Acta Technica *62* (2017), No. 7A, 887–900

Random response cloud computing resource association rules mining for privacy data

Huabing Wang[1], Chunrong Qiu[1],
Yuzhu Cheng[1]

Abstract. To strengthen data privacy protection, and improve data mining accuracy, random response mode is adopted to design privacy protection mining method based on association rules. Granular computing method and technology are applied to mining fields of association rules in data mining, and mining to association rules is researched in more extensive way from another perspective in this paper. Firstly, partial concealing mode is adopted to conceal and transform original privacy data and improve data security; secondly, associated frequent item set is utilized to construct simple and efficient privacy protection mining algorithm; finally, algorithm proposed is verified to have higher privacy and accuracy through theoretical analysis and experimental verification. After classical association rules mining algorithm is analyzed and researched in detail with its characteristics and restrictions summarized through examples in this paper, association rules mining model based on granular computing is proposed, which makes theoretical preparation for proposal and construction of association rules pick-up algorithm based on granular computing. Experimental result shows that association rules mining method based on granular computing is feasible and effective.

Key words. Association rules, Cloud computing, Data resource, Random response

1. Introduction

In recent years, data mining technology has been widely concerned by information industry circle, which is the inexorable outcome of paradoxical movement between rapidly increasing data quantity and increasingly poor information amount. Systematic and intensive research to data mining technology is objective requirement of global information-based development. Data mining technology includes many research fields, of which association rule is an important research direction, having

[1]Department of Software, Changsha Social Work College, Changsha Hunan Province, China

vitally important application value in business decision. This topic mainly makes related research to association rules mining. Traditional association rules mining algorithm, such as Apriori algorithm and its improved algorithm etc., makes mining to certain and accurate concept, and it is difficult to mine non-accurate or blurry concept. Through experiment, it can be found that main calculation to search frequent item set lies in frequent 2-item set generation, and frequent 2-item set generation process is Apriori algorithm mining bottleneck, and therefore, a kind of new association rules mining algorithm based on fuzzy sets is proposed in this paper, fuzzy set theory and semantic association rule concept are introduced in the algorithm, reasonable and non-accurate semantic translation is made to numerical attribute of database, and algorithm efficiency is improved by improving size of item set of pruning part scanned, which avoids exponential growth tendency of length of set scanned. Because core problem of Apriori algorithm is to find the maximum item set, the process to find the maximum item set is global search process, and genetic algorithm is a kind of global optimization algorithm, and avoids local optimization in search process. Therefore, truly useful rules can be found by applying genetic algorithm to rule finding and extraction. Therefore, a kind of association rules mining algorithm based on genetic algorithm is proposed in this paper, which mainly mines quantitative association rules, and algorithm mainly includes association rules coding method design, fitness function construction and genetic operator improvement etc. According to 2 kinds of association rules mining algorithm that are proposed and designed in this paper and that are based on computational intelligence, we extract association rules respectively by taking medical database and student database as mining prototype, and make experimental analysis; experimental result verifies effectiveness of 2 kinds of algorithm and also illustrates wide application prospect of association rules mining.

2. PFP algorithm

PFP algorithm is a kind of parallel FP-Growth algorithm that was proposed by Li et al. of Google Beijing Research Institute in 2008 and that is based on MapReduce frame. Because of high expansibility and high fault tolerance of MapReduce frame, algorithm can process big data in relatively good way.

Basic thought of algorithm is to transform transaction database into a new "intra-group dependency transaction" database and distribute it to corresponding Reducer, and in the process of recursive construction of Reducer to FP-Tree, local FP-Tree generated through different "intra-group dependency transaction" is mutually independent.

2.1. PFP algorithm description

PFP algorithm divides database into P blocks, which shall be processed by P Mappers.

The algorithm shall make MapReduce operation for 3 times in total:

Step 1: scan database, generate first-order frequent item set F-List and rank

item set according to size of support degree. Divide i frequent items in F-List into Q groups, called as G-List, and distribute a gid to each group.

Step 2: scan database again, rank items in transaction according to size of support degree in F-List, and delete items in transaction not existing in F-List.

Step 3: make parallel FP-Growth mining to data processed.

In MapReduce operation of step 3:

1) Mapper For Mapper

1 Generate 1 Hash table H by G-List, key of table is frequent item, and value of table is gid corresponding to frequent item.

2 Traverse all transaction data, search items of transaction data in H, and distribute all "left" items of the item to corresponding group according to corresponding gid of item; delete corresponding line of the gid in H, to prevent repeated distribution of "left" data of the item.

2) Reducer For Reducer

1 Each Reducer obtains 1 group from G-list and drag to obtain data of Mapper distributed according to gid of group.

2 After drag and obtaining, construct FP-Tree according to data dragged and obtained, invoke FP-Growth to FP-Tree, and generate K^{th}-order frequent item set.

See algorithm 1-1 and algorithm 1-2 for pseudocode of MapReduce operation of step 3.

Algorithm 1-1 Map algorithm of step 3 of PFP

```
void Map (key k, value T_i)
{
  G-List.Load();
    HashTable H = G-List.getHashTable();
    Item a[] = T_i.Split();
    for j = a.length-1 to 0
    {
      HashNum = H.getHashNum(a[j]);
      if(HashNum! = Null)
      {
        H.deleteRowByVal(HashNum);
        OutPut(<HashNum, a[0] + a[1] + ... +a[j]>);
      }
    }
}
```

Algorithm 1-2 Reduce algorithm of step 3 of PFP

```
void Reduce (key gid, value DB_gid)
{
  G-List.Load();
nowGroup = G-List.getGroupByGid(gid);
    LocalFPtree = new FP-tree(Null);
    foreach T in DB_gid
```

```
    LocalFPtree.insert_tree(T, LocalFPtree.root, G-List);
  foreach item in nowGroup
{
    Heap HP = new Heap();
    LocalFPtree.FPGrowth(LocalFPtree.root, item, HP);
    foreach pattern in HP
      OutPut(<null, pattern+pattern.supp()>);
    }
}
```

2.2. PFP algorithm analysis

Because data communication between components is the uppermost bottleneck of distributed system, this paper mainly analyzes data transmission quantity of algorithm in implementation, i.e. data communication complexity.

Assumed that the number of transaction data in database is n, the length of the i^{th} transaction is l_i, then data quantity of the whole transaction data can be expressed as:

$$M_{DB} = \sum_{i=1}^{n} l_i \,. \tag{1}$$

Data transmission quantity in F-List calculation process in step 1 of PFP algorithm is MDB, only Map operation is required in step 2 where Reduce operation is not required, and therefore data transmission quantity is 0.

Size of data needing to be transmitted in step 3 can be expressed as:

$$\begin{aligned} M_{Dup} &= \sum_{i=1}^{n} \left(f(I_1 I_{l_i}) + 2f(I_2 I_{l_i}) + ... + l_i f(I_{l_i} I_{l_i})\right) \\ &= \sum_{i=1}^{n} l_i + \sum_{i=1}^{n} \left(f(I_1 I_{l_i}) + 2f(I_2 I_{l_i}) + ... + (l_i - 1) f(I_{l_i - 1} I_{l_i})\right) \\ &= M_{DB} + \sum_{i=1}^{n} \sum_{j=1}^{l_i - 1} j f(I_j I_{l_i}) \end{aligned} \tag{2}$$

Where, I represents item in transaction, and function f(I_j, I_k) is defined as:

$$f(I_j, I_k) = \begin{cases} 1 & I_j, I_k \text{ in } group \text{ among } I_j = I_k \\ 0 & I_j, I_k \text{ not in } group \end{cases} \tag{3}$$

I_j and I_k are not in the same *group*, or $I_j = I_k$

I_j and I_k are in the same *group*

Add data transmission quantity of 3 steps together to obtain data transmission

quantity of algorithm in the whole process:

$$M = M_{DB} + M_{Dup} = 2M_{DB} + \sum_{i=1}^{n} \sum_{j=1}^{l_i - 1} jf(I_j I_{l_i}) . \tag{4}$$

According to formula 3 and formula 4, data transmission quantity of PFP is concerned with grouping of G-List. When the number of grouping Q = 1, i.e. G-List = F-List, M = $2M_{DB}$, the minimum value can be obtained, but PFP will be degraded into a single-unit algorithm. Assumed that the number of item in F-List is m, when the number of grouping Q = m, $M = (3/2)M_{DB} + (1/2) \sum_{i=1}^{n} l_i^2$, the maximum value can be obtained, but system will have the highest parallelism.

For a distributed system, the greater the number of component is, the higher the system parallelism will be, and the higher the processing efficiency will be, which will cause increase of data transmission quantity simultaneously. According to analysis on PFP algorithm, when average length l of transaction data is relatively long, data transmission quantity will increase rapidly with increase of the number of component, which causes reduction of system operation efficiency, thus becoming system operation bottleneck.

3. PFP-P algorithm

Basic starting point of PFP-P algorithm is to replace mining to all frequent item sets with mining to closed frequent item set. Different from mining to all frequent item sets, mining to closed frequent item set does require "intra-group dependency transaction" of all items in transaction data, but "intra-group dependency transaction" of partial suffix items to reduce data transmission quantity. In addition, another advantage of mining to closed frequent item set is that mining result data can be significantly reduced under the premise that information completeness is guaranteed, which is convenient for storage and further processing.

3.1. Suffix item list

For the convenience of discussion, this paper maps transaction data as transaction mode, which means that transaction data ranked according to sequence of items in F-List after non-frequent items in transaction data are deleted is expressed as T; item in transaction T is expressed as I; support degree of mode T and submode is expressed as sup(), and the minimum support degree meeting frequent mode is expressed as sup_min.

Definition 2.1 Closed frequent mode. Assumed that submode X = $I_1 I_2 \ldots I_k$, sup(X) = u≥sup_min; if $\forall I_p$, where p ≥ k+1, sup($I_1 I_2 \ldots I_k \ldots I_p$) <u, then X is called as closed frequent mode of item I_k, and I_k is called as suffix of X. Closed frequent mode corresponds to closed frequent item set one by one.

Definition 2.2 Suffix item. For submode X = $I_1 I_2 \ldots I_k$ of T, if sup(I_1) = sup(I_2) = ... = sup(I_k) = u is met, then:

Definition 2.2.1 If $\forall I_p$, where p $\geq$k+1 and sup(I_p) <u, then I_k is called as u support degree suffix item of T.

Definition 2.2.2 For any item I_i and any suffix item I in X ($I_i \neq I$), if sup(I_iI)<u, then I_i is called as u support degree suffix item of T.

Theorem 2.1 Suffix of closed frequent mode must be suffix item.

Prove. Prove through proof by contradiction. Assumed that submode X = $I_1I_2 \ldots I_k$ of T is closed frequent mode, sup(X) = u and I_k is not suffix item, then according to definition 2.2.1, item I_p must exist in T, where p $\geq$k+1 and sup(I_p) $\geq$u, because support degree of item in T is monotonous and does not increase, sup(I_p) = u; according to definition 2.2.2, sup(I_kI_p) = u; therefore, a submode X' = $I_1I_2 \ldots I_kI_p$ must exist in T, to make sup(X') = u; according to definition 2.1, X is not closed frequent mode, which conflicts with assumption, so original conclusion is verified.

According to theorem 2.1, when Mapper distributes data to Reducer, it just need to distribute data to suffix item to obtain enough information to construct closed frequent item set, and it does not need to distribute data to non-suffix item. List consisting of all suffix items in transaction data is called as suffix item list, and it is a subset of F-List.

3.2. Construction of suffix item list

Suffix item list can only be constructed after F-List is constructed and transaction data is deleted and ranked. But in construction of suffix item list, MapReduce process does not need to be made independently, and the process can be made in MapReduce process deleting and ranking transaction database.

Divide items in transaction mode T into several sets according to definition 2.1.1 and support degree, final item of item of the same support degree turns to be the first suffix item, and recursively judge support degree of composite mode of other items and suffix item according to definition 2.1.2, if support degree of composite mode equal to support degree of item exists, then delete it and change it to non-suffix item, and otherwise add it to suffix item list.

Although the method is simple in flow, and can be easily understood, it has infeasible actual operation. Composite mode of non-suffix item and suffix item has huge quantity, and to judge global support degree of composite mode, support of second-order support degree matrix of data is required; it costs much to generate second-order support degree matrix, which may not reduce data transmission quantity, but increase it.

Therefore, this paper proposes a kind of suffix item list construction algorithm based on vertical format mining, where suffix item list can be constructed by traversing transaction mode, and the process can also be made in MapReduce process for transaction data preprocessing.

Basic thought of algorithm: identify each item with item position feature (such as TID set where item is located), according to suffix item definition, the rightmost item of many items having the same position feature under the same support degree must be suffix item, while other items cannot be suffix items, thus they can be deleted. So suffix item list can be constructed.

The algorithm is described as follows:

1) Mapper For Mapper

1. F-List;Read F-List;

2Traverse transaction mode T, and transform TID of T into corresponding integer value ID_NUM according to certain rule; obtain support degree sup of item I in T according to F-List;

3. Output sup as key, and value pair <I, ID_NUM> is value;

2) Reducer For Reducer

1. F-List;Read F-List;

2. Summarize the same I input to value, rank its ID_NUM from the small to the large and splice it into a character string P as position feature (if the character string is excessive long, MD5 algorithm etc. can be considered to extract its feature);

3. Compare P of items of string P with the same length, for items with the same P, only rightmost item shall be reserved and other items shall be deleted.4. Output all items I reserved.3) Driver For Driver1. Summarize item I of each Reducer output, and delete item not being item I in F-List to obtain suffix item list P-List;See algorithm 2-1 and algorithm 2-2 for pseudocode of Map and Reduce algorithm.

Algorithm 2-1 map algorithm for suffix item list construction

```
void Map(key k, value T_i)
{
  F-List.Load();
  int ID_NUM = T_i.TID.toInteger();
  Item a[] = T_i.Split();
  for j = a.length-1 to 0
    {
      int sup = F_list.getSup(a[j]);
      OutPut(sup, <a[j], ID_NUM >);
  }
}
```

Algorithm 2-2 Reduce algorithm for suffix item list construction

```
void Reduce (key key, value[] Pairs)
{
F-List.Load();
pairlist<item, string> plist = new pairlist();
foreach pair in Pairs
{
  if(plist.find(pair.first))
    plist.getPair(pair.first).second.
    concatenateInOrder(pair.second);
  else
    plist.addPairInOrder(pair, F-List);
}
foreach pair in plist
```

```
{
  if(pair.compareBySecondPart(plist. getRests(pair))
    plist.delete(pair);
}
    OutPut(<null, plist>);
}
```

3.3. PFP-P algorithm description

Similar to PFP algorithm, PFP-P algorithm also makes 3-step MapReduce operation, and execution of former 2 steps is slightly different from that of PFP algorithm.

In step 1, only F-List shall be generated without generation of G-List.

In step 2, after transaction data is ranked and deleted, suffix item list construction algorithm shall be executed to generate P-List; divide i frequent items in P-List into Q groups as G-List and every group obtains 1 gid.

In step 3, Mapper end in MapReduce operation shall make the operation equal to PFP algorithm.

Redecer For Redecer

1. Each Redecer obtains 1 group from G-List, and drag to obtain data of Mapper distributed according to gid of group.

2. After drag and obtaining, obtain condition data DB_{item} of each item in G-List.

3. Invoke closed frequent item set mining algorithm to each DB_{item} to construct FP-Tree and generate closed frequent item set. Relatively representative CLOSET algorithm [11] based on FP-Tree mining is adopted as closed frequent item set mining algorithm in this paper.

See algorithm 2-3 for pseudocode of Reduce operation in step 3 of algorithm.

Algorithm 2-3 Reduce algorithm in step 3 of PFP-P

```
void Reduce (key gid, value DB_gid)
{
G-List.Load();
nowGroup = G-List.getGroupByGid(gid);
foreach item in nowGroup
{
  DB_item.get(item, DB_gid);
  Heap HP = new Heap();
  CLOSET(item, DB_item , G-List , HP);
  foreach pattern in HP
    OutPut(<null, pattern+pattern.supp()>);
}
}
```

3.4. PFP-P algorithm analysis

This section analyzes data communication complexity of PFP-P. Quantity of data transmitted in step 1 of algorithm is the same with that of PFP, being M_{DB}. Quantity of data needing to be transmitted in step 2 to construct P-List is also M_{DB}.

Data transmission quantity of step 3 is:

$$M_{Dup} = M_{DB} + \sum_{i=1}^{n} \sum_{j=1}^{l_i - 1} jf(I_j I_{l_i}) g(I_j) \,. \tag{5}$$

Where function g(I) is defined as follows:

$$g(I) = \begin{cases} 0 & I \text{ is non-suffix item.} \\ 1 & I \text{ is suffix item.} \end{cases} \tag{6}$$

Add data transmission quantity of 3 steps together to obtain data transmission quantity of algorithm in the whole process:

$$M = M_{DB} + M_{Dup} = 3M_{DB} + \sum_{i=1}^{n} \sum_{j=1}^{l_i - 1} jf(I_j I_{l_i}) g(I_j) \,. \tag{7}$$

For the convenience of discussion, assumed that average length of transaction data is l, the maximum value of grouping is obtained, and average total data transmission quantity of PFP and PFP-P can be obtained according to formula 4 and 7:

$$\overline{M_1} = (3/2)nl + (1/2)nl^2 \,. \tag{8}$$

$$\overline{M_2} = 3nl + n \sum_{j=1}^{l-1} jg(I_j) \,. \tag{9}$$

According to formula 8 and formula 9:

$$\overline{M_2} - \overline{M_1} = n \sum_{j=1}^{l-1} jg(I_j) - n(1/2)l(1-l) + nl \,. \tag{10}$$

When all g(I) is 1, which means that all items of transaction data are suffix items, value of formula 10 is nl of the third item, which means that PFP-P transmits one more M_{DB} data than PFP. When g(I) is item of 0, which means that sum of coefficient j of non-suffix item is greater than average length l of transaction, data transmission quantity of PFP-P will be lower than that of PFP. When average length l increases, effect of nl of the third item in formula 10 on total transmission quantity will decrease rapidly, and what plays a decisive role will be former 2 items in the formula. When l is relatively great, only few non-suffix items are required to make

sum of coefficient exceed 1.

According to suffix item list construction algorithm, assumed that the number of item meeting support degree sup is k_{sup}, then probability of non-suffix item in item with support degree being sup can be expressed as:

$$P(\exists I \notin PList) = Min\left(1, \frac{k_{\sup} - 1}{n(n-1)(n-2)...(n-\sup+1)}\right). \tag{11}$$

Assumed that the minimum support degree of frequent item in transaction data is minsup, and the maximum support degree of item is maxsup, then

$$M_{DB} = \sum_{i=1}^{n} l_i = \sum_{i=\min\sup}^{\max\sup} ik_i \tag{12}$$

Assumed that n and support degree of each item are kept unchanged in transaction data, according to formula 12, k will increase with increase of l; according to formula 11, probability that item in transaction data is non-suffix item also increases with increase of it. Therefore, for data with relatively great average transaction length, PFP-P can reduce average transmission quantity of data effectively.

Taking data {T1 = (a_1, a_2, ... a_{50}), T2 = (a_1, a_2, ... a_{100})} as example, assumed that the minimum support degree threshold is 1 with the maximum grouping adopted, then in PFP, data shall be divided into 100 groups, and total transmission quantity of data is 150*(3/2)+(1/2)*(50*50+100*100) = 6475. But in PFP-P, a_{50} is suffix item of support degree 2, while a_{100} is suffix item of support degree 1, and other items are non-suffix items with division of 2 groups (a_{50} and a_{100}), and total transmission quantity of data is 150*3+49+99 = 589.

4. Experiment and result analysis

Experiment is based on Hadoop platform [12], with 8 computational nodes in total, including 1 NameNode and 7 DataNodes. Each node adopts Intel Core2 2GHz processor, with 2GB memory and 260GB disk space, connected through 100M Ethernet while operating system adopts CentOS6.2.

This paper rewrites PFP algorithm in mahout open source project as PFP-P algorithm, and makes contrast experiment with PFP algorithm and PFP-C algorithm under above experimental environment. For the convenience of description, we describe experimental dataset in the form of D#T#C#, where D represents transaction length, i.e. dimension; T represents transaction quantity; C represents transaction potential, i.e. the number of possible values per dimension. 3 experimental datasets are adopted, respectively representing 3 kinds of data: accidents dataset [13] (D570T340000C45) represents low-dimension data with a great deal of transaction quantity; breast cancer dataset [14] (D24481T78C100) represents high-dimension data with few transaction quantity; generated dataset testdata (D15000T8000C100) represents high-dimension data with a great deal of transaction quantity.

What is shown in Fig.1, Fig.2 and Fig.3 respectively is change of mining time

of 3 kinds of algorithms in 3 datasets with the minimum support degree threshold (sup_min).

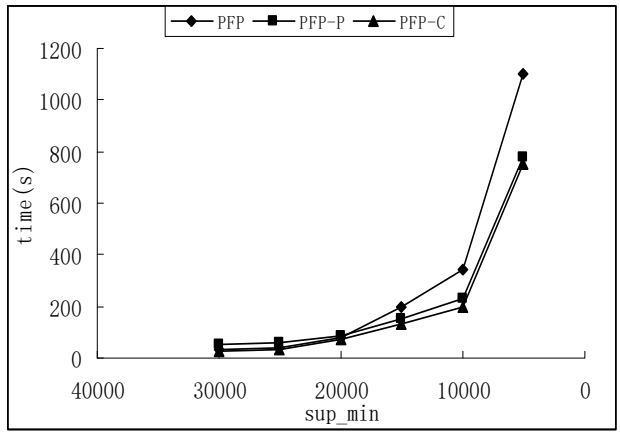

Fig. 1. Comparison to performance of 3 kinds of algorithms on accidents dataset mining

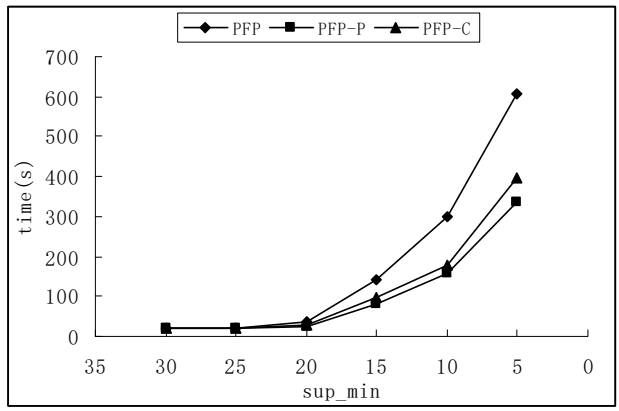

Fig. 2. Comparison to performance of 3 kinds of algorithms on breast cancer dataset mining

According to Fig.1, for low-dimension data, when value of sup_min is relatively great, mining time of PFP-P algorithm is longer than that of other 2 kinds of algorithms; with decrease of sup_min, mining time of PFP algorithm increases rapidly, while increase of PFP-P and PFP-C is relatively slow, but difference of 3 kinds of algorithms is not quite large. When sup_min is relatively great, because PFP-P has 1 more suffix item list calculation step when compared with other 2 kinds of algorithms, difference between quantity of closed frequent item and quantity of frequent item is comparatively small, and therefore, PFP-P algorithm does not have advantage; with decrease of sup_min, quantity of frequent item increases rapidly, advantage of PFP-P is embodied gradually. In addition, for low-dimension data, be-

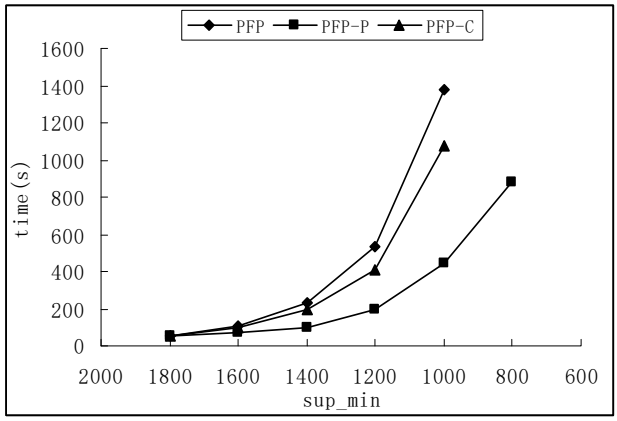

Fig. 3. Comparison to performance of 3 kinds of algorithms on testdata dataset mining

cause probability of non-suffix item is comparatively small, PFP-P algorithm cannot reduce transmission quantity of grouped data massively, and therefore mining time difference of PFP-P and PFP is not large, and the whole performance is slightly superior to that of PFP-C algorithm.

According to Fig.2 and Fig.3, increase of mining time of PFP-P is obviously slower than that of PFP and PFP-C, while for the third dataset testdata, PFP-P shows more obvious advantage, because for high-dimension data, although frequency for appearance of non-suffix item increases obviously, which makes PFP-P reduce transmission quantity of grouped data effectively, for breast cancer dataset with relatively small transaction quantity, its data quantity is not large, and transmission of grouped data is not main system operation bottleneck, and therefore, although PFP-P can reduce transmission time of grouped data, its contribution to total mining time reduction is not large; advantages of PFP-P can only be embodied for testdata with high-dimension and massive data.

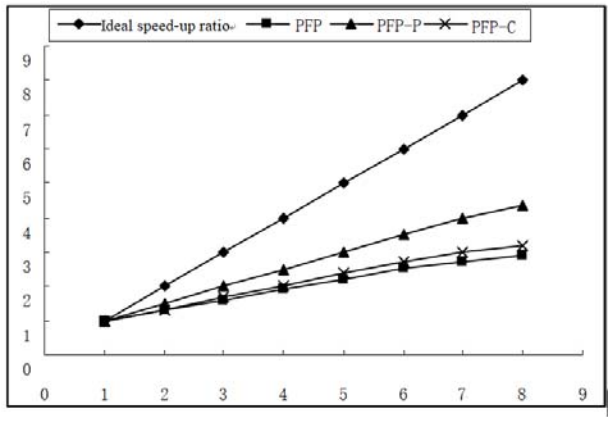

Fig. 4. Comparison on speed-up ratio of 3 kinds of algorithms

Fig.4 is speed-up ratio comparison figure of 3 kinds of algorithms on testdata dataset mining under the same sup_min. Speed-up ratios of 3 kinds of algorithms are lower than ideal speed-up ratio, and speed-up ratio of PFP-C is similar to that of PFP, while speed-up ratio of PFP-P is superior to that of other 2 kinds of algorithms obviously, which is caused by the fact that PFP-P reduces transmission quantity of grouped data in mining, and lowers I/O between nodes and network bandwidth consumption, so with increase of nodes, speedup effect will be superior to that of PFP and PFP-C obviously.

5. Conclusion

Based on PFP algorithm, a kind of parallel closed frequent item set mining algorithm PFP-P based on suffix item list is proposed in this paper. This algorithm replaces mining to all frequent items in original algorithm with mining to closed frequent item set to improve mining efficiency; aimed at features of closed frequent item set, suffix item list is introduced in mining process to reduce transmission quantity of grouped data in mining process, and lower internal consumption of system. Experiment shows that the algorithm is superior to original algorithm in average performance, and in decrease of the minimum support degree threshold, it can shorten mining time effectively; the algorithm can lower consumption of communication between nodes effectively with good speedup quality; compared with processing to low-dimension dataset, the algorithm has more advantages in processing high-dimension dataset, and therefore, the algorithm is more applicable to mining task of massive high-dimension data.

Acknowledgement

Hunan natural science research project “Extraction and modeling of rapeseed seed characteristic information based on multi scale geometric analysis”.

References

[1] W. S. Pan, S. Z. Chen, Z. Y. Feng.: *Investigating the Collaborative Intention and Semantic Structure among Co-occurring Tags using Graph Theory.* International Enterprise Distributed Object Computing Conference, IEEE, Beijing, (2012), 190–195.

[2] J. W. Chan, Y. Y. Zhang, and K. E. Uhrich: *Amphiphilic Macromolecule Self-Assembled Monolayers Suppress Smooth Muscle Cell Proliferation*, Bioconjugate Chemistry, *26* (2015), No. 7, 1359–1369.

[3] Y. Y. Zhang, E. Mintzer, and K. E. Uhrich: *Synthesis and Characterization of PEGylated Bolaamphiphiles with Enhanced Retention in Liposomes*, Journal of Colloid and Interface Science, *482* (2016), 19–26.

[4] J. J. Faig, A. Moretti, L. B. Joseph, Y. Y. Zhang, M. J. Nova, K. Smith, and K. E. Uhrich: *Biodegradable Kojic Acid-Based Polymers: Controlled Delivery of Bioactives for Melanogenesis Inhibition*, Biomacromolecules, *18* (2017), No. 2, 363–373.

[5] Z. Lv, A. Halawani, S. Feng, H. Li, & S. U. Réhman: *Multimodal hand and foot gesture interaction for handheld devices.* ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), *11* (2014), No. 1s, 10.

[6] Y. Z. Chen, F. J. Tang, Y. Bao, Y. Tang, G. D. Chen: *A Fe-C coated long period fiber grating sensor for corrosion induced mass loss measurement.* Optics letters, *41* (2016), 2306–2309.

[7] Y. Du, Y. Z. Chen, Y. Y. Zhuang, C. Zhu, F. J. Tang, J. Huang: *Probing Nanostrain via a Mechanically Designed Optical Fiber Interferometer.* IEEE Photonics Technology Letters, *29* (2017), 1348–1351.

[8] W. S. Pan, S. Z. Chen, Z. Y. Feng: *Automatic Clustering of Social Tag using Community Detection.* Applied Mathematics & Information Sciences, *7* (2013), No. 2, 675–681.

[9] Y. Y. Zhang, Q. Li, W. J. Welsh, P. V. Moghe, and K. E. Uhrich: *Micellar and Structural Stability of Nanoscale Amphiphilic Polymers: Implications for Anti-atherosclerotic Bioactivity*, Biomaterials, *84* (2016), 230–240.

[10] J. W. Chan, Y. Y. Zhang, and K. E. Uhrich: *Amphiphilic Macromolecule Self-Assembled Monolayers Suppress Smooth Muscle Cell Proliferation*, Bioconjugate Chemistry, *26* (2015), No. 7, 1359–1369.

[11] D. S. Abdelhamid, Y. Y. Zhang, D. R. Lewis, P. V. Moghe, W. J. Welsh, and K. E. Uhrich: *Tartaric Acid-based Amphiphilic Macromolecules with Ether Linkages Exhibit Enhanced Repression of Oxidized Low Density Lipoprotein Uptake*, Biomaterials, *53* (2015), 32-39.

[12] Y. Y. Zhang, A. Algburi, N. Wang, V. Kholodovych, D. O. Oh, M. Chikindas, and K. E. Uhrich: *Self-assembled Cationic Amphiphiles as Antimicrobial Peptides Mimics: Role of Hydrophobicity, Linkage Type, and Assembly State*, Nanomedicine: Nanotechnology, Biology and Medicine, *13* (2017), No. 2, 343–352.

Received May 7, 2017