International Journal of Signal Processing, Image Processing and Pattern Recognition
Vol. 10, No. 12 (2017), pp.63-74
http//dx.doi.org/10.14257/ijsip.2017.10.12.05

A Partition-based Attribute Reduction Data Mining Approach for Multi-dimensional Datasets

Xianyang Li[1, 2], Guihua Qiu[1, 2]* and Anshan Lu[1, 2]

[1]*College of Electronics and Information Engineering, Qinzhou University, Guangxi, China*
[2]*Key Laboratory for Electronic Devices Inspection, Qinzhou, Guangxi, China*
5579934@qq.com

Abstract

Data is critical for decision-makings, however, data from multiple sources are difficult to link, match, cleanse and transform data across systems. This paper takes decision tables as an example which is a precise yet compact way to model complex rule sets and their corresponding actions, which are usually used for processing large number of data associate conditions. A partition-based attribute reduction (PAE) enabled data mining approach for multiply dimensional decision tables (MDT) is presented. Experiments are carried out for comparing the proposed algorithm with ARABCE (Attribute Reduction Algorithm Based on Condition Entropy) approach in terms of computational time, reduced attribute quantity, and precision ratio. It is found that, through the experimental results, the precision ratio is higher than that of ARABCE that because the proposed algorithm uses partition-based mechanism which can figure out the associate rules with fast iteration processes. And when the records reach a certain number (4,000,000 pieces in this experiment), the memory units are required significantly. Buffer overrun thus needs to be considered if this approach is implemented in practice.

Keywords*: Big Data, Algorithm, Attribute Reduction, Partition Mechanism*

1. Introduction

It is been a big year for Big Data as more and more businesses are accepting that data is critical for decision-makings. Today's data comes from multiple sources, which makes it difficult to link, match, cleanse and transform data across systems. However, it's necessary to connect and correlate relationships, hierarchies and multiple data linkages or your data can quickly spiral out of control [1]. For example, Data streams in at an unprecedented speed and must be dealt with in a timely manner like RFID tags, sensors and smart metering are driving the need to deal with torrents of data in near-real time [2]. Big data is a term that describes the large volume of data – both structured and unstructured – that inundates a business on a day-to-day basis [3]. But it's not the amount of data that's important. It's what organizations do with the data that matters. Big data can be analyzed for insights that lead to better decisions and strategic business moves.

Decision tables are a precise yet compact way to model complex rule sets and their corresponding actions, which are usually used for processing large number of data associate conditions with actions to perform [4]. Each decision corresponds to a variable, relation or predicate whose possible values are listed among the condition alternatives and each action is a procedure or operation to perform, and the entries specify whether (or in what order) the action is to be performed for the set of condition alternatives the entry corresponds to [5]. Many decision tables include in their condition alternatives which don't care symbol, a hyphen. Using that can simplify decision tables, especially when a

Received (March 7, 2017), Review Result (November 23, 2017), Accepted (Decmber 1, 2017)

ISSN: 2005-4254 IJSIP

given condition has little influence on the actions to be performed, for example in some cases, entire conditions thought to be important initially are found to be irrelevant when none of the conditions influence which actions are performed. Aside from the basic four quadrant structure, decision tables vary widely in the way the condition alternatives and action entries are represented by using simple true/false values to represent the alternatives to a condition (akin to if-then-else), other tables may use numbered alternatives (akin to switch-case), and some tables even use fuzzy logic or probabilistic representations for condition alternatives [6]. In a similar way, action entries can simply represent whether an action is to be performed (check the actions to perform), or in more advanced decision tables, the sequencing of actions to perform (number the actions to perform).

In order to reduce the attribute in decision tables, the positive region should be figured out. A wide used approach is partition-based algorithm [7-9]. However, for big dataset which includes enormous texts, it is difficult to calculate the positive region precisely and efficiently. For example, the average time complexity for multi-dimensional decision table is $O(|U|\times(|C|+\log|U|))$ and space complexity is $O(|U|)$ [10]. This paper introduces a fast sorting algorithm using partition-based attribute reduction mechanism to calculate the positive region. Basic definitions are proposed according to specific research problems and associated lemmas are proposed for the reduction algorithm. In order to test the feasibility and effectiveness of the proposed algorithm, experiments are carried out for comparing with ARABCE (Attribute Reduction Algorithm Based on Condition Entropy) approach. The rest of this paper follows: Section 2 give some definitions. Section 3 introduces the proposed algorithm and analysis. Section 4 demonstrates the experimental results and analysis of complexities. Section 5 concludes this paper by giving our key findings and future research directions.

2. Definitions

This section gives some definitions. Let $S=<U, A=C\cup D, V, f>$ is a decision table. The attribute sequence is defined as $SO: c_1 \prec c_2 \prec ... \prec c_{|c|}$. Assume M is a Skowron differentiate matrix of S. Thus, we can get the necessary and sufficient condition for the elements B_{xy}^S in M are not null is:

$$(x\in Pos_C(D)\wedge y\in Pos_C(D)\wedge(d(x)\neq d(y))$$
$$\vee(x\in Pos_C(D)\wedge y\notin Pos_C(D))$$
$$\vee(x\notin Pos_C(D)\wedge y\in Pos_C(D))$$

Let $[k]\in M/L(S)$, the necessary and sufficient condition for $[k]\neq\varnothing$ is $\exists x, y\in U$, the following two conditions should be met: firstly, B_{xy}^S in M are not null and secondly $\forall_{k_1(1\le k_1<k)}(C_{k_1}(x)=c_{k_1}(y))\wedge(c_k(x)\neq c_k(y))$. Given an attribute sequence $SO: c_1 \prec c_2 \prec ... \prec c_{|c|}$, the necessary and sufficient condition for $c_k(c_k\in C\wedge 2\le k\le|C|)$ is not null is after the Skowron differentiate matrix remove the elements including $c_1, c_2, ..., c_{k-1}$, $\exists B_{xy}^S\in M$ satisfies $c_k\in B_{xy}^S$.

Based on the definitions, the following texts present fast reduction algorithms for the attribute processing. These algorithms are used for the final proposed approach. Firstly, the non-null attribute set should be worked out by NonEmptyAttr () function.

Void NonEmptyAttr(int r, Ojectset Oset)
{
While $(1 \le r \le |C|)$ *//r is the attribute number, Oset is an object set.*
If $|Oset|=1$, *then Return;*
Else if $\forall_{x \in Oset} x \notin POS_c(D)$, *then Return;*
Else if $\forall_{x,y \in Oset} d(x) = d(y)$, *then Return;*
Else if $\forall_{x,y \in Oset} c_r(x) = c_r(y)$, *then NonEmptyAttr(r+1,Oset);*
Else $\exists_{x,y \in Oset} c_i(x) \ne c_i(y)$, *then NonEmptyLabel [r]=1;*
$Oset=\{ Oset_1^r, Oset_2^r \}; \forall x \in Oset_1^r; \forall y \in Oset_2^r;$
$c_r(x) \le \overline{X} < c_r(y)$;
NonEmptyAttr(r, $Oset_1^r$ *);*
NonEmptyAttr(r, $Oset_2^r$ *);*
Endif
endwhile
}

After getting the non-null attributes set, the following algorithm is used for working out the equal class non-null attribute sets according to $L(SO)$.

Input: Decision table $S =< U, A = C \cup D, V, f >$
Output: Non-null attribute set R_1 *of* S
Step 1: let $C = \{c_1, c_2, ...c_{|C|}\}$, *for a given* $SO: c_1 \prec c_2 \prec ... \prec c_{|C|}$;
Step 2: $R_1 = \varnothing$, $r = 1$, $Oset_1^1 = U$;
Step 3: for $j = 1$ *to* $|C|$, *do NonEmptyLabel* $[j] = 0$; *end for;*
Step 4: Call NonEmptyAttr(1, $Oset_1^1$*);*
Step 5: for $j = 1$ *to* $|C|$, *do*
If (NonEmptyLabel $[j] == 1$*) then*
$R_1 = R_1 + \{c_j\}$
Endif
Endfor
Step 6: Return R_1

Considering the complexity of the above algorithm, the average time and space complexities are $O(|U| \times (|C| + \log |U|))$ and $O(|U| + |C|)$.

Based on the definitions and algorithm, a reduction algorithm is proposed for the dataset. The details are given as follows:

Input: Decision table $S =< U, A = C \cup D, V, f >$
Output: reduced attribute Red of S
Step 1: let $C = \{c_1, c_2, ...c_{|C|}\}$, *for a given* $SO: c_1 \prec c_2 \prec ... \prec c_{|C|}$;
Step 2: $r = 1$; $U_1^1 = U$, *Red=* $\varnothing$;
Step 3: Calculate the positive region $POS_C(D)$ *and* R_1;
Step 4: let $c_{N'}$ *is the maximum attribute of* R_1;
If $c_{N'} \in$ *Red, then go to Step 5;*
Else

$Red=Red\cup\{c_{N'}\}$;
$R_1=R_1-\{c_{N'}\}$;
$C'=\varnothing$;
Emerge attribute Red and R_1, $C'=\{Red|Dec\}$ *and* $C'=\{R_1|Inc\}$;
$C=\varnothing$; $C=C'$;
Use the above algorithm to calculate R_1;
Endif
Step 5: Return Red.

From the above algorithm, the average time complexity for step 3 is $O(|U|\times(|C|+\log|U|))$ and space complexity is $O(|U|)$. Step 4 has the average time complexity $O(|U|\times(|C|+\log|U|))$ and space complexity $O(|U|+|C|)$. Step 5 has the average time complexity $O(|U|\times|C|\times(|C|\times\log|U|))$ and space complexity $O(|U|+|C|)$. Thus, the average time and space complexities of algorithm are $O(|U|\times|C|\times(|C|\times\log|U|))$ and $O(|U|+|C|)$ respectively.

3. Partition-based Attribute Reduction Algorithm

3.1. Proposed Algorithm

Based on the definitions, several lemma are presented for the proposed algorithm in this paper.

Lemma 1: For a decision table $S=<U,A=C\cup D,V,f>$, $\forall x\in POS_C(D)$, x generates a deterministic rule in $C\cup D$.

Lemma 2: For a decision table $S=<U,A=C\cup D,V,f>$, DR is the deterministic rule set of S. $\forall_{d_i\in DR(1\le i\le|DR|)}d_i$ is created by $x(x\in POS_C(D))$.

Lemma 3: For a decision table $S=<U,A=C\cup D,V,f>$, let DR is the deterministic rule set, $\forall x\in POS_C(D)$, d_1 is the decision rule driven by x, $M_1=\{B_{xy}^S\,|\,y\in U\}$, then $\forall\alpha\in M_1(M_1\neq\varnothing)$, $\exists c\in d_1\,|\,C$ meets $c\in\alpha$. This is the partition mechanism which could be used in the reduction algorithm for improving the efficiency and effectiveness when facing large number of datasets [11].

Based on the lemma, the proposed partition-based attribute reduction algorithm is given as follows:
Input: Decision table $S=<U,A=C\cup D,V,f>$
Output: Deterministic rule set DR
Step 1: initialize $DR=\varnothing$ *and* $Aset=\varnothing$;
Step 2: based on the definition, decomposition of U *in condition attribute* C, *calculate* $POS_C(D)$ *in* S;
Step 3: Assume $SO:c_1\prec c_2\prec\ldots\prec c_m$ $m=|C|$, *calculate the non-null attribute set and save in* $Aset$;
Step 4: let $Aset=\{c_1^{'},c_2^{'},\ldots,c_{|ASet|}^{'}\}$ *and* $c_1^{'}\prec c_2^{'}\prec\ldots\prec c_{|ASet|}^{'}$
If $|ASet|=1$, *go to step 5;*

Else

For $i = |ASet|$ *to 1 do*

Step: 4.1: for $j = 1$ *to* $|U|$ *do*

CoreValueAttr[j]=0;

Endfor

In the attribute set $(ASet - c_i^{'})$*, follow* $c_1^{'} \prec c_2^{'} \prec ... \prec c_{|ASet|}^{'}$ *execute step 4.2 and step 4.3*

Step 4.2: in the attribute $c_i^{'}$*, carry the decomposition operation in* U *with* $V_1^{'}$*, divide* U *into* $|V_1^{'}|$*:* $U_1, U_2, ..., U_{|V_1^{'}|}$ *which meets:* $(1 \le j \le |V_i^{'}|)$

1) $\forall_{x \in U_j} \forall_{y \in U_j} ((c_1^{'}(x) = null) \vee (c_1^{'}(y) = null) \vee (c_1^{'}(x) = c_1^{'}(y)))$*;*

2) $\forall_{x \in U} ((c_1^{'}(x) = null) \Rightarrow \forall_{U_j (1 \le j \le |V_i^{'}|)} x \in U_j)$*;*

3) $\forall_{x \in U_j} \forall_{y \in U_k} ((c_1^{'}(x) \ne null) \wedge (c_1^{'}(y) \ne null) \Rightarrow (c_1^{'}(x) \ne c_1^{'}(y)))$*;*

Step 4.3: follow $c_1^{'} \prec c_2^{'} \prec ... \prec c_{|ASet|}^{'}$*, carry the decomposition operation in the attribute set* $ASet - c_i^{'} - c_1^{'}$*, decompose the results from step 4.2* $U_1, U_2, ..., U_{|V_1^{'}|}$*;*

Step 4.4: assume that $ASet - c_i^{'}$*, after 4.2 and 4.3,* U *is decomposed into* w *parts:* $U_1, U_2, ..., U_w$*;*

For $j = 1$ *to* w *do*

If $\exists_{x \in U_j \wedge y \in U_j} (x \in POS_c(D) \vee y \in POS_c(D)) \wedge (d(x) \ne d(y))$ *in* U_j

then the objects are sequenced $q_1, q_2, ..., q_{|U_j|}$*;*

endif

for $k = 1$ *to* $|U_j|$ *do*

CoreValueAttr [q_k *]=1*

Endfor

Endfor

Step 4.5: remove some attributes in $c_i^{'}$

For $j = 1$ *to* $|U_j|$ *do*

If(CoreValueAttr[j] $\ne 1 \wedge x_j \in POS_C(D))$ *then*

$c_i^{'}(x_j) = NULL$*;*

Endif

Endfor

Endfor

Step 5: output the results

For $i = 1$ *to* $|U|$ *do*

If $x_i \in POS_C(D)$ *then*

In the $ASet$ *and decision set,* d_i *is created according to* x_i*;*

$DR = DR \cup \{d_i\}$

Endif

Endfor

Step 6: return DR

3.2. Complexity Analysis

In order to analyze the proposed algorithm, this section discuss the complexities. Assume that $|C|=m$ and $|U|=n$, the time complexity for step 1 is $O(m\times n)$. Step 2 and 3 have the average time complexity $O(n\times(m+\log n))$.

For step 4, the time complexity is $O(u\times(2n+T(1,n)))$, where $T(1,n)$ is the sum of step 4.2, 4.3 and 4.4. $T(1,n)$ is the result of calling $T(r,n)$,

$$T(r,n)=\begin{cases} 1, n=1 \\ 2n, r\geq u \\ 2n+T(r+1,n), None-decomposition \\ pn+T(r,n_1+n')+T(r,n_2+n')+...+t(r,n_p+n'), n_1+n_2+...+n'=n \\ n', Decomposible \end{cases}$$

Thus the complexity for step 4 is between $O(u^2\times n)$ and $T(u\times n\times p_1\times...\times p_u)$. Step 5 and 6 have the time complexity $O(m\times n)$. In a summary, the time complexity is among $\max(O(n\times(m+\log n))$, $O(u^2\times n)$ and $O(u\times n\times p_1\times...\times p_u)$.

4. Experiments and Discussions

4.1. Experiment 1: Evaluation of Feasibility

In order to evaluate the feasibility of the proposed algorithm, this section uses several simple decision tables to carry out the experiments. Assume there is a decision table $S=<U,A=C\cup D,V,f>$ as in table 1.

Table 1. Decision Table S

U	c_1	c_2	c_3	c_4	c_5
x_1	1	0	1	0	0
x_2	2	1	0	1	0
x_3	2	1	1	0	1
x_4	1	2	0	1	1
x_5	2	2	0	1	0

Using the proposed algorithm, the deterministic rule set could be worked out by the following steps:

Step 1: $DR=\varnothing$;

Step 2: calculate the positive region of S, $POS_C(D)=\{x_1,x_2,x_3\}$;

Step 3: assume there is an attribute sequence $c_1\prec c_2\prec c_3\prec c_4$, under the attribute, the non-null attribute set is $\{c_1,c_2,c_3\}$;

Step 4: according to the non-null attribute set, we can get a new decision table S_1;

Table 2. Decision Table S_1

U	c_1	c_2	c_3	D
x_1	1	0	1	0
x_2	2	1	0	0
x_3	2	1	1	1
x_4	1	2	0	1
x_5	2	2	0	0

From Table 2, c_3 has the maximum non attribute so that c_3 will be removed. U will be decomposed in attribute $\{c_1, c_2\}$. We can get $\{\{x_1\},\{x_2, x_3\},\{x_4, x_5\}\}$. Given the attribute, it is conclude that c_3 is the positive region object which makes x_2 and x_3 for a deterministic conditional attribute. For x_1, the attribute value is removed thus, null will be used. We can get the following table 3:

Table 3. Decision Table S_2

U	c_1	c_2	c_3	D
x_1	1	0	null	0
x_2	2	1	0	0
x_3	2	1	1	1
x_4	1	2	0	1
x_5	2	2	0	0

Finally, we can get that: remove the non-null attribute c_1, U will be decomposed in attribute $\{c_2, c_3\}$. Then we can get $\{\{x_1, x_2\},\{x_1, x_3\},\{x_2, x_4, x_5\},\{x_3\}\}$.

Step 5: for x_1, $d_1 : (c_1 = 1 \wedge c_2 = 0) \Rightarrow (D = 0)$;

For x_2, $d_2 : (c_1 = 2 \wedge c_3 = 0) \Rightarrow (D = 0)$;

For x_3, $d_3 : (c_1 = 2 \wedge c_3 = 1) \Rightarrow (D = 1)$;

Then $DR = \{d_1, d_2, d_3\}$

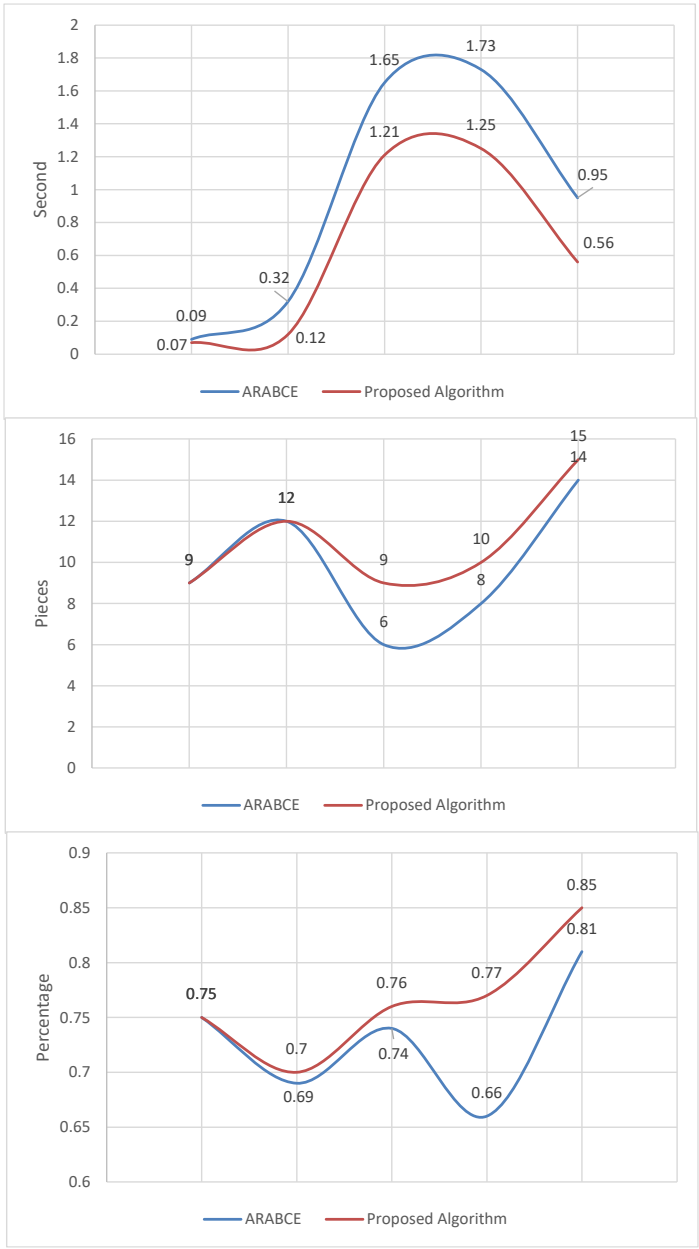

Figure 1. Comparison of T, Q, and P

4.2. Experiment 2: Testing Under Big Dataset

After testing the feasibility, this section reports on the testing under big dataset so that the algorithm's efficiency and effectiveness could be examined. We used the C++ programming to carry out the experiments with the datasets from KDDCUP 99 under the experiment environment: CPU: Intel ® Core i7-6700 @ 3.40GHz, RAM: 16.0 GB, System type: 64-bit Operating system, Windows 7 Enterprise.

Five datasets like Glass, Heart, Crx, Pima and Liver are picked up from UCI for the experiment 2. This section compares the proposed algorithm with ARABCE (Attribute Reduction Algorithm Based on Condition Entropy) approach in terms of computational time (T), reduced attribute quantity (Q), and precision ratio (P). Table 1 presents the results with specific records. Figure 1. illustrates the comparison of T, Q, and P with a graphical view and their trends.

Table 5. Comparison Results -1

Dataset	Record	Attribute	ARABCE			Proposed Algorithm		
			T	Q	P	T	Q	P
Glass	412	12	0.09	9	0.75	0.07	9	0.75
Heart	603	39	0.32	12	0.69	0.12	12	0.70
Crx	690	14	1.65	6	0.74	1.21	9	0.76
Pima	739	16	1.73	8	0.66	1.25	10	0.77
Liver	1620	24	0.95	14	0.81	0.56	15	0.85

Table 6. Pressure Testing Results

Records	T	Q	M
489686	61.25	38	198804
976843	135.62	36	246942
1348693	216.45	35	303652
1987348	289.66	35	356241
2568594	386.47	35	398224
3012458	468.20	35	446852
3569149	604.23	34	501268
4012359	667.15	34	578966
4589726	758.36	34	604879
5012448	849.65	32	701249

From Table 5, it is observed that, the proposed algorithm outperforms ARABCE generally. Figure 1 presents the outperformance in detail. For example, the computational time used by the proposed algorithm is 0.306 less than that of ARABCE averagely. And for the reduced attribute quantity, the proposed algorithm is able to work more attributes in Crx and Pima. The precision ratio as shown in the last diagram, is higher than that of ARABCE that because the proposed algorithm uses partition-based mechanism which can figure out the associate rules with fast iteration processes.

4.3. Discussions

In order to test the algorithm in pressure experiments, we used increasing records approach for testing the proposed algorithm. Table 6 presents the experimental results by increasing about 10% records each time. The targeted attribute quantity is set as 45 for each time. The computational time (T: Second), reduced attribute quantity (Q), and maximum used memory (M: KB) are presented in the following table.

From Table 2, it is observed that the processing time increases when the datasets are increasing. The increase follows linear trend. While, the reduced attribute quantity is decreasing slightly as the increasing of records. That because the algorithm has to spend more time for iteratively finding out the targeted attributes. As the increasing of records, the accuracy will be decreased. Additionally, as the increasing of records, the used memory increases as well. The later stage, for example the last three rows, the used memory increases sharply. It could be concluded that, when the records reach 4,000,000 pieces, the memory units are required significantly. That means for processing larger datasets, RAM should be considered in order to avoid the buffer overrun.

5. Conclusion

Reduction operation is one of key processing in the Big Data Analytics which is usually used for addressing large number of records which could be texts or numbers. This paper introduces a partition-based attribute reduction algorithm for big datasets, specifically for the decision tables. Basic definitions are given based on specific research problems and associated lemmas are proposed for the reduction algorithm. In order to test the feasibility and effectiveness of the proposed algorithm, experiments are carried out for comparing with ARABCE (Attribute Reduction Algorithm Based on Condition Entropy) approach in terms of computational time (T), reduced attribute quantity (Q), and precision ratio (P). Large number of datasets was used for testing the proposed algorithm too. Several key findings are significant for enriching the corresponding research area:

- It is found that, through the experimental results, the precision ratio is higher than that of ARABCE that because the proposed algorithm uses partition-based mechanism which can figure out the associate rules with fast iteration processes.
- It could be observed, when the records reach a certain number (4,000,000 pieces in this experiment), the memory units are required significantly. Buffer overrun thus needs to be considered if this approach is implemented in practice.

It is important to carry future research to overcome some limitations in this proposed approach. Firstly, this paper only consider single attribute properties. While, in practice, multiple properties may be applied. How to extend this algorithm to face the multi-attribute properties needs to be further investigated [12-14]. Secondly, this paper only considers the texts-based dataset. Other data such as images, GIS data, *etc* may be further included so that this algorithm can deal with more datasets [15].

Acknowledgement

Authors would like to acknowledge the support from Research on the Basic Skills Improvement for Young Teachers in Guangxi University in 2017, Short Texts Research on the Construction Technology of the Mass Organizations of Media Information Structures (No.2017KY0795); Qinzhou Science and Technology Project: Development of IoT-based Aquatic Environment Monitoring System Key Technology Research (No. 20164410); Qinzhou Electronic Product Testing Key Laboratory Open-project Grant.

References

[1] L. Y. Pang, R. Y. Zhong, J. Fang and G. Q. Huang, "Data-source interoperability service for heterogeneous information integration in ubiquitous enterprises", Advanced Engineering Informatics, vol. 29, **(2015)**, pp. 549-561.
[2] R. Y. Zhong, G. Q. Huang, S. L. Lan, Q. Y. Dai, C. Xu and T. Zhang, "A Big Data Approach for Logistics Trajectory Discovery from RFID-enabled Production Data", International Journal of Production Economics, vol. 165, **(2015)**, pp. 260-272.
[3] R. Clarke, "Big data, big risks", Information Systems Journal, vol. 26, **(2016)**, pp. 77-90.
[4] B. S. Yang, D.-S. Lim and A. C. C. Tan, "VIBEX: an expert system for vibration fault diagnosis of rotating machinery using decision tree and decision table", Expert Systems with Applications, vol. 28, **(2005)**, pp. 735-742.
[5] Y. Kato, T. Saeki and S. Mizuno, "Proposal of a statistical test rule induction method by use of the decision table", Applied Soft Computing, vol. 28, **(2015)**, pp. 160-166.
[6] S. H. Nguyen and M. Szczuka, "Feature Selection in Decision Systems with Constraints", in International Joint Conference on Rough Sets, **(2016)**, pp. 537-547.
[7] Y. Song and J. Luedtke, "An adaptive partition-based approach for solving two-stage stochastic programs with fixed recourse", SIAM Journal on Optimization, vol. 25, **(2015)**, pp. 1344-1367.

[8] R. Y. Zhong, Q. Y. Dai, T. Qu, G. J. Hu and G. Q. Huang, "RFID-enabled Real-time Manufacturing Execution System for Mass-customization Production", Robotics and Computer-Integrated Manufacturing, vol. 29, no. 2, **(2013)**, pp. 283-292.
[9] R. Y. Zhong, G. Q. Huang, Q. Y. Dai and T. Zhang, "Mining SOTs and Dispatching Rules from RFID-enabled Real-time Shopfloor Production Data", Journal of Intelligent Manufacturing. , vol. 25, **(2014)**, pp. 825-843.
[10] M. T. Elbatta and W. M. Ashour, "A dynamic method for discovering density varied clusters", International Journal of Signal Processing, Image Processing and Pattern Recognition, vol. 6, **(2013)**, p. 14.
[11] R. Y. Zhong, C. Xu, C. Chen and G. Q. Huang, "Big Data Analytics for Physical Internet-based intelligent manufacturing shop floors", International Journal of Production Research, vol. 55, **(2017)**, pp. 2610-2621.
[12] R. Y. Zhong, S. T. Newman, G. Q. Huang and S. L. Lan, "Big Data for supply chain management in the service and manufacturing sectors: Challenges, opportunities, and future perspectives", Computers & Industrial Engineering, vol. 101, **(2016)**, pp. 572-591.
[13] R. Y. Zhong, S. Lan, C. Xu, Q. Dai and G. Q. Huang, "Visualization of RFID-enabled shopfloor logistics Big Data in Cloud Manufacturing", The International Journal of Advanced Manufacturing Technology, vol. 84, **(2016)**, pp. 5-16.
[14] M. L. Wang, R. Y. Zhong, Q. Y. Dai and G. Q. Huang, "A MPN-based scheduling model for IoT-enabled hybrid flow shop manufacturing", Advanced Engineering Informatics, vol. 30, **(2016)**, pp. 728-736.
[15] Y. Zhang, G. Zhang, T. Qu, Y. Liu and R. Y. Zhong, "Analytical target cascading for optimal configuration of cloud manufacturing services", Journal of Cleaner Production, vol. 151, **(2017)**, pp. 330-343.

Authors

Xianyang Li, he received bachelor's degree in Computer Science and Technology from Gannan Normal University in 2002. From September 2002 to July 2014, he worked as lecturer in Ganzhou Teachers College China. From September 2014 to now, he worked as associate professor in Qinzhou University, Guangxi, China. His research interests include Computer Networks, Data Mining, and Education Technology. He has published several papers in international journals and conferences.

Guihua Qiu, he is an associate professor in Qinzhou University, Guangxi, China. His research areas include Computer Education, Big Data, and Network Security.

Anshan Lu, he is a professor in Qinzhou University, Guangxi, China. His research areas include Electronic Technology, Chaos Control and Synchronization Technologies.

