Journal of Information Hiding and Multimedia Signal Processing ©2018 ISSN 2073-4212
Ubiquitous International Volume **9**, Number **2**, March **2018**

An Approximately Duplicate Records Detection Method for Electric Power Big Data Based on Spark and IPOP-Simhash

Ren-Jie Song[1], Tong Yu[2]*, Yu-Hong Chen[3], Yu-Yang Chen[3], Bin Xia[3]

[1,2] Department of Information Engineering, Northeast Electric Power University, Jilin 132012, China
[3] State Grid Jilin Power Supply Company, Jilin 132000, China
*Corresponding Author: 862888620@qq.com
[1]yutongnedu@sina.com

Received January, 2017; revised November, 2017

Abstract. *Aiming at the problems that the single computing resource is unable to fulfill the requests of approximately duplicate records detecting for massive and high-dimensional electric power grid data, and those algorithms based on MapReduce require numerous IO overheads, hence a new method based on Spark platform and IPOP-Simhash (improved and optimized Simhash algorithm) is proposed for the approximately duplicate records detecting of electric power big data. Firstly, a FGA (Fingerprint Generation Algorithm) is designed so as to improve the traditional Simhash algorithm, and on the basis of Data-Deficiency and Bi-Gram processing, a weight calculation method based on the combination of subjective and objective analysis is used to optimize the traditional Simhash algorithm, proceeding to improve the accuracy of the approximately duplicate records detection. Secondly, a FCA (Fingerprint Classification Algorithm) based on a graph clustering analysis and a FIT (Fingerprint Index Tree) are designed, in addition, a fingerprint search strategy based on the FIT and FCA is proposed so as to improve the accuracy and efficiency of similar records detectiing. Finally, a new algorithm (SP-MATCH) based on the IPOP-Simhash and Spark platform is proposed, it is applied to the approximately duplicate records detection for massive electric power grid data, moreover, a comparatively experimental analysis about the electric power grid data from UCI is made on a cloud computing cluster, the experimental results show that the presented method is applied to the approximately duplicate records detectiing for electric power big data, and it is more efficient and more accurate.*
Keywords: Spark, electric power big data, Approximately duplicate records detection, Graph clustering analysis, cloud computing

1. **Introduction.** Approximately duplicated records detection for electric power data is the key point of guaranteeing the quality of power grid data, more importantly, it is an important guarantee to obtain a reliable and effective decision-making knowledge of a electric power system control, and it is of great significance to improve the safe and stable operation of electric power grid and promote the healthy development of the national economy. However, with the depth development of the smart grid, the electric power data is increasing exponentially, hence the approximately duplicated records detection for massive data have widely become a hot research [1, 2, 3].

For a long time, the study of approximately duplicated records detection has achieved substantial results. For example, the approximately duplicated problems are converted into a special clustering problem, so that the approximative records are detected [4]. A

data cleaning method based on time series analysis is proposed, which is used to detect the noise data of device status information [5]. Aiming at the Similarity of graph structure data, a definition and a matching algorithm about the graph pattern are proposed [6]. In addition, in the literatures [6, 7], some relevant ideas of entity analysis are proposed, such as, a pattern-based entity resolution algorithm is proposed [7], which generates a record pattern by merging similar records, and then judges the identity of the entity by comparing the record patterns, while for the accuracy of entity analysis, the information gain tactic and the probability statistics method are used to calculate the weight of the record attribute to measure its importance to improve the accuracy of the entity analysis [8]. A divide-and-conquer algorithm for big datasets is proposed [9], and the literatures [10, 11] in machine learning and other aspects achieved good results. However, these methods above have a good performance in processing small datasets, but when faced with the high-dimensional and large-scale electric power datasets, the computational resources of single-machine can't meet the datasets processing requirements, and there will be a serious imbalance between accuracy and recall rate, it is easy to fall into local optimum and lack of efficiency, etc. so that the performance of the approximately duplicated records detection is significantly decreased. On the other hand, although the algorithms based on MapReduce can process massive datasets, the data access has high dependency on local disk and HDFS in the data calculation process, and when the algorithms face with a large number of iterative operations, the overall efficiency of it is low and the disk access is inefficient, hence the detection performance of approximately duplicate records is seriously affected, and with the intelligent development in electric power system, the electric power data is exponentially increasing, the annual growth in data volumes has grown from GB to TB [12], and the rapid growing of the data volume is also expanding the data dimension, more than 70% of the data is redundant [13], these make the approximately duplicate records detection more difficult, and it is unable to effectively complete the approximately duplicate records detection for the massive and high-dimensional electric power data.

Therefore, in this paper, the traditional Simhash algorithm [14] is optimized and improved (IPOP-Simhash). and in view of the Spark's advantages that the rationality of computing resource allocation and the superiority of data processing, a approximately duplicate records detection algorithm (SP-MATCH) for massive power grid data based on the Spark and the IPOP-Simhash is proposed. which can effectively realize the efficient and accurate detection of the near-duplicate records for distributed electric power big data, and solve the bottlenecks caused by the high-dimensional and massive electric power grid data. Experimental results show the good performance of the presented algorithm.

This paper gives a beginning work in this area. Then the traditional Simhash algorithm and its Shortcomings are described in Section 2. Section 3 gives the improvement and optimization of the traditional Simhash algorithm, namely the IPOP-Simhash is proposed. In Section 4, the SP-MATCH is proposed to realize the near-duplicate records detection of distributed electric power big data. In Section 5, a case analysis is discussed. Conclusion remarks are finally offered in section 6.

2. **Traditional Simhash Algorithm.**

2.1. **The Description of Traditional Simhash Algorithm.**

Traditional Simhash algorithm is mainly used for removing redundant document by converting the documents into low-dimensional fingerprint (binary string), the more similar of documents, the closer

fingerprints of them. It can detect different contents in the bit on the degree of difference, the steps are as follows:

(1) The keywords of a document to be detected are extracted.

(2) A ψ-dimensional vector v is initialized to 0 and the ψ-dimensional Simhash fingerprint s is initialized to 0.

(3) A standard hash algorithm is used to calculate ψ-bit hash value $b_{i,n}$ of each keyword, then look at each bit of $b_{i,n}$, if the i-bit is 1, then the i-bit in v also is incremented by1, otherwise, the i-bit in v is subtracted 1.

(4) After every keyword of the document to be detected are calculated, if the value of the i-bit in v is positive, then the i-bit in s is 1, otherwise, the i-bit in s is 0, finally, the s is output, which is the fingerprint of the document to be detected.

(5) The Jaccard distances among fingerprints are compared to determine their similarity, for fingerprint A and fingerprint B, $Jaccard(A, B) = \left|\frac{A \cap B}{A \cup B}\right|$.

2.2. **The Shortcomings of Traditional Simhash Algorithm.** The traditional Simhash algorithm has good effects on the detection of near-duplicate documents containing numerous keywords, the documents to be detected are converted into corresponding fingerprints by the algorithm, and the similarity among documents are measured by comparing their fingerprints. However, when the amount of keywords in a document is few, the generation accuracy of their fingerprints is also extremely low. Therefore, the traditional Simhash is not applied to convert a record (tuple) into corresponding fingerprint. In addition, the newly entered data is not preprocessed, such as filter out some irrelevant vocabularies and process the missing values before keywords are extracted. On the other hand, the similarity among fingerprints are measured by the method of calculating Jaccard distance, that every bit of the fingerprints is traversed and the position of every bit in the fingerprints is not guaranteed, thus the near-duplicate fingerprint detecting is inefficient and the accuracy is also low. Therefore, in this paper, we propose a new IPOP-Simhash algorithm based on the improvement and optimization of the traditional Simhash, and a new algorithm SP-MATCH based on the IPOP-Simhash and Spark computing platform is proposed proceeding to cope with the trend of power big data, which can effectively solve the bottlenecks caused by high-dimensional and massive power grid data, improve the performances of approximately duplicate records detection of distributed electric power big data.

3. **The Improved and Optimized Simhash algorithm (IPOP-Simhash).** In view of the shortcomings and deficiencies of the traditional Simhash algorithm, its improvement and optimization schemes are as follows: we design a FGA (fingerprint generation algorithm) as to convert the text data belonging to the same tuple to its corresponding feature fingerprint (a binary string), proceeding to improve the traditional Simhash algorithm, and on the basis of Data-Deficiency and Bi-Gram processing, we use a weight calculation method based on the combination of subjective and objective analysis to improve the accuracy of the approximately duplicate records detection. In addition, a FCA (Fingerprint Classification Algorithm) based on graph clustering analysis and a FIT (Fingerprint Index Tree) are designed, proceeding to optimize the traditional Simhash algorithm, Simultaneously improve the accuracy and efficiency of the approximately duplicate records detection. Finally, a fingerprint search strategy based on the FIT and the FCA is proposed, which can implement the output of approximately duplicate records.

3.1. **The FGA (Fingerprint Generation Algorithm) Based on Simhash.** Aiming at the defects of the traditional Simhash algorithm that it is only applied to those documents that including numerous keywords, and with low detection accuracy, a new

algorithm for transforming records into fingerprints based on the traditional Simhash is designed, and its steps are as follows:

(1) Enter the records (ID_i) to be detected, and the $n \times f$ arrays are initialized to 0, v and s. The f-dimensional vectors are initialized to 0, $b_{i,j}$ and s_i, where i is the row number, j is the number of keywords, n is the number of records to be detected, f is the number of bit output by a hash function.

(2) The records ID_i are processed based on its fields by Bi-gram method to divide keywords $t_{(i,j)}$, so that the keyword sets $B_i = \{t_{(i,j)}\}$ of the records ID_i are formed. For example: the keyword "gram" is processed, the elements in B_i are "gr, ra, am", If there is only a character, the character is the element of B_i alone. The features of records ID_i are characterized by elements in B_i.

(3) Each element (keyword) $t_{i,j}$ in B_i is processed by a same hash function, the hash values $h_{i,j}$ of each keyword $t_{i,j}$ are output respectively.

(4) If the weight of $t_{i,j}$ is $W_{i,j}$, then $b_{i,j} = W_{i,j} \times h_{i,j}$, and all the $b_{i,j}$ are summed based on their corresponding bit, if all the keywords of records ID_i are added over, then the results (v_i) of the sum are output to the i row of the array v.

(5) If the r bit in v_i is positive, then the corresponding bit in s_i is 1, otherwise is 0.

(6) Output s_i, namely the s_i is the fingerprint of record ID_i, array $s = (s_i)^{\mathrm{T}}$, and all the records to be detected are transformed into fingerprints and stored in s.

3.2. **A Data-Deficiency Method Based on Neville Algorithm.** In order to improve the precision of near-duplicate records detection, the Neville algorithm [15] with the characteristics of precision controlling and less computation is used to impute the missing data. which is a linear interpolation method based on the Lagrange interpolation polynomial.

Let $G = \{g_1, g_2, L, \ldots, g_m\}$ is a sample dataset of electric power grid, g is a interpolation point, and m is the number of interpolation nodes, and $p = q = 1$, $p \in [1, m]$, $q \in [1, p]$, the Lagrange polynomial interpolation is as follows:

$$L_{p,q} = \frac{[(g - g_{p,q})L_{p,q-1} - (g - g_p)L_{p-1,q-1}]}{(g_p - g_{p-q})} \tag{1}$$

$$|L_{p,q} - L_{p-1,q-1}| < \sigma \tag{2}$$

The (2) is calculated recursively, if it is true, the $L_{p,q}$ is output, else the (1) and (2) are recursively calculated, where the $g_{p,q}$ is the interpolation points, it belongs to the p column and the q row, and the σ is the precision set in advance.

3.3. **A Weight Calculation Method Based on Subjective and Objective Combination.** In order to process the newly input data to filter out some irrelevant vocabularies and simultaneously improve the detection accuracy of approximately duplicate records, this paper comprehensively considers the weight calculation based on subjective and objective combination. The Delphi method (a subjective method of weight calculation) was used to calculate the part-of-speech weights $w_{1(i,j)}$ of keywords $t_{(i,n)}$, the TF-IDF [16] algorithm (a objective method of weight calculation) is used to calculate the word frequency weight $w_{2(i,j)}$ of the keywords. Finally, the weight $W_{(i,n)}$ of the $t_{(i,n)}$ is defined based on the $w_{1(i,j)}$ and $w_{2(i,j)}$, the related contents are as follows:

TF: The TF is the occurrence frequency of a keyword $t_{(i,n)}$ in a data table, namely the Word Frequency of a keyword, which is defined as follows:

$$tf_{i,j} = \frac{n_{i,j}}{\sum\limits_{k} n_{k,j}} \tag{3}$$

where the $n_{i,j}$ is the occurrence number of a keyword $t_{(i,n)}$ in the data table E_j, the $\sum_k n_{k,j}$ is the sum of the occurrence number of all the keywords in the data table E_j.

IDF: The IDF is used to weigh the importance of a keyword, IDF of the keyword $t_{(i,n)}$ is defined as follows:

$$idf_i = \frac{|D|}{|\{j : t_{(i,n)} \in E_j\}|} \tag{4}$$

where $|\{j : t_{(i,n)} \in E_j\}|$ is the total number of the records containing the keyword $t_{(i,n)}$ in the data table E_j, $|D|$ is the total number of the keywords in the data table E_j.

Definition 3.1. *TF-IDF function, which is used to calculate the word frequency weight of keywords, and the word frequency weight $w_{2(i,j)}$ of the keyword $t_{(i,n)}$ is defined below:*

$$w_{2(i,j)} = IF - IDF(t_{(i,n)}) = tf_{i,j} \times idf_i \tag{5}$$

Definition 3.2. *The weight $W_{(i,n)}$ of the keyword $t_{(i,n)}$ is defined as follows:*

$$W_{(i,j)} = w_{1_{(i,j)}} + w_{2_{(i,j)}} \tag{6}$$

3.4. **The FCA (Fingerprint Classification Algorithm) Based on Graph Clustering.** In order to improve the detection efficiency of near-duplicate records of massive electric power grid data, a classfication algorithm (FCA) for fingerprints based on graph clustering analysis is proposed, the fingerprints with certain similarity are categorized by the algorithm. Firstly, the fingerprints s_i (binary strings) are transformed into different decimal number d_i and mapped to different points (A_i) in a polar coordinates, and $A_i(\rho, \theta) = (d_i, d_i^o)$. Moreover, define the edge (E) $AB = |d_A - d_B|$, and it represents the similarity between the points. Finally, the points are clustered based on their neighbor relations, thereby the corresponding fingerprints with certain similarity are clustered in a collection. Γ^a is a set of points that are neighbors of the point a. The specific steps of the algorithm are as follows:

Input: figure $G = (V, E)$ translated by fingerprints and parameter η.

Output: Graph clusters $P = (G_1, G_2, \ldots, G_n)$, in which $G_i = \{s_i \,|\, s_i$ refer to the fingerprints belonging to same collection $\}$.

Step 1: For each $E = (a, b)$ do
Step 2: if $|\Gamma^a \cup \Gamma^b| \leq \eta|\Gamma^a \cap \Gamma^b|$ then
Step 3: $a' = \text{merge}\,(a, b)$
Step 4: $d_{a'} = \min(d_a, d_b)$
Step 5: update
Step 6: end if
Step 7: if update = 1 then
Step 8: go to the step 2
Step 9: end for
Step 10: output the value of G

Distinctly, the algorithm relies on the connected components of vertices, the vertices a and b are calculated based on the formula $|\Gamma^a \cup \Gamma^b| \leq \eta|\Gamma^a \cap \Gamma^b|$so as to judge whether they can be contracted to a vertex. Finally, the vertices in the graph is output and they are the results of clustering, each vertex outputted represents a set of fingerprints with certain similarity.

3.5. **The Fingerprint Search Strategy Based on Fingerprint Index Tree and FCA.** In order to improve the accuracy and efficiency of approximately duplicate records detection for massive electric power grid data, FIT is designed and a fingerprint search strategy based on the FIT and FCA is proposed.

3.5.1. *The Construction of a FIT.* The identifications (uniqueness) of fingerprints (s_i) are defined, and denoted as $flag = (d_i, ID_i)$, where ID_i is the line number of the s_i, d_i is the decimal number (uniqueness) corresponding to the s_i (binary) . The steps of how to build a FIT are as follows:

(1) Initializes the root node as an empty set.

(2) The f-dimensional fingerprint s_i of the record ID_i is calculated, and which is divided into $\omega = f/r$ segments, each segment is denoted as β_k, $k \in [1, \omega]$, and the β_k is a binary string of r bit, namely $s_i = \beta_1\beta_2 \ldots \beta_\omega$.

(3) Let the β_k as a node of the FIT respectively, and if $\beta_k = \beta_\xi$, they are regarded as a same node in the FIT, where $k, \xi \in [1, \omega]$. When β_k is a leaf node, the identification $flag_i = (d_i, ID_i)$ is inserted therein, and each path in the FIT is a fingerprint.

(4) The steps (2) and (3) are executed cyclically, until the s_i is empty.

The structure of the FIT is shown in Figure 2:

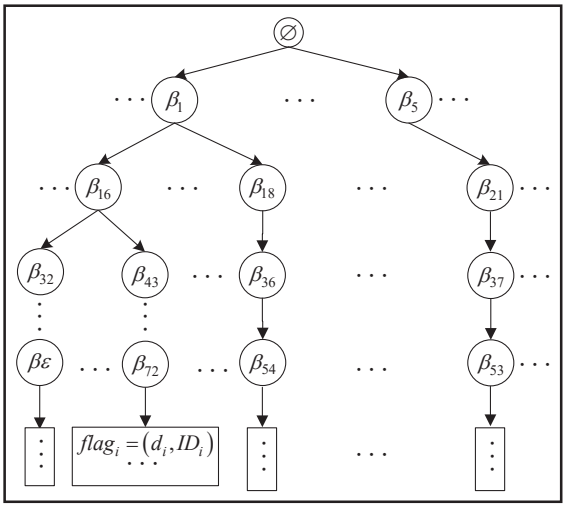

FIGURE 1. The structure of fingerprint index tree

3.5.2. *A Fingerprint Search Strategy Based on the Fingerprint Index Tree and FCA.* In order to compensate the weaknesses of the method of *Jaccard* used in the traditional Simhash algorithm, Hamming distance is used to measure the similarity of fingerprints. As is depicted in the structure of the fingerprint index tree, If we want to detect the similarity of a fingerprint, its leaf node will be traversed, if we want to detect the similarity for all fingerprints in a FIT, the entire FIT will be traversed. In addition, there is no regularity between the paths (namely the fingerprints) in the fingerprint index tree, thus the near-duplicate detection is inefficient. Therefore, the fingerprint search strategy of based on the fingerprint index tree and FCA is designed by the combination of the breadth-first algorithm, the principle of drawer and the Hamming distance. Suppose that the fingerprints that are similar to the fingerprint s_i are searched in the $S = (f_i)^{\mathrm{T}}$, and these fingerprints whose Hamming distance is not greater than μ are similar, every fingerprint has same dimension. The steps of the search strategy based on the fingerprint index tree and FCA are as follows:

(1) FCA (S), namely the set S is processed by the FCA algorithm.

(2) T = FIT(FCA(S)), that is, the fingerprint index tree T is created by using the output of the (1) ;

(3) divide the fingerprint s_i into pieces by the way of processing the fingerprints in the T, namely the s_i is divided into $\omega = f/r$ pieces equally, $s_i = \alpha_1\alpha_2 \ldots \alpha_\omega$.

(4) According to the breadth-first search algorithm, calculate the Hamming distance h_i between the node β_k in the T and the α_i respectively.

(5) According to the principle of drawer, If the $h_i = 0$, and the β_k whose amount is $\omega - \mu$ belong to same fingerprints, then all the fingerprints s_k containing the β_k are similar, and with transitivity among the similar fingerprints.

(6) For all similar fingerprints s_k, the $flag_k$ in leaf nodes are output, and stored in a vector u (a column vector initialized to 0) . however, if a leaf node corresponds to more than one parent nodes, then the $flag_k$ is output according to its d_k.

(7) Output u, that is, the similar fingerprints are output.

4. The Detection of Approximately Duplicate Records for Electric Power Big Data Based on Spark and IPOP-Simhash.

4.1. Spark-A Distributed Computing Framework Based on Memory Calculations.

Spark [17] is a distributed computing framework based on memory Calculations, it's core part is Resilient Distributed Datasets (RDD), which is an abstract concept of a distributed memory, and is also the programming model of Spark.

RDD is a fault-tolerant collection of elements that can be operated in parallel, each RDD is composed of multiple read-only partitions, these partitions run on different work nodes, and how the data are distributed on different work nodes are optimized by RDD by controlling the division of its partitions, and the data to be calculated are stored in memory, thus a lot of I/O overheads are saved, and lots of communication overheads among computers are reduced, which will make the performance of Spark greatly improved. In addition, a RDD mainly has two kinds of operations, namely the Actions and the Transformations, and a result will be produced by the operations of Actions, a new RDD will be defined by the operations of Transformations, moreover, the new RDD relies on its parent RDD, each RDD is linked according to their dependence so as to form a directed acyclic graph (DAG), and the DAG is submitted to the Spark as a Job by performing an Action on it. The execution mechanism of RDD shown in figure 2. where the critical operations are as follows: (1) input: the raw data is read from the HDFS or other file systems and converted to RDD_1; (2) flatMap: according to user's program logic, the RDD_1 is mapped to multiple RDD_{1i}, and the data in RDD_{1i} is stored in the format of $\langle \mathrm{key}, \mathrm{value} \rangle$; (3) Map: the data belonging to RDD_{1i} is mapped according to user's Map function logic, simultaneously the formation of it is transformed into a new key-value pairs $\langle \mathrm{key}, \mathrm{value} \rangle$ and outputted. (4) Reduce: input the data outputted by the Map function, and group the data by key value, then each set of data is processed by according to user's program logic. (5) Join: the RDD generated by Map and the RDD generated by Reduce are connected according to the same key, and simultaneously a new RDD is generated. (6) cache: cache RDD in memory.

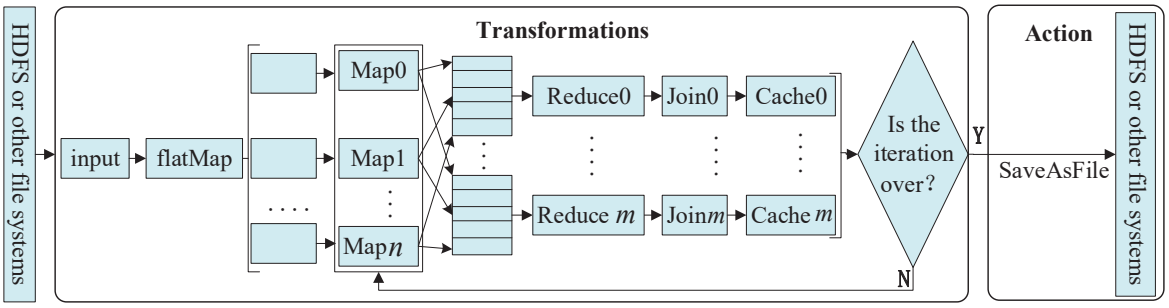

FIGURE 2. Mechanism of RDD operation

In view of the characteristics of RDD, the IPOP-Simhash algorithm and the Spark computing platform are combined in this paper to realize the detection of Approximately Duplicate records of distributed electric power big data.

4.2. **The Realization of IPPO-Simhash Algorithm Based on Spark.** In order to solve the bottlenecks caused by high-dimensional and massive electric power grid data, a algorithm SP-MATCH of approximately duplicate records detection for distributed electric power big data is designed based on the iterative RDD and the IPPO-Simhash, and the algorithm framework is shown in figure 3:

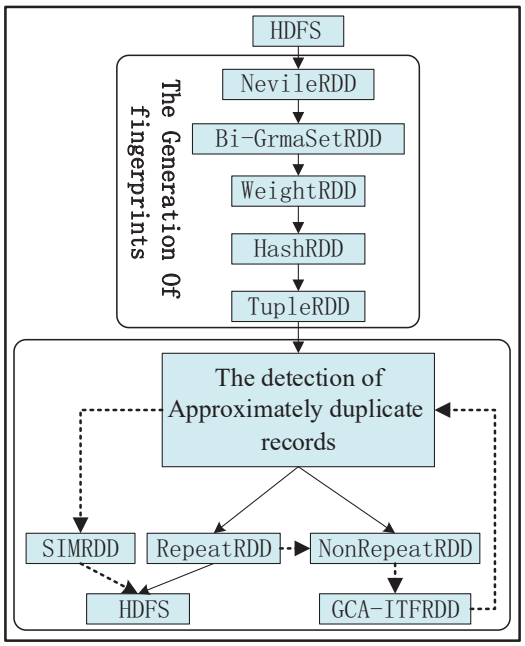

FIGURE 3. The framework of SP-MATCH algorithm

where the NevilleRDD is the RDD generated by the method of imputing missing data based on Neville algorithm, Bi-GramSetRDD is the RDD generated by the processing of records by the method of Bi-Gram, HashRDD is the RDD generated by the processing of keywords by Hash function, WeightRDD is the RDD generated by weight calculation, TupleRDD is the RDD that is generated when the fingerprints of records are generated, RepeatRDD is the RDD that is generated when the duplicate fingerprints are detected, NonRepeatRDD is the RDD that is generated when the duplicate fingerprints are merged and detected, FIT-FCARDD is the RDD that is generated when the approximately fingerprints are detected by the fingerprint search strategy based on the fingerprint index tree and FCA, SIMRDD is the RDD that is generated when the approximately fingerprints are generated. The detection of the approximately duplicate records for distributed electric power big data is divided into two processes, where the solid arrow in the figure 3 is the detection process of duplicate records, the dotted arrow is the detection process of near-duplicate records.

For the table E_k, $K \in \{\text{real numbers}\}$, the row number of it is denoted by ID, the attribute value of the i row and the j column is $A_{i,j}$, $A_{i,j} \in A_i$, To detect the approximately duplicate records in the table E_k, the steps of the algorithm SP-MATCH are described as follows:

Process 1: duplicate records detection;

Input: the table E_k to be detected;

Output: duplicate records;

Step 1: The TextFile function of the SparkContext class is called to read the newly entered data and the Cache function of RDD class is called to load the data into memory.

Step 2: The NevilleRDD is generated by using the Neville algorithm.

Step 3: The Bi-GramSetRDD is generated by using the Bi-gram rules.

Step 4: The WeightRDD is generated by using the method of weight calculation.

Step 5: A filter function is called to filter out some irrelevant vocabularies and simultaneously a new RDD is generated, the data in the RDD is stored in the format of $\langle \text{key} = ID_i, \text{value} = A_i \rangle$.

Step 6: A Hash function is called to process the data outputted by the step 5, and the HashRDD is generated.

Step 7: The TupleRDD is generated by using the IP-Simhash, and the data (fingerprint) of it is stored in $\langle \text{key} = ID_i, \text{value} = s_i \rangle$ format.

Step 8: A Map operation is performed by Executor process so as to the fingerprints are maped to a new key-value pair $\langle \text{key} = ID_i, \text{value} = s_i \rangle$.

Step 9: The results output by the Map are input in format of $\langle \text{key} = ID_i, \text{value} = s_i \rangle$, and a Reduce operation is performed by Executor process.

Step 10: the fingerprints are merged according to the value of key.

Step 11: The RDD generated by step 9 and the RDD generated by step 10 are input, then a Join function is called, and a new RDD is generated.

Step 12: a filter function and other Transformations are called and simultaneously the RepeatRDD and the NonReatRDD are generated.

Step 13: Cache the NonReatRDD in memory

Step 14: Execute Actions, the duplicate fingerprints are output in $\langle \text{key} = s_i / \ldots, \text{values-list} = (ID_1, \ldots / \ldots) \rangle$ format and stored in HDFS or other's file systems by calling the saveAsTextFile function, the fingerprints are clustered together, therefore, the duplicate records are output according to the line number of the its fingerprints.

Process 2: near-duplicate records detection

Input: A threshold μ, and the number of fingerprint segments ω.

Output: near-duplicate records.

Step 1: The RepeatRDD is executed by the filter function to merge the duplicate records and simultaneously generate a NonRepeatRDD, then cache the NonRepeatRDD in memory, and these NonRepeatRDD in memory are processed by the FCA.

Step 2: Input the output of the step 1 to build a Fingerprint Index Tree (FIT), and simultaneously the FIT-FCARDD is generated.

Step 3: The SIMRDD is generated by using the fingerprint search strategy based on fingerprint index tree and FCA algorithm.

Step 4: Execute Actions, and the near-duplicate fingerprints are output and stored in the HDFS or other's file systems by calling the saveAsTextFile function, thus the near-duplicate records are output according to the line number of the it's fingerprints.

The algorithm generally draws lessons from the ideas of "Map" and "Reduce", but all the iterative operations are complete in memory by operating RDD, thus a lot of I/O overheads are saved, which make the performance of Spark greatly improved, and is not the advantage of MapReduce.

5. **Case Analysis.** In order to evaluate the accuracy of the SP-MATCH algorithm and verify its efficiency and parallelism, A Spark cluster platform on 16 Dawning I620-G10 servers is build. and the system configurations are Hadoop 2.6.0, Spark 1.3.1, Redhat 6.4, Scala 2.10.5, JDK 1.8_60, Intel(R), Xeon(R), CPU E5-2620, v2@2.30GHz, 64G memory, 500G hard drive.

Experimental data from the database of University of California-Irvine (UCI) and a electric power company in china. Use the threshold and fingerprint length in the literature [14] as the standard of this paper, namely these fingerprints whose length is 64-bit and with Hamming distance less than or equal to 3 are similar or duplicate, the experimental data were pretreated by the method in the literature [18].

5.1. **The Analysis of Accuracy and Stability.** In order to test accuracy and stability of the algorithm proposed in this paper, the detection results of the SP-MATCH are compared with the traditional SimHash algorithm and the algorithms in the literature [19]. In addition, use the Recall Ratio (R), the Precision Ratio (P) and the F_1-Score (F_1) as the measurements [8], the experimental datasets size are 20GB, and the experimental results are shown in Table 1.

Table 1. Sample data

Attributes	COSY			F TokenSet			F Trigram			F Winkler			Simhash		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
UCI data	0.82	0.81	0.81	0.31	0.97	0.47	0.4	0.98	0.57	0.39	0.90	0.54	0.71	0.80	0.75
Company data	0.78	0.81	0.79	0.41	0.98	0.58	0.4	0.98	0.57	0.39	0.91	0.55	0.72	0.78	0.75
Average	0.80	0.81	0.80	0.36	0.98	0.53	0.4	0.98	0.57	0.39	0.91	0.55	0.71	0.79	0.75

Similarly, the accuracy of the algorithm SP-MATCH is verified with the datasets size of 20GB, In addition, the influence of the datasets size on the accuracy is verified by using datasets size of 20GB, 200GB and 2000GB respectively, and the experimental data includes 12 attributes. The experimental results are shown in Table 2.

Table 2. Detection results of algorithm SP-MATCH

Attributes	Dataset/GB	SP-MATCH		
		P	R	F_1
UCI data	20	0.96	0.98	0.97
Company data	200	0.94	0.96	0.95
Company data	2000	0.88	0.90	0.89
Average	800	0.93	0.95	0.94

It can be seen from the table 2 that the algorithm has high accuracy, and with the rapidly increase of the datasets size, the accuracy is kept at a same level. In addition, when the amount of data is increased from 20GB to 2000GB, the average accuracy of the algorithm is 93%, and the average F_1 is about 94%, namely the algorithm SP-MATCH has high accuracy and stability.

It can be seen from the table 1 and table 2 that the average Recall Ratio of the SP-MATCH is slightly lower than two algorithms, but its average Precision Ratio and average F_1 are highest than all algorithms, that is, the algorithm proposed in this paper has good stability and detection precision, simultaneously with the best performance, and it is most reliable.

5.2. **The Analysis of Efficiency.** In order to verify the superiority of the detection efficiency of the algorithm SP-MATCH, the execution time of the SP-MATCH and the IPOP-Simhash based on MapReduce (MR-IPPO-Simhash) are compared in experiment,

TABLE 3. The Comparison of Time

Algorithms	Dataset/GB	The execution time/s
MR-IPPO-Simhash	2000	1159
SP-MATCH	2000	621

the experimental datasets size is 2000GB, and the experimental results are shown in Table 3:

It can be seen from the table that the execution efficiency of the algorithm SP-MATCH is about 86.7% higher than the MR-IPPO-Simhash under the same experimental condition. Due to Spark is based on distributed memory calculation, thus the algorithm based on it with a memory-to-memory data flow pattern, and only when a RDD is generated, the operations of data reading and results output involve into disk I/O costs, besides the Job processing mode based on DAG is more efficient than that of MapReduce. Therefore the algorithm proposed in this paper has higher efficiency.

5.3. **The Sensitivity Analysis of Parameters in Algorithm SP-MATCH.** Analyze the influence of the datasets size and the number of computing nodes.

5.3.1. *The Relationship Between Datasets Size and Algorithm Efficiency.* Test the influence between datasets size and the efficiency of the algorithm in this paper, the experimental data of 20GB, 40GB, 60GB, 80GB, 100GB, 200GB, 400GB, 800GB, 1600GB, 2000GB are used respectively, and the experimental results are shown in Figure 4. It can be seen from the Figure that the execution time of the algorithm is positively correlated with the data size, and with high efficiency. In a certain range, with the increase of the data size, the parallel performance of the algorithm is brought into full play, and the characteristics of Spark make the reuse ratio of the intermediate results more efficient, hence the data processing capacity per unit time is increased, namely the algorithm proposed in this paper has high execution efficiency. However, with the increase of the data volume, the running efficiency of the algorithm is reduced due to the shortage of memory resources. Even so, the curve of each segment is generally smooth in the figure, that is, the algorithm has a high efficiency and can meet the quasi-realtime of the massive historical data processing for electric power grid.

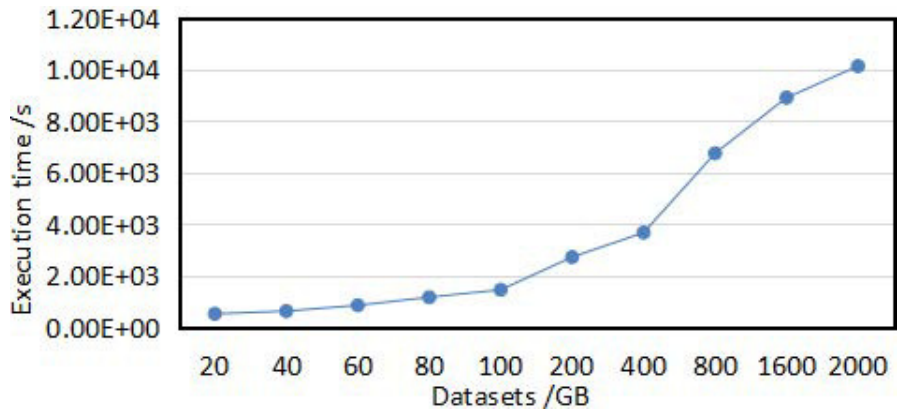

FIGURE 4. The relation between execution time and datasets size

5.3.2. *The Speed-up Ratios and Scalability.* The number of computing nodes measures the Speed-up ratios of a system and represents the number of parallel processes, the Speed-up ratios is the performance of a parallel algorithm when the amount of data is constant and the number of the computing nodes is increased continuously, and the ideal Speed-up ratios is linear [21]. While the scalability refers to performance of a parallel algorithm when the amount of datasets and computing nodes are increased proportionally. The experimental results are shown in figure 5:

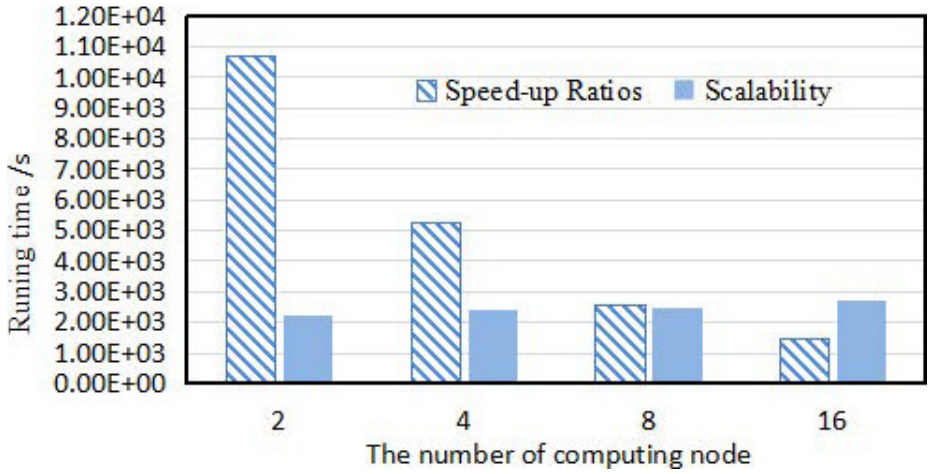

FIGURE 5. The Speedup and scalability

In this experiment, the experimental data of 100GB is used to evaluate the Speed-up ratios of the algorithm presented. While the experimental data of 25GB, 50GB, 100GB, 200GB and the computing nodes of 2, 4, 8, 16 are used respectively to evaluate its scalability. It can be seen from the Figure that the ideal effect of speed-up ratios can't be achieved due to the communication among computers and other costs, but the algorithm SP-MATCH achieves good acceleration ratios. However, when the number of computing nodes is increased to 8 and 16, the running time of the algorithm is slightly longer due to the resource consumption of the hardware and platform. Even so, the running time of the algorithm is generally maintained at a considerable level, namely the algorithm SP-MATCH has a good scalability.

6. **Conclusion.** The contributions of this paper are summarized as following:

1. We improve and optimize the traditional Simhash algorithm (IPOP-Simhash), in which a FGA (Fingerprint Generation Algorithm) based on the traditional Simhash is proposed, and a weight calculation method based on the combination of subjective and objective analysis is used. Moreover, a FIT (Fingerprint Index Tree) and a FCA (Fingerprint Classification Algorithm) based on graph clustering analysis are designed, and a fingerprint search strategy based on the FIT and FCA is proposed.

2. A algorithm (SP-MATCH) is designed based on Spark and the IPOP-Simhash, which can effectively solve the problems that the approximately duplicate records detection of distributed electric power big data and the efficient utilization of computing resource, simultaneously improve the detection precision and Recall Ratio, break through the bottlenecks caused by the high-dimensional and massive electric power grid data.

3. The experimental results show that the proposed method not only can accurately realize the approximately duplicate records detection of electric power big data, but also has a higher stability and running efficiency.

In the future work, we plan to study the optimization strategys of the proposed algorithm, consider more large-scale datasets and computing nodes, make the algorithm more effective on electric power big data pre-processing.

Acknowledgments. The paper is supported by National Natural Science Foundation of China (No. 61271115).

REFERENCES

[1] Y. S. Xue and Y. N. Lai, Integration of Macro Energy Thinking and Big Data Thinking Part One Big Data and Power Big Data, *Automation of Electric Power Systems*, vol. 40, no. 1, pp. 1–8, 2016.
[2] Z. Y Qu, L. Q. Sun, S. Q. Xun, et al. Power Big Data Distributed Index Based on B+ Tree And Inverted Index, *Journal of Northeast Dianli University*, vol. 36, no. 5, pp. 80–85, 2016.
[3] J. Z. Li, H. Z. Wang, H. Gao, State-of-the- Art of research on big data usability, *Journal of Software*, vol. 27, no. 7, pp. 1605–1625, 2016.
[4] N. Wang and J. Li, Two-tiered correlation clustering method for entity resolution in big data, *Journal of Computer Research & Development*, vol. 51, no. 9, pp. 2108–2116, 2014.
[5] Y. J. Yan, G. H. Sheng, Y. F. Chen, X. C. Jiang, Z. H. Guo and S. P. Qin, Cleaning Method for Big Data of Power Transmission and Transformation Equipment State Based on Time Sequence Analysis, *Automation of Electric Power Systems*, vol. 39, no. 07, pp. 138–144, 2015.
[6] W. F. Fan and J. Z. Li, Query preserving graph compression, *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, New York, USA, 157–168, October, 2012.
[7] L. M. Zhen, X. C. Yang and B. Wang, An Entity Resolution Approach Based on Attributes Weights, *Journal of Computer Research & Development*, vol. 50, no. 36, pp. 281–289, 2013.
[8] H. P. Liu, C. Q. Jin and A. Y. Zhou, A Pattern-Based Entity Resolution Algorithm, *Chinese Journal of Computers*, vol. 38, no. 9, pp. 1796–1808, 2015.
[9] R. R. Kannan, D. R. Abarna and G. Aswini, Effective Progressive Algorithm for Duplicate Detection on Large Dataset, vol. 36, no. 06, pp. 325–332, 2016.
[10] J. Zobel and K. Verspoor, Evaluation of a Machine Learning Duplicate Detection Method for Bioinformatics Databases, *International Journal of Digital Content Technology and Applications*, vol. 12, no. 9, pp. 4–12, 2015.
[11] G. Valkenhoef, R. Loane and D. Zarin, Previously Unidentified Duplicate Registrations of Clinical Trials:An Exploratory Analysis of Registry Data Worldwide, *Systematic Reviews*, vol. 5, no. 1, pp. 116–117, 2016.
[12] W. D. Wang and Z. W. Sun, Big data analysis and parallel load forecasting of electric power user side, *Proceedings of the Chinese Society of Electrical Engineering*, vol. 35, no. 3, pp. 527–537, 2015.
[13] A. LI, and S. J. WU, Data deduplication techniques, *Journal of Software*, vol. 21, no. 5, pp. 916–929, 2010.
[14] G. S. Manku, A. Jain and A. D. Sarma, Detecting near-duplicates for web crawling, *Proceedings of the 16th international conference on World Wide Web*, New York, USA, 141–150, October, 2007.
[15] W. Q. Jiang, X. R. LI and J. Qian, Application of improved FCM algorithm in outlier processing of power load, *Proceedings of the Chinese Society of Universities for Electric Power System & Its Automation*, vol. 23, no. pp. 1–5, 2011.
[16] C. H. Huang, J. Yin and F. Hou, A Text Similarity Measurement Combining Word Semantic Information with TF-IDF Method, *Chinese Journal of Computers*, vol. 34, no. 5, pp. 856–864, 2011.
[17] J. Z. Zhu, Y. T. Jia, J. Xu, J. Z. Qiao, Y. Z. Wang and X. Q. Cheng, SparkCRF: A Parallel Implementation of CRFs Algorithm with Spark, *Journal of Computer Research and Development*, vol. 53, no. 8, pp. 1819–1828, 2016.
[18] Li C, Cao P, Li J, Zhao B, Review on Reactive Voltage Control Methods for Large-Scale Distributed PV Integrated Grid, *Journal of Northeast Dianli University*, vol. 37, no. 2, pp. 82–87, 2017.
[19] H. Pcke, A. Thor and E. Rahm, Evaluation of entity resolution approaches on real-world match problems, *Proceedings of the Very Large Data Bases Endowment*, vol. 3, no. 1, pp. 484–493, 2010.
[20] Z. Y. Qu, S. Chen, F. Yang and L. Zhu, An attribute reducing method for electric power big data preprocessing based on cloud computing technology, *Automation of Electric Power Systems*, vol. 38, no. 8, pp. 67–71, 2014.