11
Results for $\lambda = 0$, λ with the optimal setting, and λ with a number of choices.

Dataset	Finding optimal factor			Maximal exceeding factor			Average exceeding factor		
	$\lambda = 0$	$\lambda = \lambda^*$	$\lambda \in L$	$\lambda = 0$	$\lambda = \lambda^*$	$\lambda \in L$	$\lambda = 0$	$\lambda = \lambda^*$	$\lambda \in L$
Iris	0.0000	0.9375	0.9685	121.00	0.3511	0.2435	4.2058	0.0048	0.0019
Glass	0.0000	0.4270	0.6510	32.851	0.5526	0.5526	3.2732	0.0508	0.0219
Wine	0.0000	0.7138	0.7810	465.00	0.8182	0.6800	10.351	0.0466	0.0320
Wpbc	0.0000	0.6928	0.7543	70.500	0.7647	0.5000	9.2205	0.0526	0.0371
Wdbc	0.0000	0.5995	0.7305	45.125	0.5714	0.4583	6.3530	0.0548	0.0295
Credit	0.0000	0.7300	0.9020	4.1842	0.3974	0.1845	0.4018	0.0191	0.0050
Image	0.0000	0.6070	0.7132	32.428	1.0000	0.6000	2.1375	0.0593	0.0341
Iono	0.0000	0.5700	0.7170	90.833	0.9333	0.5806	11.482	0.0626	0.0313
Liver	0.0000	0.5155	0.5990	9.5747	0.4103	0.3383	1.8969	0.0423	0.0274
Diab	0.0000	0.5642	0.6892	9.4706	0.6018	0.4915	1.6631	0.0529	0.0300

1. The non-weighting approach never finds the optimal reduct. It is unacceptable from all three metrics. This is because without considering test costs, the algorithm tends to choose tests with narrower error ranges, which are more expensive.
2. The competition approach significantly improves the quality of results especially in datasets where the optimal reduct is hard to find. For example, on the Glass dataset, the finding optimal factor of the competition approach is 0.6510, 0.224 better than the best λ setting.

In general, the competition approach is a simple and effective method to improve the performance of the algorithm. From the application point of view, it is also much easier to specify L than to guess the best λ. We can specify as many λ values as we want, as long as run-times are acceptable.

6. Conclusions and further works

In this paper, we addressed the TARER problem covering data model, computational model, problem and algorithm. Experimental results indicate the efficiency of the backtrack algorithm, the effectiveness of the λ-weighting heuristic algorithm, and the significance of the competition approach in terms of performance improvement. In the future, much work needs to be undertaken at four levels:

1. At the data model level, new data models addressing common-test-cost [38], different error ranges, and other types of uncertainty [1,7] can be built. These models are generally more complex than that presented in this work. These could have some interesting areas of application.
2. At the computational model level, error-range-based covering rough sets deserve more investigation. This paper only considered concepts closely related to attribute reduction. Other aspects of the theory should be studied.
3. At the problem level, other problems such as rule synthesis should be defined. Moreover, with different data models, attribute reduction problems should also be reconsidered.
4. At the algorithm level, other algorithms should be developed. One could design discernibility-matrix-based exhaustive algorithms [52], entropy-based heuristic algorithms [34,54,57], genetic algorithms [31,61], or set-covering-based algorithms [8].

In summary, this study suggests new research trends concerning covering rough set theory, the attribute reduction problem, and cost-sensitive learning applications.

Acknowledgements

We are grateful to the anonymous reviewers for their valuable comments and suggestions. This work is in part supported by National Science Foundation of China under Grant No. 61170128, the Natural Science Foundation of Fujian Province, China under Grant No. 2011J01374 and the Education Department of Fujian Province under Grant No. JA11176.

References

[1] C.C. Aggarwal, On density based transforms for uncertain data mining, in: Proceedings of IEEE 23rd International Conference on Data Engineering, 2007, pp. 866–875.
[2] A. Bargiela, W. Pedrycz, Granular Computing: An Introduction, Kluwer Academic Publishers, Boston, 2002.
[3] W. Bartol, J. Miro, K. Pioro, F. Rossello, On the coverings by tolerance classes, Information Sciences 166 (1–4) (2004) 193–211.
[4] J.G. Bazan, A. Skowron, Dynamic reducts as a tool for extracting laws from decision tables, in: Proceedings of the 8th International Symposium on Methodologies for Intelligent Systems, 1994, pp. 346–355.
[5] D. Bianucci, G. Cattaneo, D. Ciucci, Entropies and co-entropies of coverings with application to incomplete information systems, Fundamenta Informaticae 75 (1–4) (2007) 77–105.

[6] X.Y. Chai, L. Deng, Q. Yang, C.X. Ling, Test-cost sensitive Naïve Bayes classification, in: Proceedings of the 5th International Conference on Data Mining, 2004, pp. 51–58.
[7] M. Chau, R. Cheng, B. Kao, J. Ng, Uncertain data mining: an example in clustering location data, in: Proceedings of Advances in Knowledge Discovery and Data Mining, LNCS, vol. 3918, 2006, pp. 199–204.
[8] V. Chvatal, A greedy heuristic for the set-covering problem, Mathematics of Operations Research 4 (3) (1979) 233–235.
[9] T.M. Cover, Nearest neighbor pattern classification, IEEE Transactions on Information Theory 13 (1967) 21–27.
[10] J. Dai, Rough 3-valued algebras, Information Sciences 178 (8) (2008) 1986–1996.
[11] J. Dai, Q. Xu, Approximations and uncertainty measures in incomplete information systems, Information Sciences 198 (1) (2012) 62–80.
[12] M. Diker, Textural approach to generalized rough sets based on relations, Information Sciences 180 (8) (2010) 1418–1433.
[13] J. Du, Z.H. Cai, C.X. Ling, Cost-sensitive decision trees with pre-pruning, in: Proceedings of Canadian AI, No. 4509 in LNAI, 2007, pp. 171–179.
[14] Y. Du, Q. Hu, P. Zhu, P. Ma, Rule learning for classification based on neighborhood covering reduction, Information Sciences 181 (24) (2011) 5457–5467.
[15] S. Greco, B. Matarazzo, R. Slowinski, J. Stefanowski, Variable consistency model of dominance-based rough sets approach., in: Proceedings of Rough Sets and Current Trends in Computing, LNCS, vol. 2005, 2000, pp. 170–181.
[16] H. He, F. Min, Accumulated cost based test-cost-sensitive attribute reduction, in: Proceedings of the 13th International Conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing, LNAI, vol. 6743, 2011, pp. 244–247.
[17] H. He, F. Min, W. Zhu, Attribute reduction in test-cost-sensitive decision systems with common-test-costs, in: Proceedings of the 3rd International Conference on Machine Learning and Computing, vol. 1, 2011, pp. 432–436.
[18] Q. Hu, W. Pedrycz, D. Yu, J. Lang, Selecting discrete and continuous features based on neighborhood decision error minimization, IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics 40 (1) (2010) 37–50.
[19] Q. Hu, D. Yu, J. Liu, C. Wu, Neighborhood rough set based heterogeneous feature subset selection, Information Sciences 178 (18) (2008) 3577–3594.
[20] Q. Hu, D. Yu, Z. Xie, Numerical attribute reduction based on neighborhood granulation and rough approximation (in chinese), Journal of Software 19 (3) (2008) 640–649.
[21] B. Huang, H.-X. Li, D.-K. Wei, Dominance-based rough set model in intuitionistic fuzzy information systems, Knowledge-Based Systems 28 (2012) 115–123.
[22] J. Järvinen, Approximations and rough sets based on tolerances, in: Proceedings of Rough Sets and Current Trends in Computing, LNCS, vol. 2005, 2000, pp. 182–189.
[23] W. Jin, A.K. Tung, J. Han, W. Wang, Ranking outliers using symmetric neighborhood relationship, in: Proceedings of the 10th Pacific-Asia Conference on Knowledge Discovery and Data Mining, 2006, pp. 577–593.
[24] M. Kukar, I. Kononenko, Cost-sensitive learning with neural networks, in: Proceedings of the 13th European Conference on Artificial Intelligence, 1998, pp. 445–449.
[25] H. Li, M. Wang, X. Zhou, J. Zhao, An interval set model for learning rules from incomplete information table, International Journal of Approximate Reasoning 53 (2012) 24–37.
[26] T.Y. Lin, Neighborhood systems and approximation in database and knowledge base systems, in: Proceedings of the 4th International Symposium on Methodologies of Intelligent Systems, ACM, 1989, pp. 75–86.
[27] T.Y. Lin, Granular computing on binary relations-analysis of conflict and chinese wall security policy, in: Proceedings of Rough Sets and Current Trends in Computing, LNAI, vol. 2475, 2002, pp. 296–299.
[28] T.Y. Lin, Granular computing – structures, representations, and applications, in: Lecture Notes in Artificial Intelligence, vol. 2639, 2003, pp. 16–24.
[29] T.Y. Lin, Neighborhood systems: mathematical models of information granulations, in: Proceedings of IEEE International Conference on Systems, Man & Cybernetics, 2003, pp. 75–86.
[30] C.X. Ling, Q. Yang, J.N. Wang, S.C. Zhang, Decision trees with minimal costs, in: Proceedings of the 21st International Conference on Machine learning, 2004, p. 69.
[31] P. Lingras, C. Davies, Rough genetic algorithms, in: Lecture Notes in Computer Science, vol. 1711, 1999, pp. 38–46.
[32] D. Liu, T.R. Li, D. Ruan, Probabilistic model criteria with decision-theoretic Rough sets, Information Sciences 181 (2011) 3709–3722.
[33] G. Liu, Generalized rough sets over fuzzy lattices, Information Sciences 178 (6) (2008) 1651–1662.
[34] Q. Liu, F. Li, F. Min, M. Ye, G. Yang, An efficient reduction algorithm based on new conditional information entropy, Control and Decision 20 (8) (2005) 878–882 (in Chinese).
[35] Z. Meng, Z. Shi, A fast approach to attribute reduction in incomplete decision systems with tolerance relation-based rough sets, Information Sciences 179 (16) (2009) 2774–2793.
[36] 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, Information Sciences 179 (24) (2009) 4140–4150.
[37] F. Min, H. He, Y. Qian, W. Zhu, Test-cost-sensitive attribute reduction, Information Sciences 181 (2011) 4928–4942.
[38] F. Min, Q. Liu, A hierarchical model for test-cost-sensitive decision systems, Information Sciences 179 (2009) 2442–2452.
[39] F. Min, W. Zhu, Attribute reduction with test cost constraint, Journal of Electronic Science and Technology of China 9 (2) (2011) 97–102.
[40] F. Min, W. Zhu, Optimal sub-reducts in the dynamic environment, in: Proceedings of IEEE International Conference on Granular Computing, 2011, pp. 457–462.
[41] F. Min, W. Zhu, Optimal sub-reducts with test cost constraint, in: Proceedings of Rough Set and Knowledge Technology, LNAI, vol. 6954, 2011, pp. 57–62.
[42] F. Min, W. Zhu, H. Zhao, G. Pan, Coser: Cost-senstive rough sets, 2011. <http://grc.fjzs.edu.cn/~fmin/coser/>.
[43] Z. Pawlak, Rough sets, International Journal of Computer and Information Sciences 11 (1982) 341–356.
[44] Z. Pawlak, Rough Sets: Theoretical Aspects of Reasoning about Data, Kluwer Academic Publishers, Boston, 1991.
[45] Z. Pawlak, Rough set theory and its applications, Journal of Telecommunications and Information Technology 3 (2002) 7–10.
[46] Z. Pawlak, Rough sets and intelligent data analysis, Information Sciences 147 (12) (2002) 1–12.
[47] J.F. Peters, Near sets. General theory about nearness of objects, Applied Mathematical Sciences 1 (53) (2007) 2609–2629.
[48] L. Polkowski, A set theory for rough sets: toward a formal calculus of vague, Fundamenta Informaticae 71 (1) (2006) 49–61.
[49] Y. Qian, J. Liang, W. Pedrycz, C. Dang, Positive approximation: an accelerator for attribute reduction in rough set theory, Artificial Intelligence 174 (9–10) (2010) 597–618.
[50] J.R. Quinlan, Induction of decision trees, Machine Learning 1 (1986) 81–106.
[51] S. Romański, Operations on families of sets for exhaustive search given a monotonic function, in: Proceedings of the 3rd International Conference on Data and Knowledge Bases, 1988, pp. 28–30.
[52] A. Skowron, C. Rauszer, The discernibility matrices and functions in information systems, in: Intelligent Decision Support, 1992, pp. 331–362.
[53] A. Skowron, J. Stepaniuk, Tolerance approximation spaces, Fundamenta Informaticae 27 (1996) 245–253.
[54] D. Ślęzak, Approximate entropy reducts, Fundamenta Informaticae 53 (3–4) (2002) 365–390.
[55] P.D. Turney, Cost-sensitive classification: empirical evaluation of a hybrid genetic decision tree induction algorithm, Journal of Artificial Intelligence Research 2 (1995) 369–409.
[56] P.D. Turney, Types of cost in inductive concept learning, in: Proceedings of the Workshop on Cost-Sensitive Learning at the 17th ICML, 2000, pp. 1–7.
[57] G. Wang, Attribute core of decision table, in: Proceedings of Rough Sets and Current Trends in Computing, LNCS, vol. 2475, 2002, pp. 213–217.
[58] G. Wang, H. Yu, D. Yang, Decision table reduction based on conditional information entropy, Chinese Journal of Computers 2 (7) (2002) 759–766.
[59] X.-Z. Wang, J.-H. Zhai, S.-X. Lu, Induction of multiple fuzzy decision trees based on rough set technique, Information Sciences 178 (2008) 3188–3202.
[60] W. Wei, J. Liang, Y. Qian, A comparative study of rough sets for hybrid data, Information Sciences 190 (2012) 1–16.

[61] J. Wróblewski, Finding minimal reducts using genetic algorithms, in: Proceedings of International Workshop on Rough Sets Soft Computing at Second Annual Joint Conference on Information Sciences, 1995, pp. 186–189.
[62] Q. Yang, X. Wu, 10 challenging problems in data mining research, International Journal of Information Technology and Decision Making 5 (4) (2006) 597–604.
[63] Y.Y. Yao, Relational interpretations of neighborhood operators and rough set approximation operators, Information Sciences 111 (1–4) (1998) 239–259.
[64] Y.Y. Yao, A partition model of granular computing, Lecture Notes in Computer Science 3100 (2004) 232–253.
[65] Y.Y. Yao, S. Wong, A decision theoretic framework for approximating concepts, International Journal of Man–Machine Studies 37 (1992) 793–809.
[66] Y.Y. Yao, Y. Zhao, Attribute reduction in decision-theoretic rough set models, Information Sciences 178 (17) (2008) 3356–3373.
[67] Y.Y. Yao, Y. Zhao, J. Wang, On reduct construction algorithms, in: Proceedings of Rough Set and Knowledge Technology, LNAI, vol. 4062, 2006, pp. 297–304.
[68] Z. Zhou, X. Liu, Training cost-sensitive neural networks with methods addressing the class imbalance problem, IEEE Transactions on Knowledge and Data Engineering 18 (1) (2006) 63–77.
[69] W. Zhu, Basic concepts in covering-based rough sets, in: Proceedings of the International Conference on Natural Computation, vol. 1, 2007, pp. 283–286.
[70] W. Zhu, Generalized rough sets based on relations, Information Sciences 177 (22) (2007) 4997–5011.
[71] W. Zhu, Topological approaches to covering rough sets, Information Sciences 177 (6) (2007) 1499–1508.
[72] W. Zhu, F. Wang, Reduction and axiomization of covering generalized rough sets, Information Sciences 152 (1) (2003) 217–230.
[73] W. Ziarko, Variable precision rough set model, Journal of Computer and System Sciences 46 (1) (1993) 39–59.