Knowledge and Information Systems (2019) 61:277–299
https://doi.org/10.1007/s10115-018-1288-5

REGULAR PAPER

Heuristic attribute reduction and resource-saving algorithm for energy data of data centers

Mincheng Chen[1] · Jingling Yuan[1] · Lin Li[1] · Dongling Liu[1] · Yang He[1]

Received: 27 February 2018 / Revised: 24 May 2018 / Accepted: 28 November 2018 /
Published online: 17 December 2018

Abstract
Energy data, which consist of energy consumption statistics and other related data in green data centers, grow dramatically. The energy data have great value, but many attributes within them are redundant and unnecessary, and they have a serious impact on the performance of the data center's decision-making system. Thus, attribute reduction for the energy data has been conceived as a critical step. However, many existing attribute reduction algorithms are often computationally time-consuming. To address these issues, firstly, we extend the methodology of rough sets to construct data center energy consumption knowledge representation system. Energy data will occur some degree of exceptions caused by power failure, energy instability or other factors; hence, we design an integrated data preprocessing method using Spark for energy data, which mainly includes sampling analysis, data classification, missing data filling, outlier data prediction and data discretization. By taking good advantage of in-memory computing, a fast heuristic attribute reduction algorithm (FHARA-S) for energy data using Spark is proposed. In this algorithm, we use an efficient algorithm for transforming energy consumption decision table, a heuristic formula for measuring the significance of attribute to reduce the search space, and introduce the correlation between condition attribute and decision attribute, which further improve the computational efficiency. We also design an adaptive decision management architecture for the green data center based on FHARA-S, which can improve decision-making efficiency and strengthen energy management. The experimental results show the speed of our algorithm gains up to 2.2X performance improvement over the traditional attribute reduction algorithm using MapReduce and 0.61X performance improvement over the algorithm using Spark. Besides, our algorithm also saves more computational resources.

Keywords Energy data · Attribute reduction · Rough sets · Heuristic · Spark

1 Introduction

Today, cloud computing is sweeping the globe [29]. Data centers have become the major facilities to support highly varied big data processing tasks [12]. Nevertheless, their huge

✉ Jingling Yuan
yuanjingling@126.com

Extended author information available on the last page of the article

IT energy consumption, environmental issues have become increasingly prominent [23]. Therefore, widely deploying large-scale green data centers over the world is an approach to tackle the dual challenges of energy deficiency and environmental pollution [21,22]. It produces the energy data, which comprise monitoring data and other relevant data in green data centers, rise extremely [6]. Velocity and volume are presented by energy data due to the instable renewable energy and the enormous scale. Besides, energy data show variety as their shapes are complex, usually mixing with structured data, semi-structured data and unstructured data. Moreover, energy data have great value for monitoring operation status, making decision and scheduling resources if they could be processed and analyzed effectively. In fact, plenty of attributes within the energy data are redundant for making decision, that is, unnecessary attributes may raise the computing cost and degenerate the performance of mining algorithms [1,14]. Furthermore, storing all attributes will be extremely expensive and unreasonable [26]. Therefore, it is necessary to select informative attributes for decreasing the cost of storing, shortening the computation time and obtaining the better decision making.

To address these issues, many efficient attribute reduction methods have been proposed. However, these methods are still time-consuming in dealing with large-scale data. As we have seen, attribute reduction algorithms for the energy data using Spark have not been widely researched. And Spark proposes a new concept called resilient distributed datasets [37,45], through which we can cache intermediate results in memory. This characteristic is benefit to iterative algorithm like attribute reduction [2]. In this thesis, the following issues are investigated:

(1) Rough set theory is an important method to extract rules from information systems. Thus, we introduce a theoretical framework based on rough sets and extend the methodology of rough sets to construct data center energy consumption knowledge representation system.
(2) The energy data will occur some degree of exception caused by servers, switches and rack failures. So we design a data preprocessing method for the energy data using Spark. The method analyzes the sampled data and then uses the corresponding programs to clean the different types of abnormal data, which include missing data filling and outlier data prediction. It discretizes energy data finally.
(3) By taking good advantages of Apache Spark, we propose a fast heuristic attribute reduction algorithm (FHARA-S) for energy data using Spark. In our algorithm, we use an efficient algorithm for transforming energy consumption decision table, a heuristic formula for measuring the significance of attribute to reduce the search space, and introduce the correlation between condition attribute and decision attribute. We also design an adaptive decision management architecture for the green data center based on FHARA-S. The experiments reveal that the speed of our algorithm gains up to 2.2X performance improvement over the traditional attribute reduction algorithm using Hadoop and 0.61X performance improvement over the algorithms using Spark. Besides that, our algorithm also saves computational resources.

The rest of the paper is organized as follows. Section 2 surveys related work. Section 3 gives a description of data center energy consumption knowledge representation system. Section 4 presents the design of an integrated data preprocessing method using Spark for energy data. Section 5 proposes a heuristic attribute reduction algorithm for energy data using Spark. Section 6 presents an adaptive decision management architecture for the green data center. Section 7 presents three groups of experimental results. Section 8 gives a conclusion.

2 Related work

Rough set theory, put forward by Pawlak [31,32], is one of the most useful mathematical tools utilized in processing information with imprecision and incompleteness. It has been employed in a lot of fields successfully, such as financial investment [7], fault diagnosis [30], medical research [8] and pattern recognition [20].

As a significant application of rough set theory, attribute reduction contributes to diminish attribute space and raise algorithm efficiency by reducing attributes which are redundant and unnecessary [3,16]. However, the massive data still cannot be handled through most existing reduction algorithms as high computational cost. To improve these algorithms, many efficient methods were proposed in the past several years. Wei et al. [40] introduced a new information measure and proposed an algorithm to obtain an attribute reduct from hybrid data. Zhang et al. [48] proposed an efficient matrix-based approach for fast updating approximations in dynamic information systems, which was vital for attribute reduction. Lu et al. [27] devised a fast attribute reduction algorithm based on boundary region, which has the ability to efficiently find a reduct from a large incomplete decision system. Wang and Liang [38] studied an efficient attribute reduction algorithm based on the idea of decomposition and fusion for large-scale hybrid data sets. Zheng et al. [51] enhanced the efficiency of the reduction algorithm by improving the computing method of heuristic information. Liang et al. [25] developed a group incremental approach for attribute reduction. Xie et al. [41] provided three update strategies: object-related strategy, attribute-related strategy and both-related strategy and then developed an efficient incremental attribute reduction algorithm for the dynamic incomplete decision system. Chen et al. [4] proposed an incremental algorithm for attribute reduction with variable precision rough sets, to decrease the time complexity. Jing et al. [18] developed an incremental attribute reduction algorithm with a multi-granulation view to deal with large-scale decision systems. Wang et al. [39] computed the corresponding discernibility matrix by the most informative instances selected from large-scale datasets and then found all reducts. Liang et al. [24] considered the subtables within a large-scale dataset as small granularities, and he designed an algorithm to build an approximate reduct by fusing the feature selection results of small granularities. In conclusion, many efficient approaches have been developed. However, the approaches mentioned above cannot perform effectively if the data set is huge volume or high-dimensional. Moreover, sampling techniques merely support the samples can represent the whole data or meet the hypothesis space.

Recently, many scholars have attempted to parallelize the attribute reduction algorithms for enhancing their performance on large-scale data. Zhang et al. [49] studied parallel algorithms for knowledge acquisition on varying MapReduce runtime systems. Furthermore, they presented three different parallel matrix-based methods to handle large-scale data [50]. Ma et al. [28] put forward a parallel heuristic technique to find attribute reduct and apply it to distribution network for fault detection. Qian et al. [33,34] presented various parallelization strategies for attribute reduction to raise the computational efficiency. Czolombitko et al. [9] proposed a parallel attribute reduction algorithm MRCR. To reduce the memory complexity, they used counting tables to compute discernibility measure of the datasets. Chen et al. [5] studied the algorithms for attribute reduction in parallel using dominance-based neighborhood rough sets, which considered the partial orders among numerical and categorical attribute values. Hu et al. [15] designed a strategy for large-scale multimodality attribute reduction with multi-kernel fuzzy rough sets and developed it in the framework of MapReduce. Ding et al. [10] presented a multi-agent consensus MapReduce optimization model and co-evolutionary quantum PSO with self-adaptive memeplexes for designing the attribute

reduction approach. El-Alfy and Alshammari [11] presented a MapReduce method for discovering the minimal reduct by the parallel genetic algorithm. However, parallel attribute reduction algorithms based on rough sets need to output a lot of intermediate results to Hadoop Distributed File System (HDFS), which causes plenty network IO and disk. Therefore, they are still time-consuming to deal with massive data.

3 Related concepts in energy consumption knowledge representation system

Rough set theory is an important mathematical tool to extract rules from information systems. To promote its application in attribute reduction for energy data, the rough set approach is extended to construct data center energy consumption knowledge representation system. We use the energy consumption data collected as the domain U. By considering the multiple supply situations and the energy change parameters, the energy supplies are classified, so as to determine the condition attribute set C and the decision attribute set D. The specific related definitions are as follows:

Definition 1 An energy consumption decision table can be described as $S = (U, At, \{V_a | a \in At\}, \{I_a | a \in At\})$, where $U = \{x_1, x_2, \ldots, x_n\}$ is a non-empty finite set of energy consumption records; $At = C \bigcup D$ is a non-empty finite set of attributes, where C is the set of energy consumption conditional attributes of system components, D is the set of decision attributes, and $C \bigcap D = \emptyset$; V_a is a domain of the attribute $a \in At$; I_a is an information function that maps an energy consumption record x in U to exactly one value v in V_a, that is, $I_a(x) = v$.

Definition 2 Suppose that $S = (U, At, \{V_a | a \in At\}, \{I_a | a \in At\})$ is an energy consumption decision table, and let $P \subseteq C$ be a subset of condition attributes. An equivalence relation with respect to P is defined as:

$$IND(P) = \{(x, y) \in U \times U | \forall a \in P, \quad I_a(x) = I_a(y)\} \tag{1}$$

$IND(P)$ partitions U into several equivalence classes given by:

$$U/IND(P) = \{[x]_P | x \in U\} \tag{2}$$

where $[x]_P$ denotes the equivalence class determined by x with respect to P, $[x]_P = \{y \in U | (x, y) \in IND(P)\}$. For simplicity, U/P will be instead of $U/IND(P)$.

Definition 3 Let $U/D = \{d_1, d_2, \ldots, d_m\}$ be the partition of the universe U with respect to the decision attribute D. Then the positive region and boundary region of D with respect to C are defined as:

$$POS_C(D) = \bigcup_{i=1}^{m} \underline{Apr_C}(d_i) \tag{3}$$

$$BND_C(D) = U - POS_C(D) \tag{4}$$

where $\underline{Apr_C}(d_i) = \{x \in U | [x]_C \subseteq d_i\}$ is the lower approximation of d_i defined by $IND(C)$.

An energy consumption decision table S is consistent if whole records in an equivalence class defined by C have the same decision attribute value, i.e., $\forall (x, y) \in U \times U, \forall a \in C, [I_a(x) = I_a(y)] \rightarrow [I_d(x) = I_d(y)]$. In this case, $POS_C(D) = U$, and $BND_C(D) = \emptyset$. Otherwise, the table is inconsistent.

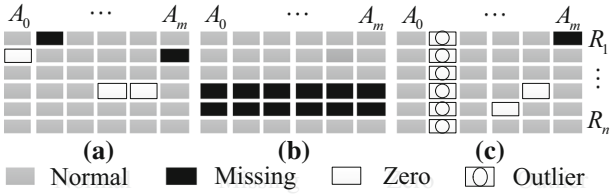


Fig. 1 Different types of abnormal data

Definition 4 Let B be a subset of C; a conditional attribute $a \in B$ is indispensable in B if $POS_{B-\{a\}}(D) \neq POS_B(D)$; otherwise, a is dispensable in B. A conditional attribute set $B \subseteq C$ is a reduct of C with respect to D if it satisfies the following two conditions:

(1) $POS_B(D) = POS_C(D)$;
(2) $\forall a \in B, POS_{B-\{a\}}(D) \neq POS_B(D)$.

The reduct of an energy consumption decision table is the minimal subset of C, which ensures the positive region unchanged.

4 Data preprocessing

Energy data will occur some degree of exception caused by servers, switches, rack failures due to power off, energy instability or other factors. Figure 1a shows short-term data missing and zero-value data caused by inrush current. Figure 1b shows multi-point data missing of partial time segment caused by the power supply failure. Figure 1c shows outlier data caused by the unstable renewable energy. Meanwhile, the quality of the energy big data will be affected by RTU (Remote Terminal Unit) collection, power meter acquisition, channel status, parameter setting and other factors. These will influence the reduction results even though they can reflect the actual operation situation. So it is imperative to clean and preprocess the energy data. But most existing researches merely concentrate on some aspects of data preprocessing [19,36,46]. And the input data cannot be effectively handled by the traditional statistical methods because of its large scale [13,52]. So we design a systematic and suitable data preprocessing method using Apache Spark for energy data analysis in green data center. Figure 2 illustrates the flowchart of our method.

First of all, we do sampling analysis for the raw data and find the quality problems existed in the energy data. In the process of sampling analysis, how to determine the sample size is one of the most significant problems. Hence, we utilize a familiar method [24,38] to make sure the sample size of the energy data set. The specific method is as follows:

Let the size of the original energy data set S be denoted by N. After that, the sample size n_0 can be calculated as:

$$n_0 = \frac{Z^2 \times \sigma^2}{d^2} \tag{5}$$

where σ denotes the standard deviation on S, Z denotes the Z-statistic under confidence intervals (e.g., 95% confidence interval of Z-statistic is 1.96 and 90% confidence interval of Z-statistic is 1.64) and d denotes the tolerable error which could be adjusted as necessary.

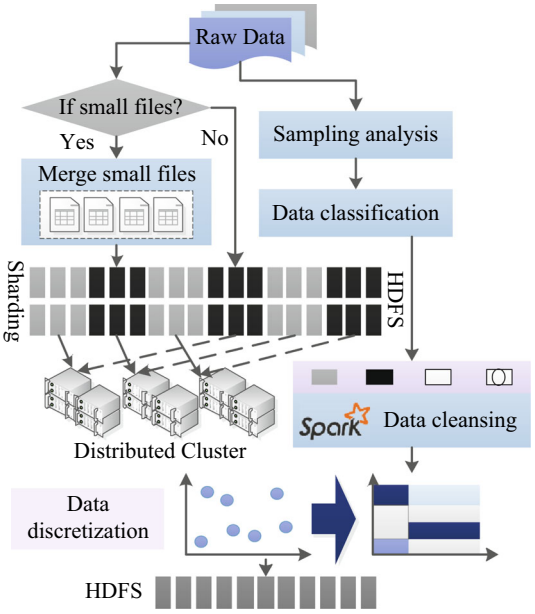


Fig. 2 Flowchart of the data preprocessing method using Spark

Moreover, if sample size n_0 is bigger than 5% of the whole size N, we should adjust the sample size n_0. The adjusted formula is calculated as follows:

$$n = \frac{n_0 \times N}{n_0 + N} \tag{6}$$

Then combined with the abnormal data sources, we classify the data. The classification results are the important basis for data cleaning. Then we call the corresponding parallel programs to clean the abnormal data in the light of these results. For example, we fill the predicted values for missing data [43,44] and carry out regression analysis for outlier data. Discrete attribute values are easier to utilize and understand than continuous attribute values, because they are much closer to the knowledge-level representation [17,35]. Many studies also indicate that rules with discrete attribute values are more comprehensive, and discretization can enhance efficiency of attribute reduction algorithms. Therefore, the equal-interval-width method is elected to discretize continuous attributes according to the characteristics of energy data in the end.

Let $\{A_1, A_2, \ldots, A_i, A_{i+1}, \ldots, A_n\}$ be the energy data set of all attributes, where attributes $A_1, A_2, \ldots, A_i$ are continuous and attributes $A_{i+1}, A_{i+2}, \ldots, A_n$ are discrete $(1 < i \leq n)$. The interval length of the attribute A_i is computed as

$$d_{A_i} = \frac{max(A_i) - min(A_i)}{t_{A_i}} \tag{7}$$

where $max(A_i)$ is the maximum value of the attribute A_i and $min(A_i)$ is the minimum value, and t_{A_i} is the number of interval. Then the breakpoint is computed as

$$q_j = min(A_i) + j \times d_{A_i} \qquad (0 < j \leq t - 1) \tag{8}$$

Algorithm 1 PDA

Input: Energy data set S
Set of interval numbers $T = \{t_{A_1}, t_{A_2}, \ldots, t_{A_i}\}$
Output: Discrete energy data set S_D
1: **begin**
2: $S \leftarrow spark.textfile(S)$;
▷ Convert S to RDD
3: let $minMax \leftarrow \emptyset$;
▷ initialize two-dimensional array
4: **for** $i \leftarrow 1, n$ **do**
5: $minMax(i) \leftarrow (S.\mathbf{map}(x \Rightarrow x(i)).min(), S.\mathbf{map}(x \Rightarrow x(i)).max())$;
6: **end for**
7: $S' \leftarrow EC.\mathbf{Map}(discretize()\{$
8: let $DsArr \leftarrow \emptyset$;
9: **for** $i \leftarrow 1, record.length$ **do**
10: $d_{A_i} \leftarrow (minMax(i)._2 - minMax(i)._1)/t_{A_i}$;
11: $q_j \leftarrow minMax(i)._1 + (record(i)/d_{A_i}) \times d_{A_i}$;
12: $q_{j+1} \leftarrow q_j + d_{A_i}$;
13: $DsArr(i) \leftarrow (q_j, q_{j+1}]$;
14: **end for**
15: **return** $DsArr$;
16: $\}).\mathbf{map}(\_mkString(""))$
17: **end**

where d_{A_i} is the interval partition and j is the sequence number of breakpoints. So the interval partition on the domain of the attribute A_i is equal to

$$Q_{A_i} = \{[q_0, q_1], \quad (q_1, q_2], \ldots, (q_{t_{A_i}-2}, q_{t_{A_i}-1}]\} \tag{9}$$

The pseudocode of parallel algorithm for discretization (PDA) is shown in Algorithm 1.

By Algorithm 1, it computes the minimum and maximal values of every continuous attribute and then stores them in the two-dimensional array $minMax$ (lines 4–6). After that, the algorithm discretizes the continuous attributes (lines 6–15). The array $DsArr$ is utilized to store the attributes after discretizing.

As the energy data source may be composed of many small files which less than 64M, we also write a program for merging small files to avoid affecting the operating efficiency of Spark. The data are stored directly in HDFS. This method can take full advantage of the data parallelization on Spark, which greatly optimizes the preprocessing efficiency.

5 Heuristic attribute reduction algorithm for energy data using Spark

5.1 The simplification and transformation of decision table

Studying the historical energy consumption data, we discover that the data have features like correlation, continuity, periodicity. Plenty of data records are similar. They may belong to the same equivalence class. Thus, equivalence class can be utilized to transform the energy consumption decision table. And energy data will show some degree of inconsistency caused by imperfect information collection, power off, servers failure, energy instability or other factors. It may affect decision making. Furthermore, the inconsistent decision table cannot be dealt with by the attribute reduction algorithms. Hence, the energy consumption decision table should be transformed.

Definition 5 Given an energy consumption decision table $S = (U, At, \{V_a|a \in At\}, \{I_a|a \in At\})$, for conditional attributes C, if the decision table produces inconsistent objects on C, the inconsistent objects are defined as

$$R_C = \{(x_i, x_j)|x_i, x_j \in U, i \neq j, \forall a \in B, f(x_i, a) = f(x_j, a) \wedge f(x_i, D) \neq f(x_j, D)\} \tag{10}$$

Analyzing the inconsistent objects by Definition 5, we can discover that there are two main categories of their distribution:

(1) Due to energy data collection error, or equipment failure, some records' decision attributes deviate from the normal values. Hence, they cause inconsistency of data. The distribution of these inconsistent objects is often irregular, and their proportion is very small.

(2) Because of incomplete energy data collection, some key conditional attributes are missing. It causes that different decision attribute values appear under the same condition attributes. The distribution of these inconsistent objects is often regular, and their proportion is larger.

In order to describe the distribution of the inconsistent objects better, we give the following definition.

Definition 6 Given an energy consumption decision table $S = (U, At, \{V_a|a \in At\}, \{I_a|a \in At\})$, $U/D = \{D_1, D_2, \ldots, D_n\}$,where $n = |U/D|$. For any $x \in U$, the distribution function of object x on every decision class with respect to C is defined as

$$\psi_{D|C}(x) = (P(D_1|[x]_C), \quad P(D_2|[x]_C), \ldots, P(D_n|[x]_C)) \tag{11}$$

where $D_j \in U/D$ $(1 \leq j \leq m)$ and $P(D_j|[x]_C) = \frac{|[x]_C \cap D_j|}{|[x]_C|}$. According to the distribution types of the inconsistent objects and their distribution function, we design a method to transform an inconsistent decision table into a consistent decision table which represents

$$f(x, D) = \begin{cases} MAX_D & (Max(\psi_{D|C}(x)) > \alpha) \\ \bigcup_{P(D_j|[x]_C) \neq 0} D_j & (Max(\psi_{D|C}(x)) < \alpha) \end{cases} \tag{12}$$

where $Max()$ denotes the function obtains the maximum and α denotes the threshold, which is taken as 0.8.

Based on the theories and methods mentioned above, we design a parallel algorithm for simplifying and transforming decision table (PASTDT) based on Spark. The pseudocode of PASTDT is shown in Algorithm 2. At the start of Algorithm 2, $spark.textfile()$ loads the original decision table (line 2). Then the different equivalence classes which induced from conditional attributes are computed by the algorithm (lines 3–6). Thereafter, it transforms the decision table (lines 7–17). $getDecisionArr\_distribution()$ is utilized to obtain the distribution of object x on every decision class. At last, the algorithm gets a simplified and consistent energy consumption decision table S'.

5.2 Parallelization strategies for attribute reduction

Calculating all attribute reducts for an energy consumption decision table is an NP-hard problem, and it often needs a huge quantity of time to search the reducts out. The time will grow rapidly with increasing the number of attributes. Hence, the time complexity of attribute reduction is decreased with the heuristic information's help. In this paper, we utilize a heuristic formula for measuring the significance of attribute to reduce search space.

Algorithm 2 PASTDT

Input: $S = (U, At, \{V_a|a \in At\}, \{I_a|a \in At\})$
Output: A simplified and consistent energy consumption decision table $S' = (U', At, \{V_a|a \in At\}, \{I_a|a \in At\})$.

```
1: begin
2: S ← spark.textfile(S_i);  ▷ Convert S_i to RDD
3: EC ← S.Map(
4:       key ← EC_i;
         ▷ EC_i is an equivalence class
5:       value ← id_x;
         ▷ id_x is the id of the record
6: ).groupByKey().values
7: S' ← EC.Map(
8:       record ← consistent_transform(EC_i){
9:           ψ_{D|C}(D_j, P(D_j|[x]_C)) ← getDecisionArr_distribution(EC_i);
             ▷ [x]_C is equivalent to EC_i
10:          if Max(ψ_{D|C}(x)) > α then
11:              DeciValue ← Max_D;
12:          else
13:              DeciValue ← ⋃_{P(D_j|[x]_C)≠0} D_j;
14:          end if
15:          return x + DeciValue;
16:      }
17: )
18: end
```

Definition 7 Given a simplified and consistent energy consumption decision table $S' = (U', At, \{V_a|a \in At\}, \{I_a|a \in At\})$. $\forall P \subseteq C, U'/P = \{EC_1, EC_2, \ldots, EC_m\}$ is condition partitions. $POS_P(D)$ is equal to

$$POS_P(D) = \bigcup_{EC \in U'/P \wedge |EC/D|=1} EC \tag{13}$$

where $|EC/D| = 1$ denotes that all records in EC have the same decision value.

Definition 8 Let $P \subseteq C$ be a subset of condition attributes, and $a \in C - P$, then dependency degree of condition attribute set P with respect to decision attribute D is defined as

$$\gamma_P(D) = |POS_P(D)|/|U'| \tag{14}$$

And the significance of attribute a with respect to condition attribute set P is defined as

$$sig_P(a) = \gamma_{P\cup\{a\}}(D) - \gamma_P(D) \tag{15}$$

Definition 9 Let $a \in C$, and the correlation between condition attribute a and decision attribute D is defined as

$$COR(a, D) = H(D) - H(D|a) \tag{16}$$

where $H(D)$ denotes an entropy function and $H(D|a)$ denotes the conditional entropy of a and D.

According to Definitions 7 and 8, we construct the heuristic attribute reduction algorithm. The algorithm takes the empty set as the starting point, calculates the significance of all the remaining attributes and selects the attribute with the largest significance and then adds it to the reduction. If more attributes than one have the best significance, we should compute the correlation between condition attribute and decision attribute in the light of Definition 9.

Algorithm 3 FHARA-S

Input: $S' = (U', At, \{V_a | a \in At\}, \{I_a | a \in At\})$
Output: Attribute reduct Red
1: **begin**
2: let $Red \leftarrow \emptyset$;
3: To each attribute $a \in C - Red$, computing equivalence classes for candidate attribute subset $Red \cup \{a\}$ by executing Algorithm 4, calculating the attribute significance $sig_{Red}(a)$ and $POS_{Red \cup \{a\}}(D)$ by Algorithm 5;
4: **if** $Count(Max(sig_{Red}(A_1), \ldots, sig_{Red}(A_n)))$ is non-unique **then**
5: $A_m \leftarrow Max(COR(A_j, D), .., COR(A_k, D))$;
▷ $A_j, .., A_k$ have the same max significance value
6: **else**
7: $A_m \leftarrow Max(sig_{Red}(A_1), \ldots, sig_{Red}(A_n))$;
8: **end if**
9: $Red = Red \cup \{A_m\}$;
10: $U' \leftarrow U' - POS_{Red}(D)$;
11: $S' \leftarrow Simplify(U')$;
▷ Simplify ECIS S'
12: **if** $U' = \emptyset$ **then**
13: Stop the algorithm, output the attribute reduct Red;
14: **else**
15: The algorithm turns to step 3;
16: **end if**
17: **end**

Algorithm 4 FHARA-S-ComputeEquivalenceClass

Input: Attribute reduct Red
$S' = (U', At, \{V_a | a \in At\}, \{I_a | a \in At\})$
Output: Equivalence classes $EC_{Red \cup \{a\}}$
1: **begin**
2: $EC_{Red \cup \{a\}} \leftarrow S'$.**Map(**
3: $key \leftarrow EC'_i$;
▷ EC'_i is an equivalence class induced by $Red \cup \{a\}$
4: $value \leftarrow id'_x$;
▷ id'_x is the id of the record
5:).**groupByKey()**.$values$
6: **end**

Algorithm 5 FHARA-S-Compute-POS-SIG

Input: Equivalence classes $EC_{Red \cup \{a\}}$
Output: $sig_{Red}(a)$ and $POS_{Red \cup \{a\}}(D)$
1: **begin**
2: $POS_{Red \cup \{a\}}(D) \leftarrow EC_{Red \cup \{a\}}$.**filter**($getPOS()$){
3: **return** $EC_i.ifDecisionAttr()$;
4: })
5: $sig_{Red}(a) \leftarrow compute_{sig}(POS_{Red \cup \{a\}}(D))$;
▷ compute $sig_{Red}(a)$
6: $outPut\ (sig_{Red}(a), POS_{Red \cup \{a\}}(D))$;
7: **end**

The attribute which has the largest correlation is selected. Finally, the algorithm outputs the reduction result.

In order to further simplify the reduction algorithm, we improve the algorithm by Definition 10 [42].

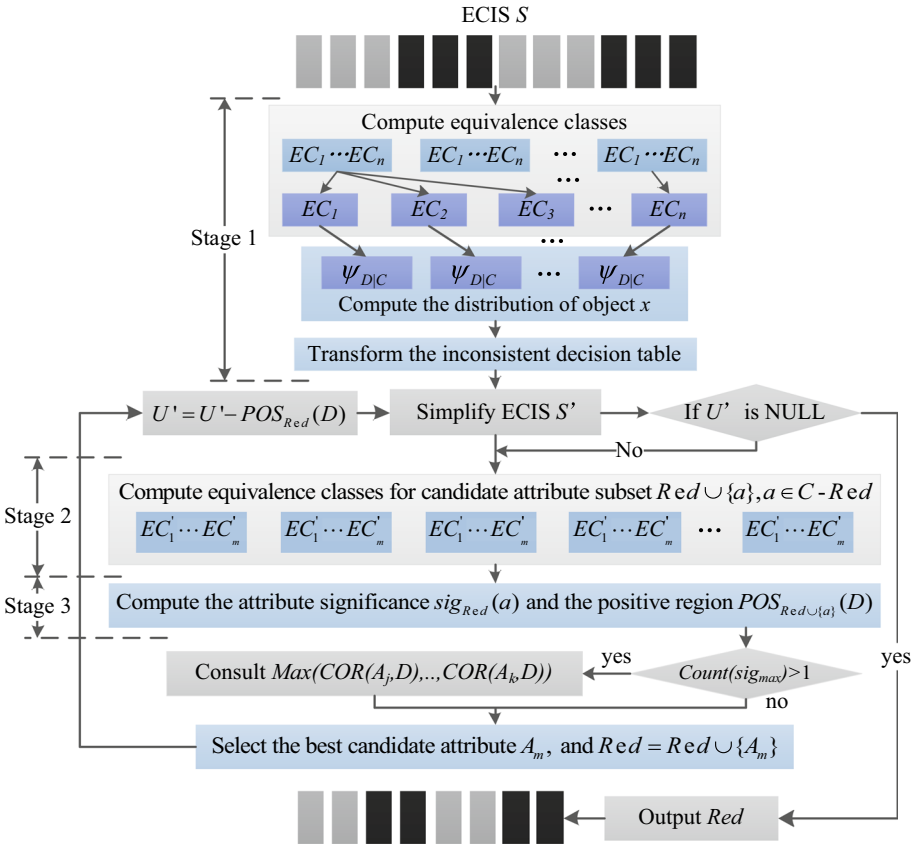


Fig. 3 Heuristic attribute reduction algorithm for energy data using Spark

Definition 10 Let $U' = \{x_1, x_2, \ldots, x_m\}$ be a finite set of energy consumption records and $P \subseteq C$ be a subset of condition attributes, for $a \in C - P$, then

$$POS_{P\cup\{a\}}(D) = \underset{(x\in U')}{POS_P(D)} \cup \underset{(x\in U'-POS_P(D))}{POS_{P\cup\{a\}}(D)} \tag{17}$$

$POS_P(D)$, $(x \in U')$ has been already obtained in the previous stage. We only calculate $POS_{P\cup\{a\}}(D)$, $(x \in U' - POS_P(D))$ to acquire $POS_{P\cup\{a\}}(D)$, $(x \in U')$. Compared with the traditional computing approaches [33], this approach can cut down time complexity from $O(|U'|/n)$ to $O(|U' - POS_P(D)|/n)$, and n is the number of computational nodes. Since we can use $POS_P(D)$, $(x \in U')$, which is get during the computation process, to simplify S' by Definition 6. More importantly, time cost of transforming B is very small. Hence, it can cut time complexity and search space down further.

According to Definitions 7, 8, 9 and 10, we design a fast heuristic attribute reduction algorithm based on Spark (FHARA-S). Algorithm FHARA-S includes Algorithm FHARA-S-ComputeEquivalenceClass and FHARA-S-Compute-POS-SIG. The pseudocodes of the algorithms are presented in Algorithms 3, 4, 5, respectively.

Firstly, Algorithm 3 initializes the attribute reduct (line 2) and then calculates equivalence classes for candidate attribute subset, the corresponding attribute significance and positive

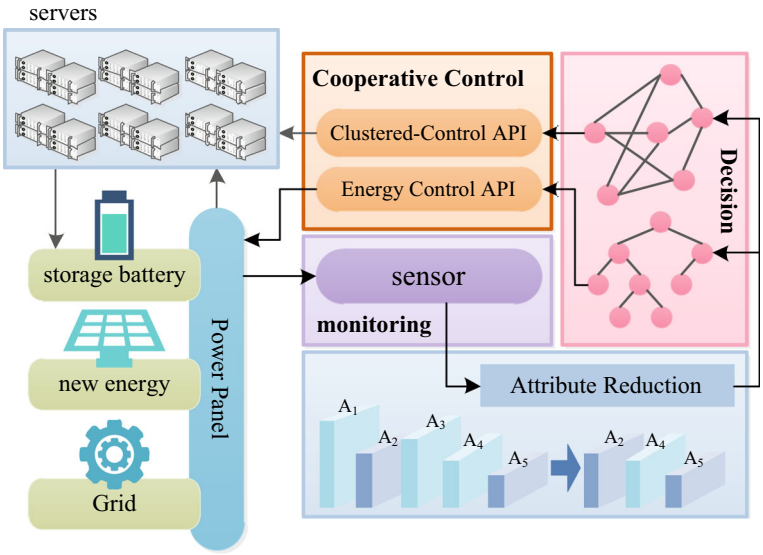


Fig. 4 Adaptive decision management architecture for the green data center based on FHARA-S

region (line 3). Then the best candidate attribute is chosen and added to the attribute reduct (lines 4–8). After that the algorithm updates the records and simplifies the decision table (lines 9–10). The algorithm outputs the reduction result when it meets the terminal condition (lines 12–16).

Algorithm 4 is mainly utilized to calculate equivalence classes induced from candidate attribute subset $Red \cup \{a\}$(lines 2–5).

Algorithm 5 firstly calculates the positive region $POS_{Red\cup\{a\}}(D)$(lines 2–4). $ifDecisionAttr()$ judges whether decision attribute values in EC_i' are consistent. If $ifDecisionAttr()$ returns false, the equivalence classes whose decision attribute values are inconsistent will be deleted by filter operator. Then the algorithm calculates the attribute significance (line 5). The intermediate result is output by $outPut()$ (line 6).

Figure 3 illustrates the heuristic attribute reduction algorithm for energy data using Spark, where Stage 1 is composed of Algorithm 2, Stage 2 is composed of Algorithm 4, Stage 3 is composed of Algorithm 5.

6 The adaptive decision management architecture

For the features as large scale of green data centers, data processing time-sensitive and the heterogeneous ways of data monitoring, we envision the adaptive decision management architecture for the green data center based on the parallel heuristic reduction algorithm mentioned above (shortened as PAHAR), which can decrease the computational complexity of the learning algorithms and improve the model accuracy as well as their description ability by eliminating the redundant information. The decision management architecture is shown in Fig. 4.

In the system architecture, renewable energy, grid and storage battery are the mainly components that provide the power for data centers. The architecture monitors the envi-

Table 1 Environment of experiments

Category	Item	Configuration or version
Hardware	CPU	Intel Core i5-2410M
	Memory	4 GB
	Hard disk	500 GB
Software	Operation system	Ubuntu 14.04
	Hadoop	2.6.0
	Spark	1.5.1

Table 2 Data sets information

	Data sets	Instances	Attributes
1	DS1	1,152,067	25
2	DS2	2,099,647	25
3	DS3	3,114,464	25
4	DS4	3,114,464	37
5	DS5	3,114,464	50
6	DS6	4,013,624	25
7	DS7	8,008,986	25
8	DS8	16,060,782	25

ronment via various monitoring equipments and runs the monitoring programs (such as a guardian process) on each server for the purpose of resource monitoring. These equipments and processes, comply with certain standards, communicate with the master node and transmit collected data to compute nodes. Then the reduction for initial energy data is parallel processed. The task is divided into several sub-parts for different processors, so as to select the necessary attributes rapidly. After eliminating the redundant attributes, the final data will be obtained. The architecture utilizes the data to train decision models and finally gets the power supply strategies and cluster scheduling strategies, which manage energy supply and resource allocation efficiently. Thus, it can provide desirable services for the entire data center system.

7 Experimental evaluation

Our algorithm FHARA-S is coded in Scala and implemented by the framework of Spark. We run parallel algorithms on a cluster of 4 nodes. One computer serves as a master node, and the rest computers serve as slave nodes. They connect via an Ethernet (100 Mbps). Detailed information regarding the configuration of software and hardware is described in Table 1.

We choose the energy consumption produced by green data centers as experimental data (DS1-3). And we also study the change law of energy data by analyzing the data set DS1-3 and write a corresponding simulation data generator to create new large data sets (DS6-8) owing to the limited quantity of samples. In addition, two synthetic data sets DS4-5 have been generated on the basis of DS3. DS4 has 12 more attributes which are randomly generated than DS3, and DS5 has 25 more attributes which are randomly generated than DS3. Table 2 reveals the information of these 8 data sets. Energy data present some degree of exception frequently. Hence, we apply data preprocessing approach to deal with them first.

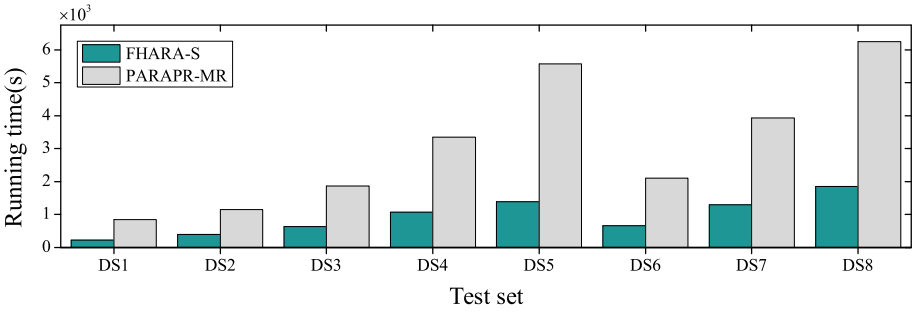


Fig. 5 Comparisons of two algorithms for eight data sets

7.1 Performance comparison with the traditional attribute reduction algorithm using MapReduce

We select PARAPR-MR as the comparison algorithm in this section. PARAPR-MR algorithm in our paper is implemented on the basis of PAAR-PR which is proposed in the literature [33]. It is a traditional representative attribute reduction algorithm using MapReduce.

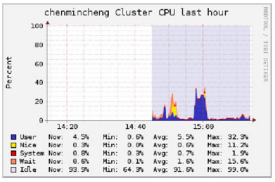
7.1.1 Running time comparison

We execute our heuristic attribute reduction algorithm FHARA-S on Spark cluster and the traditional algorithm PARAPR-MR [33] on Hadoop cluster. Figure 5 shows the total running time of FHARA-S and PARAPR-MR on DS1–DS8. We can discover that when the data set is small, e.g., DS1, FHARA-S can outperform PARAPR-MR by 2.7X, since the computation time of FHARA-S dominates the total running time. But for PARAPR-MR, its job initialization time contributes a large part of its cost. When the data set is large, e.g., DS7, DS8, in this circumstance, the computing resources of the cluster are reasonably utilized. The computation time dominates the total running time of FHARA-S and PARAPR-MR, which can show their true performance. FHARA-S also can outperform PARAPR-MR by 2.2X. Because PARAPR-MR reads intermediate results in each iteration. It needs a lot of disk I/Os, network I/O and unnecessary processes. However, the intermediate results of FHARA-S can be cached in memory during the whole iterative computation, which effectively improves the performance of attribute reduction. At last, from the experimental results on DS3-5, we find that FHARA-S promotes around 1.9X, 2.1X and 3X performance improvement over PARAPR-MR on DS3-5, respectively. This shows that FHARA-S has more advantages with dimension increasing.

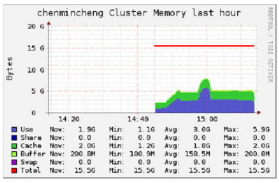
7.1.2 Resource utilization comparison

Compared with PARAPR-MR, FHARA-S significantly improves the efficiency of the algorithm, and we need to further verify whether FHARA-S has the advantage in saving computational resources. Therefore, we compare the resource utilization of FHARA-S and PARAPR-MR when they process the experimental data sets.

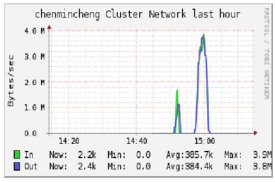
Given the limited space available, we select three groups of representative experiments for analysis. Figure 6 shows the screenshots of the cluster resource monitor. Figure 6a shows the experimental results on DS1, the running time of FHARA-S is 14:58-15:01, and PARAPR-MR is 22:04-22:16. Figure 6b shows the results on DS2, the running time of FHARA-S is



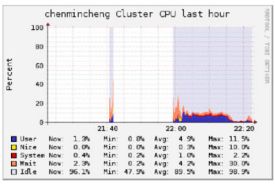
FHARA-S



FHARA-S

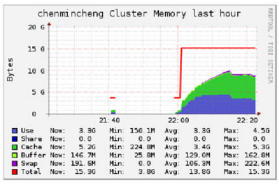


FHARA-S



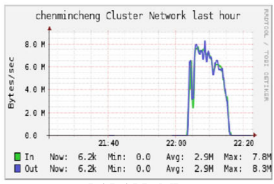
PARAPR-MR

(1) CPU utilization of the cluster



PARAPR-MR

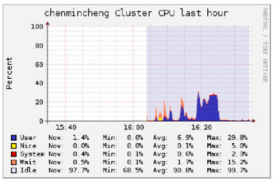
(2) Memory utilization of the cluster



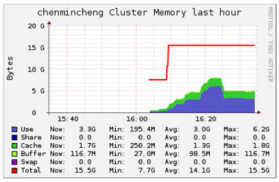
PARAPR-MR

(3) Network utilization of the cluster

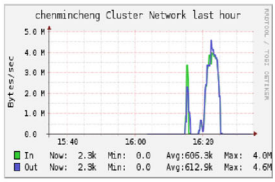
(a) DS1



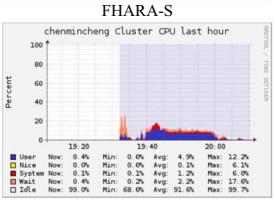
FHARA-S



FHARA-S

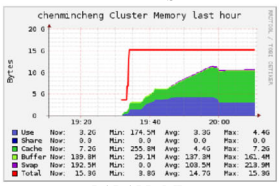


FHARA-S



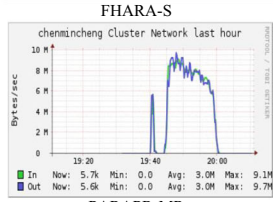
PARAPR-MR

(1) CPU utilization of the cluster



PARAPR-MR

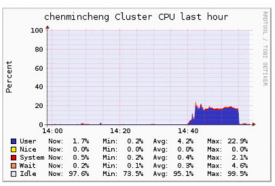
(2) Memory utilization of the cluster



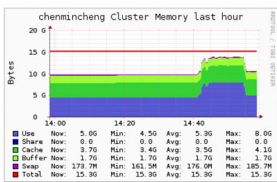
PARAPR-MR

(3) Network utilization of the cluster

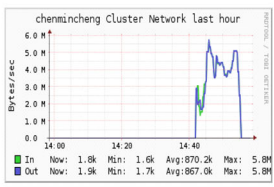
(b) DS2



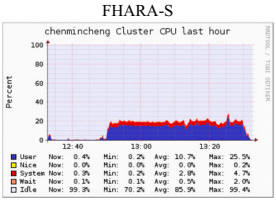
FHARA-S



FHARA-S

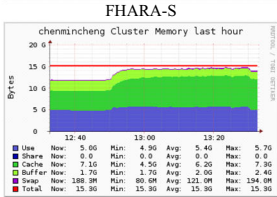


FHARA-S



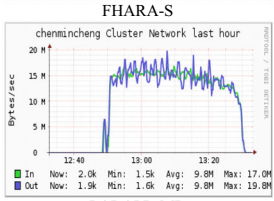
PARAPR-MR

(1) CPU utilization of the cluster



PARAPR-MR

(2) Memory utilization of the cluster



PARAPR-MR

(3) Network utilization of the cluster

(c) DS4

Fig. 6 Comparisons of resource utilization

16:19-16:25, and PARAPR-MR is 19:43-20:01. Figure 6c shows the results on DS6, the running time of FHARA-S is 14:42-14:54, and PARAPR-MR is 12:50-13:31. We find that the CPU utilization of them is all below 30% during the application execution. However, the execution time of PARAPR-MR is much longer than FHARA-S. PARAPR-MR consumes more CPU resource than FHARA-S in general. The memory utilization of FHARA-S increases significantly during its execution. Compared with PARAPR-MR, its memory utilization is only slightly improved. Therefore, FHARA-S consumes more memory resource. Then the network utilization of PARAPR-MR is much larger than that of FHARA-S. When the data size increases, the advantage of FHARA-S becomes more apparent, because PARAPR-MR transmits its intermediate results through HDFS, which needs lots of disk accesses and I/Os. Attribute reduction is an iterative algorithm, which requires repeated iteration. Therefore, this highlights the limitations of PARAPR-MR. We also find PARAPR-MR reduces the hard disk space sharply on account of mass intermediate results.

Based on the above results, FHARA-S not only extremely improves the computational efficiency, but also greatly saves the computational resources when compared with the traditional reduction algorithm PARAPR-MR.

7.2 Performance comparison with attribute reduction algorithms using Spark

Compared with the traditional attribute reduction algorithm using MapReduce, our algorithm takes full advantage of the computing platform. But it cannot reflect the details of our algorithm's improvement. So in addition to our FHARA-S, we also have realized two kinds of other attribute reduction algorithms using Spark. These baseline algorithms are extensively researched, and they also show good performance. We list them below:

① **FHARA-S:** The algorithm proposed in our paper.

② **PARAPR-S:** Parallel attribute reduction algorithms based on positive region using Spark, which is implemented on the basis of PR algorithm proposed in the literature [47]. But PR is implemented by MapReduce; PARAPR-S is implemented by Spark.

③ **PARABR-S:** Parallel attribute reduction algorithms based on boundary region using Spark, which is implemented on the basis of PAAR-BR algorithm proposed in the literature [33]. But PAAR-BR is implemented by MapReduce; PARABR-S is implemented by Spark.

Figure 7 shows the comparison of FHARA-S, PARAPR-S and PARABR-S. We use three different arrows to indicate the execution flows of these algorithms. Compared to FHARA-S, other 2 algorithms cannot simplify the decision table constantly and cut the search space down with the assistance of heuristic information.

7.2.1 Running time comparison

These 3 parallel attribute reduction algorithms are executed on Spark cluster, and we record their running time separately. Figure 8 reveals the whole running time of these algorithms on DS1–DS8. We can discover that FHARA-S promotes around 0.61X performance improvement over PARABR-S and PARAPR-S on various kinds of data sets. Through their performance on the data set DS3–DS5, we discover that PARAPR-S consumes more time than the other two algorithms as the attribute dimension increases (e.g., FHARA-S promotes around 0.89X and 1.04X performance improvement over PARABR-S on DS4 and DS5, respectively, which far exceeds the average level of improvement). This is because PARAPR-S requires to calculate plenty of positive region objects as the number of selected

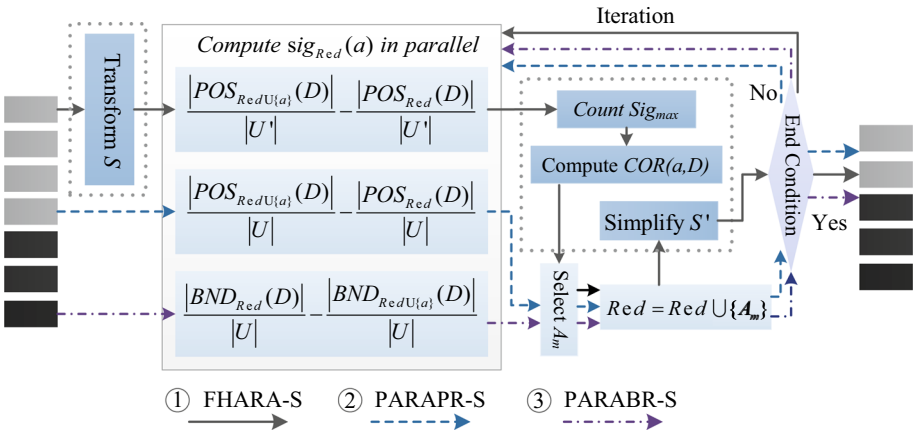


Fig. 7 Comparison of FHARA-S, PARAPR-S and PARABR-S

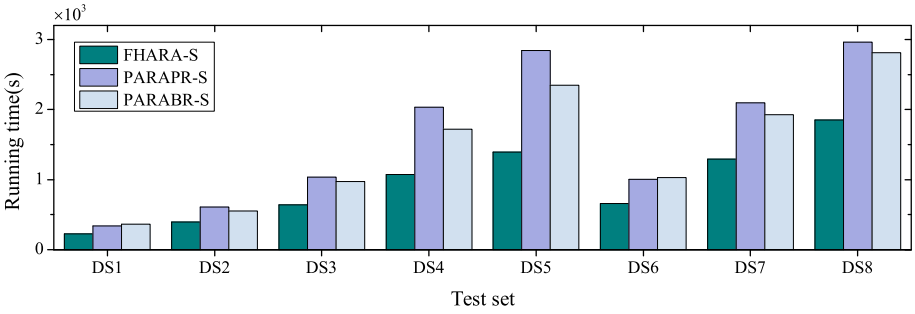


Fig. 8 Comparisons of three parallel algorithms using Spark for eight data sets

attributes grows. When the attribute dimensions of the data sets are not high, the similar performance is exhibited by PARABR-S and PARAPR-S to the whole running time, especially for DS1–DS3.

7.2.2 Iteration time comparison

In the following, in order to further study the execution details of the algorithms, each iteration time of PARAPR-S, PARABR-S and FHARA-S is also examined. Figure 9 reveals the first fourteen iterations of these three algorithms on DS1–DS8. We can find out that PARAPR-S and PARABR-S reveal a similar pattern of decline in the iteration time. Compared to them, more time is expended by the second, third and fourth iteration of FHARA-S than PARAPR-S and PARABR-S, as it costs time to calculate new U' and update S'; more importantly, the quantity of U' has not yet been cut down. Since FHARA-S simplifies the S' constantly and cuts the search space down, the 5th–14th iteration's execution time decreases significantly. The 14th iteration needs only about a tenth of the time of PARAPR-S and PARABR-S. By observing the experimental results on DS3–DS5, we also find that when increasing the number of the attributes, the first three iterations of FHARA-S consume much more time. Furthermore, each iteration of PARAPR-S consumes significantly more time than PARABR-S as the dimension of attributes increasing. These experiments show that the heuristic formula

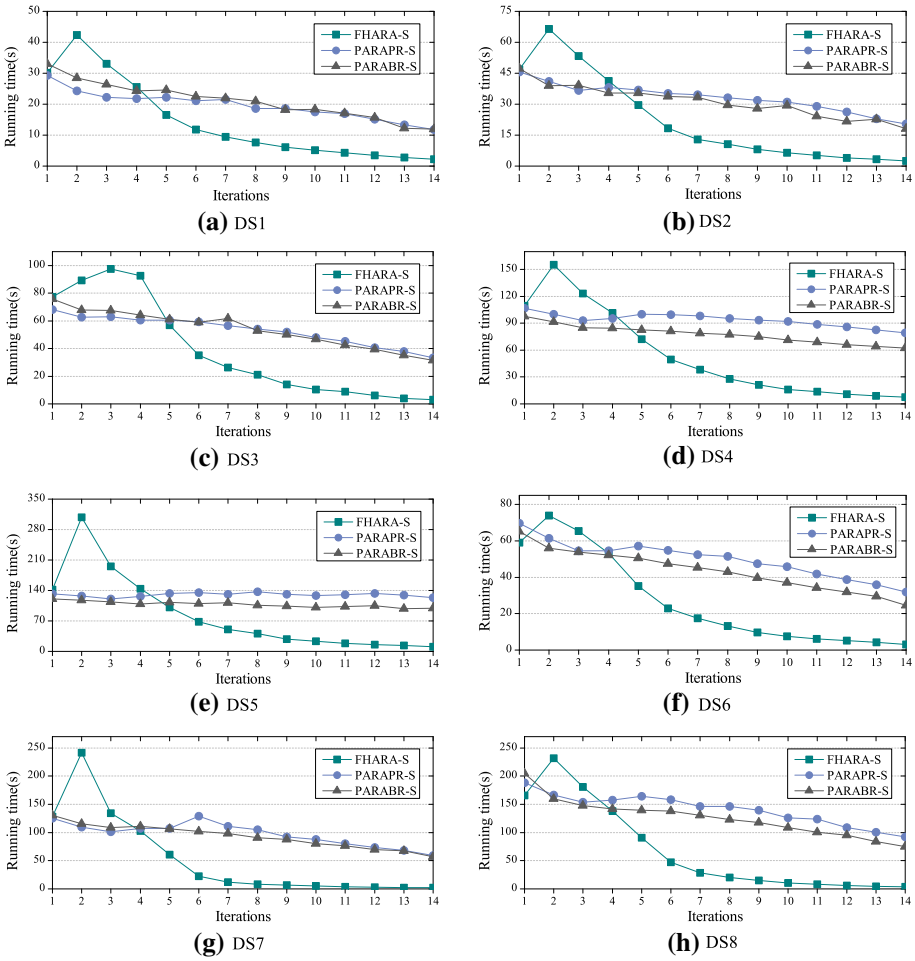


Fig. 9 Comparisons of three algorithms for iterations on DS1–DS8

which measures the significance of the attribute can promote the computation efficiency as well.

7.3 Case study

As a case study, we first randomly extract 10% of the data for DS1–DS8 as experimental data sets. Then we utilize the our algorithm(FHARA-S) to process them. Each experimental data set after reduction is divided into two parts randomly. One part of the data set is regarded as the training data for building the random forest classification model and the other part is as testing data for evaluating the model. We also use the original data sets to do contrast experiments.

Figure 10 shows the comparison of classification accuracy. We can discover that the accuracy of them is similar on the real data sets DS1–DS3. But the model established through the data after processing still has a higher accuracy than the model established through the

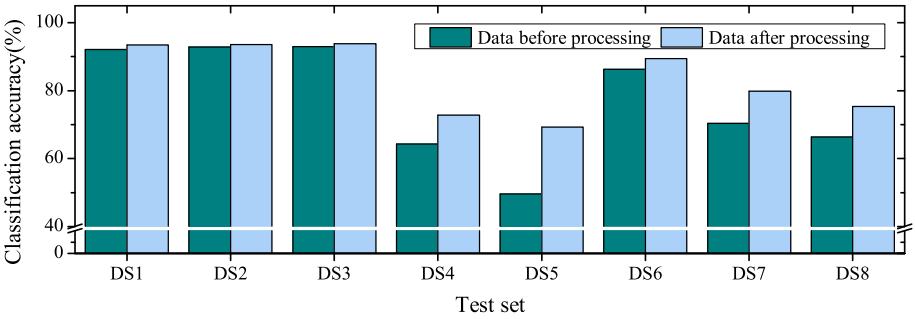


Fig. 10 Comparison of classification accuracy

data before processing. This is because the real data have high quality and little noise data. Our reduction algorithm only removes redundant data and some noise data. The data sets DS4–DS8 have a lot of synthetic data. Their data quality is low. The classification accuracy of the model established through the data after processing has greatly improved. This shows that when the data quality is not high, through our reduction algorithm we can improve the quality of data, thereby enhancing the value of data applications.

8 Conclusion

As the era of cloud computing and big data is coming, large-scale green data centers have been deployed widely all over the world. This has led to a sharp increase in energy data. The energy data always have many redundant and unnecessary attributes; thus, attribute reduction becomes a significant step. In this paper, we extend the methodology of rough sets to construct data center energy consumption knowledge representation system firstly. Then we take good advantage of in-memory computing to propose an attribute reduction algorithm for energy data using Spark. In this algorithm, we utilize an efficient algorithm for transforming energy consumption decision table, a heuristic formula for measuring the significance of attribute to reduce the search space, and introduce the correlation between condition attribute and decision attribute to further improve the computational efficiency. The adaptive decision management architecture based on FHARA-S is also designed for the green data center, which can improve decision-making efficiency and strengthen energy management. The experimental results demonstrate that the speed of our algorithm gains up to 2.2X performance improvement over the traditional attribute reduction algorithm using MapReduce and 0.61X performance improvement over the algorithms using Spark. Besides, our algorithm also saves much computation resources. Furthermore, the parallelization of other extended rough set models will be studied in the future.

Acknowledgements This research project is supported by the National Natural Science Foundation of China (Grant No: 61303029), National Social Science Foundation of China (Grant No: 15BGL048), Hubei Province Science and Technology Support Project (Grant No: 2015BAA072), the Fund for Creative Research Group of the Key Natural Science Foundation of Hubei Province of China (Grant No: 2017CFA012), the Key Technical Innovation Project of Hubei (Grant No: 2017AAA122).

References

1. Anderson MR, Cafarella M (2016) Input selection for fast feature engineering. In: 2016 IEEE 32nd international conference on data engineering (ICDE). IEEE, pp 577–588
2. Armbrust M, Xin RS, Lian C, Huai Y, Liu D, Bradley JK, Meng X, Kaftan T, Franklin MJ, Ghodsi A, et al (2015) Spark sql: relational data processing in spark. In: Proceedings of the 2015 ACM SIGMOD international conference on management of data. ACM, pp 1383–1394
3. Bennasar M, Hicks Y, Setchi R (2015) Feature selection using joint mutual information maximisation. Expert Syst Appl 42(22):8520–8532
4. Chen D, Yang Y, Dong Z (2016a) An incremental algorithm for attribute reduction with variable precision rough sets. Appl Soft Comput 45:129–149
5. Chen H, Li T, Cai Y, Luo C, Fujita H (2016b) Parallel attribute reduction in dominance-based neighborhood rough set. Inf Sci 373:351–368
6. Chen M, Yuan J, Li L, Liu D, Li T (2017) A fast heuristic attribute reduction algorithm using spark. In: 2017 IEEE 37th international conference on distributed computing systems (ICDCS). IEEE, pp 2393–2398
7. Chen YS, Cheng CH (2010) Forecasting pgr of the financial industry using a rough sets classifier based on attribute-granularity. Knowledge and information systems 25(1):57–79
8. Chen YS, Cheng CH (2013) Application of rough set classifiers for determining hemodialysis adequacy in esrd patients. Knowl Inf Syst 34(2):453–482
9. Czolombitko M, Stepaniuk J (2016) Attribute reduction based on mapreduce model and discernibility measure. In: IFIP International conference on computer information systems and industrial management. Springer, pp 55–66
10. Ding W, Lin CT, Chen S, Zhang X, Hu B (2018) Multiagent-consensus-mapreduce-based attribute reduction using co-evolutionary quantum pso for big data applications. Neurocomputing 272:136–153
11. El-Alfy ESM, Alshammari MA (2016) Towards scalable rough set based attribute subset selection for intrusion detection using parallel genetic algorithm in mapreduce. Simul Model Pract Theory 64:18–29
12. Fiandrino C, Kliazovich D, Bouvry P, Zomaya AY (2015) Performance and energy efficiency metrics for communication systems of cloud computing data centers. IEEE Trans Cloud Comput 1–1
13. García S, Luengo J, Herrera F (2016) Tutorial on practical tips of the most influential data preprocessing algorithms in data mining. Knowl Based Syst 98:1–29
14. Hu J, Pedrycz W, Wang G, Wang K (2016) Rough sets in distributed decision information systems. Knowl Based Syst 94(C):13–22
15. Hu Q, Zhang L, Zhou Y, Pedrycz W (2018) Large-scale multimodality attribute reduction with multi-kernel fuzzy rough sets. IEEE Trans Fuzzy Syst 26(1):226–238
16. Iquebal AS, Pal A, Ceglarek D, Tiwari MK (2014) Enhancement of mahalanobis-taguchi system via rough sets based feature selection. Expert Syst Appl 41(17):8003–8015
17. Jiang F, Sui Y (2015) A novel approach for discretization of continuous attributes in rough set theory. Knowl Based Syst 73:324–334
18. Jing Y, Li T, Fujita H, Yu Z, Wang B (2017) An incremental attribute reduction approach based on knowledge granularity with a multi-granulation view. Inf Sci 411:23–38
19. Khayyat Z, Ilyas IF, Jindal A, Madden S, Ouzzani M, Papotti P, Quiané-Ruiz JA, Tang N, Yin S (2015) Bigdansing: a system for big data cleansing. In: Proceedings of the 2015 ACM SIGMOD international conference on management of data. ACM, pp 1215–1230
20. Ko YC, Fujita H, Tzeng GH (2013) A fuzzy integral fusion approach in analyzing competitiveness patterns from wcy2010. Knowl Based Syst 49:1–9
21. Li C, Qouneh A, Li T (2012) iswitch: coordinating and optimizing renewable energy powered server clusters. In: 2012 39th annual international symposium on computer architecture (ISCA). IEEE, pp 512–523
22. Li C, Hu Y, Zhou R, Liu M, Liu L, Yuan J, Li T (2013a) Enabling datacenter servers to scale out economically and sustainably. In: Proceedings of the 46th annual IEEE/ACM international symposium on microarchitecture. ACM, pp 322–333
23. Li C, Zhou R, Li T (2013b) Enabling distributed generation powered sustainable high-performance data center. In: 2013 IEEE 19th international symposium on high performance computer architecture (HPCA2013). IEEE, pp 35–46
24. Liang J, Wang F, Dang C, Qian Y (2012) An efficient rough feature selection algorithm with a multi-granulation view. Int J Approx Reason 53(6):912–926
25. Liang J, Wang F, Dang C, Qian Y (2014) A group incremental approach to feature selection applying rough set technique. IEEE Trans Knowl Data Eng 26(2):294–308
26. Liu G, Shen H (2016) Minimum-cost cloud storage service across multiple cloud providers. In: 2016 IEEE 36th international conference on distributed computing systems (ICDCS). IEEE, pp 129–138

27. Lu Z, Qin Z, Zhang Y, Fang J (2014) A fast feature selection approach based on rough set boundary regions. Pattern Recognit Lett 36(1):81–88
28. Ma Y, Yu X, Niu Y (2015) A parallel heuristic reduction based approach for distribution network fault diagnosis. Int J Electr Power Energy Syst 73:548–559
29. Ouyang X, Irwin D, Shenoy P (2016) Spotlight: An information service for the cloud. In: 2016 IEEE 36th international conference on distributed computing systems (ICDCS). IEEE, pp 425–436
30. Pacheco F, Cerrada M, Sánchez RV, Cabrera D, Li C, de Oliveira JV (2017) Attribute clustering using rough set theory for feature selection in fault severity classification of rotating machinery. Expert Syst Appl 71:69–86
31. Pawlak Z (1982) Rough sets. Int J Parallel Program 11(5):341–356
32. Pawlak Z, Skowron A (2007) Rough sets: some extensions. Inf Sci 177(1):28–40
33. Qian J, Miao D, Zhang Z, Yue X (2014) Parallel attribute reduction algorithms using mapreduce. Inf Sci 279:671–690
34. Qian J, Lv P, Yue X, Liu C, Jing Z (2015) Hierarchical attribute reduction algorithms for big data using mapreduce. Knowl Based Syst 73:18–31
35. Ramírez-Gallego S, García S, Mouriño-Talín H, Martínez-Rego D, Bolón-Canedo V, Alonso-Betanzos A, Benítez JM, Herrera F (2016) Data discretization: taxonomy and big data challenge. Wiley Interdiscip Rev Data Min Knowl Discov 6(1):5–21
36. Song S, Zhu H, Wang J (2016) Constraint-variance tolerant data repairing. In: Proceedings of the 2016 ACM SIGMOD international conference on management of data. ACM, pp 877–892
37. Venkataraman S, Yang Z, Liu D, Liang E, Falaki H, Meng X, Xin R, Ghodsi A, Franklin M, Stoica I, Zaharia M (2016) Sparkr: scaling r programs with spark. In: Proceedings of the 2016 ACM SIGMOD international conference on management of data. ACM, pp 1099–1104
38. Wang F, Liang J (2016) An efficient feature selection algorithm for hybrid data. Neurocomputing 193(C):3341
39. Wang X, Wang T, Junhai Z (2012) An attribute reduction algorithm based on instance selection. J Comput Res Dev 49(11):2305–2310
40. Wei W, Liang J, Qian Y, Wang F (2009) An attribute reduction approach and its accelerated version for hybrid data. In: IEEE international conference on cognitive informatics (ICCI 2009), 15–17 June, 2009, Hong Kong, China, pp 167–173
41. Xie X, Qin X (2018) A novel incremental attribute reduction approach for dynamic incomplete decision systems. Int J Approx Reason 93:443–462
42. Xu Z, Liu Z, Yang b, wei S (2006) A quick attribute reduction algorithm with complexity of max $(o(|c||u|), o(|c|^2|u/c|))$. Chin J Comput 29(3):391–399
43. Yuan J, Zhong L, Yang G, Chen M, Gu J, Li T (2015) Towards filling and classification of incomplete energy big data for green data centers. Chin J Comput 38(12):2499–2516
44. Yuan J, Chen M, Jiang T, Li T (2017) Complete tolerance relation based parallel filling for incomplete energy big data. Knowl Based Syst 132:215–225
45. Zaharia M, Chowdhury M, Das T, Dave A, Ma J, McCauley M, Franklin MJ, Shenker S, Stoica I (2012) Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing. In: Proceedings of the 9th USENIX conference on networked systems design and implementation. USENIX Association, pp 2–2
46. Zhang CJ, Chen L, Tong Y, Liu Z (2015a) Cleaning uncertain data with a noisy crowd. In: 2015 IEEE 31st international conference on data engineering. IEEE, pp 6–17
47. Zhang J, Li T, Pan Y (2013) Plar: Parallel large-scale attribute reduction on cloud systems. In: International conference on parallel and distributed computing, applications and technologies, pp 184–191
48. Zhang J, Li T, Chen H (2014a) Composite rough sets for dynamic data mining. Inf Sci 257:81–100
49. Zhang J, Wong JS, Li T, Pan Y (2014b) A comparison of parallel large-scale knowledge acquisition using rough set theory on different mapreduce runtime systems. Int J Approx Reason 55(3):896–907
50. Zhang J, Wong JS, Pan Y, Li T (2015b) A parallel matrix-based method for computing approximations in incomplete information systems. IEEE Trans Knowl Data Eng 27(2):326–339
51. Zheng K, Hu J, Zhan Z, Ma J, Qi J (2014) An enhancement for heuristic attribute reduction algorithm in rough set. Expert Syst Appl 41(15):6748–6754
52. Zliobaite I, Gabrys B (2014) Adaptive preprocessing for streaming data. IEEE Trans Knowl Data Eng 26(2):309–321

Mincheng Chen received his B.S. and M.S. degrees in computer science from Wuhan University of Technology, China, in 2014 and 2016, respectively. He is currently a Ph.D. candidate in the School of Computer Science and Technology, Wuhan University of Technology. His research interests mainly include distributed computing, rough set and machine learning.

Jingling Yuan received her Ph.D. in computer science from Wuhan University of Technology, China, in 2004. She is currently the Professor and Ph.D. supervisor in Wuhan University of Technology. She visited University of Florida, Gainesville, as a visiting scholar between June 2008 and June 2009. Dr. Yuan's research interests mainly include but not limited to machine learning, data mining and green computing.

Lin Li received her Ph.D. in computer science from University of Tokyo, Japan, in 2009. She is currently the Professor and Ph.D. supervisor in Wuhan University of Technology. She visited University of Technology, Sydney, as a visiting scholar between February 2014 and February 2015. Dr. Li's research interests mainly include but not limited to machine learning, text mining, information retrieval and recommender system.

Dongling Liu received his B.S. degree in computer science from Wuhan University of Technology, China, in 2016. He is currently an M.S. candidate in the School of Computer Science and Technology, Wuhan University of Technology. His research interests mainly include distributed computing and machine learning.

Yang He received his B.S. degree in computer science from Wuhan University of Technology, China, in 2017. He is currently an M.S. candidate in the School of Computer Science and Technology, Wuhan University of Technology. His research interests mainly include machine learning and transfer learning.

Affiliations

Mincheng Chen[1] · Jingling Yuan[1] · Lin Li[1] · Dongling Liu[1] · Yang He[1]

Mincheng Chen
wester589@163.com

Lin Li
cathylilin@whut.edu.cn

Dongling Liu
darlingliu@whut.edu.cn

Yang He
yanghe@whut.edu.cn

[1] School of Computer Science and Technology, Wuhan University of Technology, Wuhan 430070, China