International Journal of Computer Science and Applications,
©Technomathematics Research Foundation
Vol. 13, No. 1, pp. 17 – 30, 2016

A MAPREDUCE SOLUTION FOR KNOWLEDGE REDUCTION IN BIG DATA

Weiping Cui*

School of Economics and Management, Beijing Jiaotong University, Beijing, 100000, Chian†
13113151@bjtu.edu.cn‡

Lei Huang

School of Economics and Management, Beijing Jiaotong University, Beijing, 100000, China
lhuang@bjtu.edu.cn

In order to deal with the data explosion and knowledge scarcity, we develop a parallel large-scale knowledge reduction method based on rough set for knowledge acquisition using MapReduce in this paper. It constructs the parallel algorithm framework model for knowledge reduction using MapReduce, which can be used to compute a reduction for the algorithms based on information entropy. The proposed method enables knowledge reduction algorithms to be applied over big data reduction problem without significant accuracy loss. The experimental results demonstrate that the proposed parallel knowledge reduction method can efficiently process massive datasets on Hadoop platform, with highly speed up the classification process and largely reduce the storage requirements. In all the experiments the introduced method based on information entropy is compared with the method based on positive region. The comparison clearly shows that the former method outperforms the latter one.

Keywords: Knowledge reduction; rough set; MapReduce; big data.

1. Introduction

In 2012, according to IBM, 1.5 quintillion bytes of data are created every day, which means that the 90% of the data created in the world has been produced in the last two years [López *et al.* (2015)]. Due to the fast-growing data in scientific and industrial areas, traditional data mining algorithms are facing the challenges from both the perspectives of data storage and computational complexity. The data growth of industry has rocketed the interest in effectively acquiring knowledge to analyze and predict trends in the last years. We have studied port data for several years. Port as an important node in the intersection of land and water logistics, is an important foundation and hub of the national economy. It produces large amounts of data in the daily production, but the method how to analysis this collection of large datasets and how to extract knowledge precisely from it are not available. In the actual port data analysis, factors influencing the final decision may not

be the target of all, how to identify these decisive factors in reducing the size of the data processing and improve the efficiency of data analysis plays an important role.

As indicated in [Han *et al.* (2011); Srinivasan *et al.* (2012)], it is difficult to construct enough memory to the main memory and explore effectively in the large search space with a single machine. As is known to all, not all these attributes are necessary or sufficient for decision-making. Irrelevant or redundant attributes not only increase the size of the search space, but also make generalization more complex [Guyon and Elisseeff (2003)]. Hence, Data reduction techniques [Pyle (1999)] emerged as preprocessing algorithms that aim to simplify and clean the big data, enabling data mining algorithms to be applied not only in a faster way, but also in a more accurate way by removing noisy and redundant data. Further more, attribute reduction is proposed as a preprocessing step in knowledge acquisition to find a minimum subset of attributes that provides the same descriptive or classification ability as the whole attributes [Qian *et al.* (2015)]. Framework "MapReduce for Prototype Reduction" [Triguero *et al.* (2015)] is proposed to be a suitable tool to enhance the performance of the nearest neighbor classifier with big data. Also, a parallel method for computing rough set approximations is given to process in data mining [Qian *et al.* (2013)]. These studies lay a solid foundation for further research, so we propose a novel method in this paper.

Rough set theory, introduced by Pawlak in1982 [Pawlak (1982)], has attracted a lot of attention recent years. It is regarded as an effective mathematical tool for dealing with inconsistent information in decision situations [Pawlak and Skowron (2007a); Pawlak and Skowron (2007b)]. In this framework, an attribute set is regarded as a granular space, which divides the entity into knowledge granules or elemental concepts. Partition, granulation, and approximation are the methods widely applied in human's reasoning. Rough set methodology presents a novel paradigm to deal with uncertainty and has been applied to feature selection, knowledge reduction, rule extraction, uncertainty reasoning, decision evaluation, and granular computing. Its effectiveness has been demonstrated in a number of successful applications in science and engineering fields. It plays an important role in the fields of international data mining, machine learning, pattern recognition and artificial intelligence during last decades [Błaszczyński *et al.* (2011); Chen *et al.* (2000); Cheng *et al.* (2010); Hu *et al.* (2007); Hu *et al.* (2008a); Hu *et al.* (2008b); Li *et al.* (2006); Li *et al.* (2007); Liou and Tzeng (2010); Qian *et al.* (2010)]. Knowledge reduction is a key step in knowledge acquisition in data mining, and the traditional knowledge reduction algorithms, such as the algorithms based on positive region, discernibility matrix or information entropy, etc., can not be loaded into the single massive amounts of data in memory [Qian *et al.* (2013)]. To overcome this limitation, the parallel knowledge reduction method is proposed [Deng *et al.* (2010)].

MapReduce, originally introduced by Google, a programming model and an associated implementation for processing and generating large data sets. It has been applied in data-intensive and computation-intensive tasks. It is quite novel, since it interleaves parallel and sequential computation, automatically parallelizes the computation across large-scale clusters of machines, and hides many system-level details. Processing of MapReduce can occur on data stored either in a filesystem (unstructured) or in a database (structured). MapReduce can take advantage of locality of data, processing it on or near the storage assets in order to reduce the distance over which it must be transmitted. Furthermore, MapReduce implementations offer their own

distributed file systems that provide a scalable mechanism for storing massive datasets [Ghemawat *et al.* (2003)].

Based on the above discussion, we propose a parallel large-scale knowledge reduction method based on rough set using MapReduce in this paper. Furthermore, the experiment demonstrates that the proposed parallel knowledge reduction method can efficiently process massive datasets using MapReduce, with highly speed up the classification process and largely reduce the storage requirements.

The rest of the paper is organized as follows. In Section 2, we provide some basic material about Rough set theory and MapReduce. In Section 3, we describe the knowledge reduction algorithm using MapReduce and make some improvements. We test the performance of our model and present the empirical results in Section 4. Finally, Section 5 summarizes the conclusions of the paper.

2. Basic Notions

In this section, we will introduce some notions of the Pawlak rough set model [Qian *et al.* (2011); Qian *et al.* (2015); Pawlak (1982); Pawlak (1992)] and MapReduce programming model [Dean and Ghemawat (2010); Triguero *et al.* (2015)].

2.1. *Rough set theory*

The indiscernibility relation and equivalence class are momentous concepts in Pawlak's rough set model. The indiscernibility relation expresses the fact that due to lack of information (or knowledge), we are unable to discern some objects by employing available information. It determines a partition of U and is used to construct the equivalence classes.

Definition 1. $S =< U, C \cup D, V, f >$ is a decision table, $U = \{x_1, x_2, \cdots, x_n\}$ named domain is a finite non-empty set of objects. $C = \{c_1, c_2, \cdots c_n\}$ is a set of conditional attributes describing the objects, and D is a set of decision attributes that indicates the classes of objects. $C \cup D = \varnothing$. $V = \bigcup_{a \in C \cup D} V_a$, V_a is a non-empty set of values of $a \in C \cup D$. $f : U \times (C \cup D) \to V$ is an information function that maps an object in U to exactly one value in V_a , that means , $f(x, a) = v$ means that the object x has the value v on attribute a.

Definition 2. An indiscernibility relation with respect to $R \subseteq C \cup D$ is defined as:

$$IND(R) = \{(x, y) \in U \times U \mid \forall a \in R, f(x, a) = f(y, a)\}. \tag{1}$$

The partition generated by $IND(R)$ is denoted as $U / IND(R)$, U / R for short. Any elements in the U / R , $[x]R = \{y \mid \forall a \in R, f(x, a) = f(y, a)\}$ is the equivalence classes.

Definition 3. For a decision table $S =< U, C \cup D, V, f >$, for each subset of $X \subseteq U$ and indiscernibility relation $R \subseteq C \cup D$, the lower and upper approximations of X with respect to a partition R are defined as:

$$R(X)_{-} = \cup \{Y \in U / R : Y \subseteq X\}. \tag{2}$$

$$R(X)^{-} = \bigcup \{Y \in U / R : Y \cap X = \varnothing\}. \tag{3}$$

Definition 4. For a decision table $S = < U, C \cup D, V, f >$,Let $\forall A \in C$, , then positive region $POS(D \mid A)$ is given by :

$$POS(D \mid A) = \underset{1 \le i \le k}{R(X)}_{-}. \tag{4}$$

Definition 5. Let $U / A = \{A_1, A_2, \cdots, A_r\}$, $U / D = \{D_1, D_2, \cdots, D_k\}$, then information entropy of A is given by :

$$H(A) = -\sum_{i=1}^{r} p(A_i) \log_2 p(A_i). \tag{5}$$

Conditional entropy of D conditioned on A is given by:

$$H(D \mid A) = -\sum_{i=1}^{r} p(A_i) \sum_{j=1}^{k} p(D_j \mid A_i) \log_2 p(D_j \mid A_i). \tag{6}$$

Where $p(D_j \mid A_i) = \frac{\left| D_j \cap A_i \right|}{\left| A_i \right|} (i = 1, 2, \cdots, r; j = 1, 2, \cdots, k)$.

2.2. *MapReduce*

MapReduce is a paradigm of parallel programming [Dean and Ghemawat (2010)] and an associated implementation [Dean and Ghemawat (2004)] designed to process or generate large data sets. It allows us to tackle big data sets over a computer cluster regardless the underlying hardware or software. It is characterized by its highly transparency for programmers, which allows to parallelize applications in a easy and comfortable way. MapReduce has been successfully applied in data mining [Han *et al.* (2011); Verma *et al.* (2010)], machine learning [Srinivasan *et al.* (2012); Chu *et al.* (2007); Zhao *et al.* (2009)] and web indexing [Zinn *et al.* (2010)]. In addition, Mapreduce paradigm is already adapted to demonstrate the parallel speed up technique on a variety of learning algorithms including locally weighted linear regression (LWLR), k-means, logistic regres-sion (LR), naive Bayes (NB), SVM, ICA, PCA, gaussian discriminant analysis (GDA), EM, and backpropagation (NN) [Chu *et al.* (2007)].

Based on functional programming, this model works in two different steps: the map phase and the reduce phase. Each one has key/value (<k, v>) pairs as input and output. Both phases are written by the user. The map phase takes each key/value pair and generates a set of intermediate key/value pairs. Then, The MapReduce library merges all intermediate values associated with the same intermediate key I and passes them to the reduce function. The reduce phase takes an intermediate key I and a set of values for that key as input for further produce. It groups these values together to form a valid smaller set of values. Typically just zero or one output value is produced per reduce invocation. The intermediate values are supplied to the user's reduce function via an iterator. MapReduce makes it possible to handle lists of values that are too large to fit in memory. Fig. 1 depicts a flowchart of the MapReduce framework. In a MapReduce program, all

map and reduce operations run in parallel. First, all map functions run independently. Meanwhile, reduce operations will wait until their respective maps are completed. Then, they process different keys concurrently and independently. Note that inputs and outputs of a MapReduce job are stored in an associated distributed file system that is accessible from any computer of the used cluster.

In this paper we will focus on the Hadoop implementation because of its performance, open source nature, installation facilities and its distributed file system (Hadoop Distributed File System, HDFS).

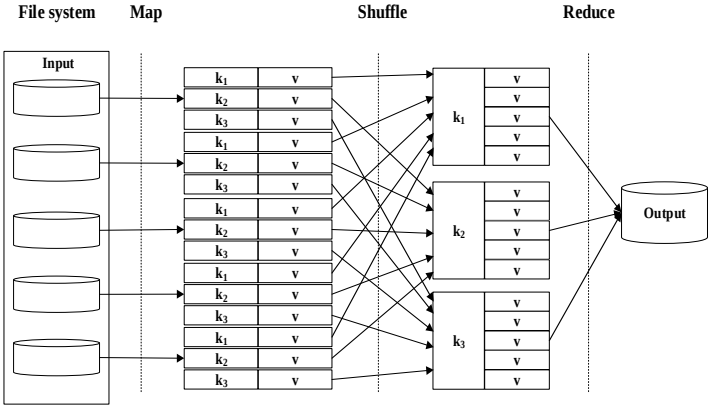

Fig. 1. Flowchart of the MapReduce framework [Triguero *et al.* (2015)].

3. Knowledge Reduction Method for Big Data using Mapreduce

In this section, we will describe the parallel method (the combination of rough set, information entropy, MapReduce) and rewritten the map and reduce function.

3.1. *The method description*

Majority of the traditional methods based on rough sets, for example, serial algorithms and existing rough set tools, only execute on single machine. It can not deal with massive data sets. This drawback is not conductive to the wide application of rough set. As a reaction to this disadvantage, we designed a new parallel method to handle large big sets based on rough set. Knowledge reduction algorithm based on rough set is redesigned to run on Hadoop platform, and we combine the rough set algorithm with MapReduce. The new method provides a feasible way for parallel reduction. To redesign algorithms (i.e., attribute reduction, rule induction algorithms and so on) for dealing with huge data effectively, we propose a parallel method for computing rough set equivalence classes and decision classes. The association is defined to describe the relation between equivalence classes and decision classes. We present a parallel method for computing rough set approximations. Finally, the parallel algorithms are implemented on Hadoop MapReduce platform.

From the definition 4 and definition 5, the calculation form based positive region or information entropy both can be expressed as $\sum_{1\le i\le l}\Delta$ (Δ represents some sort of calculation of the same equivalence class), <equivalence class, some sort of calculation of the equivalence class> is similar with <k, v>, so equivalence classes can be computed in parallel using MapReduce [Qian *et al.* (2013)]. The emphasis of this paper is the calculation form based on information entropy, so we give more details of knowledge reduction algorithm based on information entropy in part B in this section. The calculation form based positive region is regarded as experimental control group, and more details can be found in [Qian *et al.* (2013)].

3.2. *Knowledge reduction algorithm using MapReduce*

In Parallel knowledge reduction algorithm, Map function is written to complete the calculation of equivalence classes of different data blocks, Reduce function is written to calculate the same equivalence class information entropy. As indicated in [Qian *et al.* (2013)], the final summary of equivalence classes which is calculated in each data slice using MapReduce technology is the same with equivalence classes which is calculated in the entire S .

Here is the definition of information entropy attribute importance.

Definition 6. For a decision table S , let $A\subseteq C$, $\forall c\in C-A$, the information entropy attribute importance of c is given by :

$$Sig_{\inf o}(c,A,D)=H(D\mid A)-H(D\mid A\cup c). \tag{7}$$

The positive region attribute importance of c is given by :

$$Sig_{pos}(c,A,D)=\mid POS(D\mid A\quad c\mid-\mid POS(D\mid A)\mid. \tag{8}$$

Knowledge reduction algorithm based on MapReduce is mainly including 3 functions: the Map function (algorithm 1), the Reduce function (algorithm 2) and main function (algorithm 3).

Algorithm 1 Map (k, v)

Input: the selected attributes set A , the candidate attribute $c\in C-A$, a data split S_i .

Output: $Ac\_EquivalenceClass,<d(x),1>$.

1. For $x\in S_i$ do

2. For $c\in C-A$ do

Emit $Ac\_EquivalenceClass,<d(x),1>$

By algorithm 1, we can compute the equivalence classes of different data blocks.

Algorithm 2 Reduce (String $Ac\_EquivalenceClass$, pairs $[<d_1,n_1>,<d_2,n_2>,\cdots]$)

Input: equivalence classes and its corresponding decision value list.

Output:

// Sig_{Δ}^{c} is the importance of attribute c in the equivalence class. $<d,n>\in[<d_1,n_1>,<d_2,n_2>,\cdots]$

1. For do

Count the number of different decision values $(n_p^1, n_p^2, \cdots, n_p^k)$;

2. Calculate $Sig_{\inf o}(c, A, D)$;

3. Output $< c\_EquivalenceClass, Sig_{\Delta}^{c} >$.

By algorithm 2, we can compute the importance degree of attribute c by $Sig_{\inf o}(c, A, D)$.

Algorithm 3 Main function

Input: a decision table S .

Output: a reduction Red .

1. $Red = \varnothing$;

2. Calculate $H(D \mid C)$;

3. Start a job,

while $H(D \mid Red) \neq H(D \mid C)$

{execute algorithm 1 and algorithm 2, according to the result $Sig_{info}(c \in C - Red)$, select the $c_l = \{c \mid \underset{c \in C - Red}{Best}(Sig_{\Delta}^{c})\}$, (if this C is not unique, then choose one), $Red = Red \bigcup \{c_1\}$ }

4. Start a job. For each attribute c, begin from the tail to the head of Red , execute algorithm 1 and algorithm 2.

If ($H(D \mid Red) = H(D \mid Red - c)$)

then $Red = Red - \{c\}$;

5. Output Red .

By Algorithm 3, according to the importance degree of the individual candidate attributes, we can determine an optimal candidate attributes.

4. Experimental Evaluation

In this section, We intend to investigate the effectiveness of using MapReduce for big data parallel knowledge reduction, as embodied by computing attribute importance and performing parallel search. Section 4.1 describes the data sets used to evaluate the method. Section 4.2 shows the details of hardware and software used in this experiments. Section 4.3 presents and discusses the experimental results of two different algorithms achieved.

4.1. *Data sets*

Port as an important node in the intersection of land and water logistics, is an important foundation and hub of the national economy. It produces large amounts of data in the daily production, but the method how to analysis this collection of large datasets and how to extract knowledge precisely from it are not available. In addition, we have been studying on the analysis of port data for several years. Port data is available to us and the experiment is very meaningful. So port big data was selected for this experiment. Then, we regard a port big data set as a port knowledge representation system and analysis the specific condition attributes of decision attribute. The condition attribute is influence

factors of yard utilization and the decision attribute is yard utilization. This experiment can find out the significant influence attributes which affect yard scheduling more.

This paper choose the port production data to test this method. The data stored in a production business system belongs to a Chinese port. The port data decision table contains 40 attributes and 1 decision attribute. The condition attributes are influence factors of yard utilization. The decision attribute is yard utilization. The purpose is to remove the irrelevant attributes and confirm the attributes more important. Experimental results will contribute to yard scheduling and enterprise decision. In order to test better, we copy the original data set to make two data sets that respectively contain 5 million and 10 million. Table 1 shows a decision table of port yard utilization.

Table 1. A decision table of port yard utilization.

No.	**Condition Attributes**				**Decision Attribute**
	Yard type	**Customer type**	**Cargo**	**…**	**Yard utilization**
1	1	2	1	…	0.65
2	0	1	4	…	0.78
3	2	2	3	…	0.72
⋮	⋮	⋮	⋮	⋮	⋮

- Yard type: 1-outside storage, 1-silo, 2-storehouse, ect.
- Customer type: 0-VIP， 1-important customers, 2-ordinary customers, 3-small customers
- Cargo: 0-coal, 1-steel, 2-ore, 3-grain, 4-vehicle, ect.
- Transportation tools: truck, train, barge, ect.
- Handling Technology: bucket wheel, crane bridge, belt conveyor, ect.
- Season: spring, summer, autumn, winter
- …
- Yard utilization: we can analysis the service condition of yard.

Though there are many factors that affect the yard utilization, we only select certain available factors with data generated from the Enterprise information system. Different characteristic attributes have different dimensions. The unit and the order of magnitude are usually different. Considering the impact of the difference of dimension and magnitude on the results of the model evaluation, data normalization method should be used to convert the different characteristic values into dimensionless values. In this paper, we quantify the attributes first, and then make the data dimensionless. Table 1 is the result.

4.2. *Hardware and software used*

The experiments have been carried out on eleven nodes in a cluster: The master node and ten compute nodes. Each one of these computer nodes has the following features:

- Processors: Intel Core i7 4702MQ
- Cores: 4 per processor (8 threads)
- Clock speed: 2.20 GHz
- Cache: 6 MB
- Network: Gigabit Ethernet
- Hard drive: 500GB
- RAM: 4 GB

The specific details of the software used are the following:

- MapReduce implementation: Hadoop 2.6.0. MapReduce 1 runtime (Classic). Cloudera's open-source Apache Hadoop distribution.
- Maximum maps tasks: 38.
- Maximum reducer tasks: 1.
- Operating system: Cent OS 6.4.
- Java SE Development Kit: JDK1.7

Note that the total number of cores of the cluster is 40. However, the maximum number of map tasks is limited to 38 and one for the reducers.

4.3. *Experimental analysis*

Run time, speedup, scale-up are used to evaluate the performance of the $< c\_EquivalenceClass, Sig_{\Delta}^{c} >$ knowledge reduction algorithm. The comparisons between the knowledge reduction algorithm based on information entropy and the knowledge reduction algorithm based on positive region are also given below.

4.3.1. *Run time*

In this part, we run the knowledge reduction algorithm based on information entropy on 1～10 nodes.

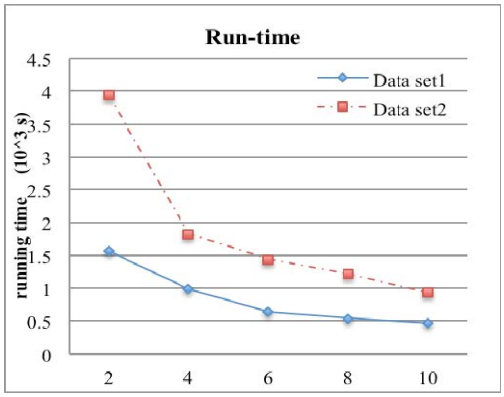

Fig. 2. Running time (information entropy).

Fig. 2 shows the running time. X axis represents the number of nodes, and Y axis represents run-time. With the increase of the number of nodes, the knowledge reduction algorithm based on information entropy process more efficiently and the running time of

same data set declines quickly. And, as the size of the data set increases, reduction performs better.

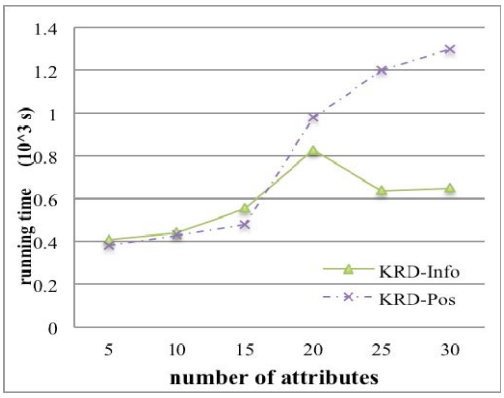

Fig. 3. Run time of two algorithms.

Fig. 3 shows the run-time of two different algorithms. Only data set2 is used here and the number of nodes is 10. X axis represents the number of attributes, and Y axis represents running time. KRD-Info represents the running time of knowledge reduction algorithm based on information entropy. KRD-Pos represents the running time of knowledge reduction algorithm based on positive region. Dealing with the high dimensional data set, the running time of the two different algorithms is obviously different. With the increase of the number of attributes, the knowledge reduction algorithm based on information entropy performs better than the knowledge reduction algorithm based on positive region.

4.3.2. *Speedup*

To measure the speedup, we keep the data set constant and increase the number of nodes in the system. Speedup given by the larger system is defined by the following formula [Xu *et al.* (1999)]:

$$Speedup(n) = \frac{T_1}{T_n}. \tag{9}$$

where n is the number of nodes, T_1 is the execution time on one node, T_n is the execution time on n nodes.

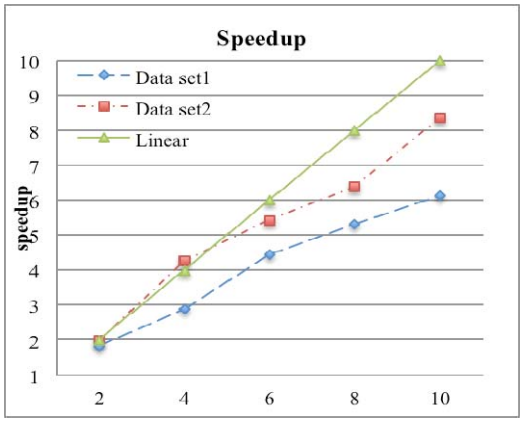

Fig. 4. Speedup(information entropy).

Fig. 4 shows the speedup of knowledge reduction algorithm based on information entropy. As the results show, the algorithms have a very good speedup performance. With the size of the data set increases, the speedup performs better. However, linear speedup is difficult to achieve because the communication cost increases with the number of clusters becomes large.

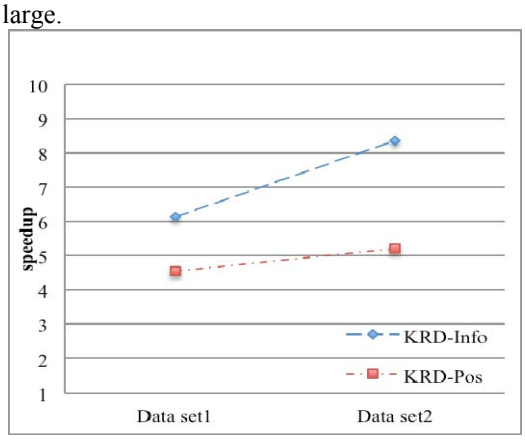

Fig. 5. The speedups of two algorithms.

Fig. 5 shows the speedups of two algorithms. The number of nodes is 10. X axis represents data sets, and Y axis represents speedup. KRD-Info represents the speedup of knowledge reduction algorithm based on information entropy. KRD-Pos represents the speedup of knowledge reduction algorithm based on positive region. Under the same conditions, the speedup of knowledge reduction algorithm based on information entropy is higher than the speedup of knowledge reduction algorithm based on positive region.

4.3.3. *Scaleup*

Scaleup is defined as the ability of a n-times larger cluster to perform a n-times larger job in the same run time as the original system. We make the data set and computer node increases in proportion, and then record the run time.

$$Scaleup(D,n)=\frac{T_{D_1}}{T_{D_n}}. \tag{10}$$

where D is the data set, T_{D_1} is the execution time for D on one node, T_{D_n} is the execution time for $n\times D$ on n nodes.

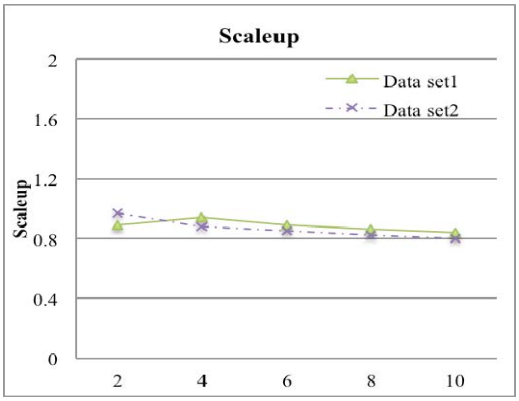

Fig. 6. Scaleup.

To test the sacleup, we run the knowledge reduction algorithm based on information entropy on 1～10 nodes.

Fig. 6 shows the scaleup of knowledge reduction algorithm based on information entropy. X axis represents the number of nodes, and Y axis represents scaleup. Obviously, the algorithm has good scalability. Comparing the performances of data set1 and data set2, we can conclude that the scaleup performs better with the size of the data set increases.

From the above experimental analysis, we can conclued the following results.

(1) Knowledge reduction method based on information entropy using MapReduce can efficiently process massive datasets. As the size of the data set increases, the method performs better.

(2) Comparing the results obtained with the two algorithms, we can observe that the more efficient test results is obtained with Knowledge reduction method based on information entropy. The algorithms corresponding to the parallel method based on MapReduce were successfully designed.

(3) Port big data can be processed into a decision table. This method can identify decisive factors in reducing the size of the data without deviation.

5. Conclusions

In this paper, we proposed knowledge reduction method using Mapreduce that can handle big data. And the MapReduce model is an efficient computational model for distributed parallel processing with big data. The knowledge reduction algorithm based on information entropy is successfully designed. The knowledge reduction algorithm based on information entropy is applied in the control experiment. The experimental results demonstrate that the knowledge reduction algorithms using MapReduce can scale well and efficiently process big data on Hadoop. The knowledge reduction algorithm based on information entropy performs better than the knowledge reduction algorithm based on positive region. Our future research work will focus on applications of the proposed parallel method in knowledge reduction based on rough sets.

Acknowledgments

This research was supported by Natural Science Foundation of China under Grant Nos. 71132008.

References

Błaszczyński, J.; Słowiński, R.; Szeląg, M. (2011): Sequential covering rule induction algorithm for variable consistency rough set approaches. Information Sciences, **181**(5), pp. 987-1002.

Cheng, C. H.; Chen, T. L.; Wei, L. Y. (2010): A hybrid model based on rough sets theory and genetic algorithms for stock price forecasting. Information Sciences, **180**(9), pp. 1610-1629.

Chu, C. T.; Sang, K. K.; Lin, Y. A.; Yu, Y. Y.; Bradski, G. R.; Ng, A. Y.; et al. (2007): MapReduce for machine learning on multicore. Advances in Neural Information Processing System, **19**, pp. 281-288.

Chen, H.; Li, T; Qiao, S.; Ruan, D. (2000): A rough set based dynamic maintenance approach for approximations in coarsening and refining attribute values. International Journal of Intelligent Systems, **25**, pp. 1005-1026.

Deng, D.; Yan, D.; Wang, J. (2010): Parallel reducts based on attribute significance. Lecture Notes in Computer Science, **6401**, pp. 336-343.

Dean, J.; Ghemawat, S. (2010): MapReduce: A Flexible Data Processing Tool. Communications of the ACM, **53**(1), pp. 72-77.

Dean, J.; Ghemawat, S. (2004): MapReduce: Simplified Data Processing on Large Clusters. Communications of the Acm, **51**(1), pp.107-113.

Guyon, I.; Elisseeff, A. (2003): An Introduction to Variable and Feature Selection. Journal of Machine Learning Research, **3**, pp. 1157-1182.

Ghemawat, S.; Gobioff, H.; Leung, S. (2003): File and storage systems: the google file system. Acm Symposium on Operating Systems Principles Bolton Landing, **37**, pp. 29-43.

Han, L.; Liew, C. S.; Hemert, J. V.; Atkinson, M. (2011): A generic parallel processing model for facilitating data mining and integration. Parallel Computing, **37**, pp. 157-171.

Hu, Q.; Liu, J.; Yu, D. (2008a): Mixed feature selection based on granulation and approximation. Knowledge-Based Systems, **21**(4), pp. 294-304.

Hu, Q.; Xie, Z.; Yu, D. (2007): Hybrid attribute reduction based on a novel fuzzy-rough model and information granulation. Pattern Recognition, **40**(12), pp. 3509-3521.

Hu, Q.; Yu, D.; Liu, J.; Wu, C. (2008b): Neighborhood rough set based heterogeneous feature subset selection. Information Sciences, **178**(18), pp. 3577-3594.

Liou, J. J. H.; Tzeng, G. H. (2010): A Dominance-based Rough Set Approach to customer behavior in the airline market. Information Sciences, **180**(11), pp. 2230-2238.

Li, T.; Da, R.; Geert, W.; Song, J.; Xu, Y. (2007): A rough sets based characteristic relation approach for dynamic attribute generalization in data mining. Knowledge-Based Systems, **20**(5), pp. 485-494.

López, V.; Río, S. D.; Benítez, J. M.; Herrera, F. (2015): Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data. Fuzzy Sets and Systems, **258**, pp. 5-38.

Li, Y.; Zhu, S.; Wang, X. S.; Jajodia, S. (2006): Looking into the seeds of time: Discovering temporal patterns in large transaction sets. Information Sciences, **176**(8), pp. 1003-1031.

Pyle, D. (1999). Data preparation for data mining. Academic Press.

Pawlak, Z. (1992). Rough Sets: Theoretical Aspects of Reasoning About Data. Kluwer Academic Publishers.

Pawlak, Z.; Skowron, A. (2007a): Rough sets: Some extensions. Information Sciences, **177**(1), pp. 28-40.

Pawlak, Z.; Skowron, A. (2007b): Rudiments of rough sets. Information Sciences, **177**(1), pp. 3-27.

Pawlak, Z. (1982): Rough sets. International Journal of Computer & Information Sciences, **11**, pp. 341–356.

Qian, J.; Lv, P., Yue, X.; Liu, C.; Jing, Z. (2015): Hierarchical attribute reduction algorithms for big data using MapReduce. Knowledge-Based Systems, **73**, pp. 18-31.

Qian, J.; Miao, D.; Zhang, Z.; Zhang, Z. (2013): Parallel algorithm model for knowledge reduction using mapreduce. Journal of Frontiers of Computer Science & Technology, **7**(1), pp. 35-45.

Qian, J.; Miao, D. Q.; Zhang, Z. H.; Li, W. (2011): Hybrid approaches to attribute reduction based on indiscernibility and discernibility relation. International Journal of Approximate Reasoning, **52**(2), pp. 212-230.

Qian, Y.; Liang, J.; Dang, C. (2010): Incomplete multigranulation rough set. Systems Man & Cybernetics Part A Systems & Humans IEEE Transactions on. **40**(2), pp. 420-431.

Srinivasan, A.; Faruquie, T. A.; Joshi, S. (2012): Data and task parallelism in ILP using MapReduce. Machine Learning, **86**(1), pp. 141-168.

Triguero, I.; Peralta, D.; Bacardit, J.; García, S.; Herrera, F. (2015): MRPR: A MapReduce solution for prototype reduction in big data classification. Neurocomputing, **150**(150), pp. 331-345.

Verma, A.; Llorà, X.; Goldberg, D. E.; Campbell, R. H. (2010): Scaling genetic algorithms using MapReduce. Ninth International Conference on Intelligent Systems Design and Applications, **16**, pp. 13-18.

Xu, X.; Jäger, J.; Kriegel, H. P. (1999): A fast parallel clustering algorithm for large spatial databases. Data Mining and Knowledge Discovery, **3**(3), pp. 263-290.

Zinn, D.; Bowers, S.; Köhler, S.; Ludäscher, B. (2010): Parallelizing XML data-streaming workflows via MapReduce. Journal of Computer and System Sciences, **76**(6), pp. 447-463.

Zhao, W.; Ma, H.; He, Q. (2009): Parallel K-Means clustering based on MapReduce. in Proc. 1st International Conference on Cloud Computing, **5931**, pp. 674-679.