Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/070648

A Parallel Computing Method of Social Tag Cooccurrence Relation

Xiangqian Wang[1], *Huizong Li*[1,*] *and Yerong He*[2,*]

[1]School of Economics and Management, Anhui University of Science and Technology, Huainan 232001, P. R. China
[2]School of Economics and Management, Huainan Normal University, Huainan 232038, P. R. China

Received: 16 Apr. 2013, Revised: 18 Aug. 2013, Accepted: 20 Aug. 2013
Published online: 1 Nov. 2013

Abstract: Cooccurrence relation extraction of abundant tags is very significant for information retrieve and tag recommendation. This paper presents a parallel extraction method of tag cooccurrence relation based on cloud computing. At last, two experiments show that the parallel extraction method can improve the performance of retrievals greatly and have good scalability. The research may play a role in the facilitative effect on the expanding application of social tagging system with abundant, multiplex, complex annotated data.

Keywords: Tag cooccurrence relation, social tagging system, parallel computing, MapReduce model.

1 Introduction

The appearance of the Web2.0 prompts the growth momentum of the network information resources. In the Web2.0 era, the users visit and create the network information resources. Namely, the network resources are propagated and shared by web users and forms many systems with strong sociality, such as forum, blog, wikipedia, micro-blog, online social networking platform, social tagging system and so on. The social tagging system allows web users to choose appropriate tags by their understanding and perceiving of the resources to annotating web resources in a free and open environment. These tags could be the words and terms in the commonly vocabularies, or the words innovated by users. Therefore, the social tagging system develops a new taxonomy, Folksonomy, which is a compound word by using Folks and Taxonomy in the Web2.0. Folksonomy was first introduced by Thomas Vander Wal [1] in 2004 when he discussed the information structure of the 'delicious' and 'flickr', he described that the structure is a Bottom-up social classification. Unlike traditional expert classification, Folksonomy is viewed as a flat and non-level tag classification method where the tags are created by non-professionals [2]. Tag is the core element of the Folksonomy, which is a free form keyword or index term [3].

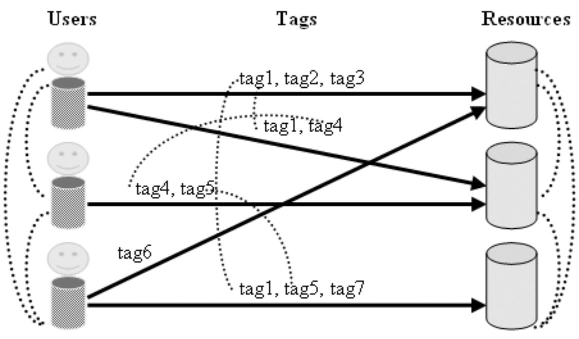

Fig. 1: Model of social tagging system.

The social tagging system produces many annotations, and forms a multi-relation between users, resources and tags. These relations contain the unary relation, binary relation and ternary relation. Then, how to dig social relations and build its relation network by using the annotation data produced in social tagging system becomes a very important issue, particularly, how to extract the relations from tags is very useful for study on the tag cluster, tag ontology, and tag recommendation. The relationship between two tags usually appears as a cooccurrence form. Many users annotate abundant resources, which produce massive tags in social tagging

* Corresponding author e-mail: lihz_aust@sina.com, hmy.2007happy@163.com

system, so how to extract the tag cooccurrence network in mass data space becomes a urgent task for the deeply application of the social tagging system.

In recent years, cloud computing was proposed to deal with huge data, which provide a new parallel computing framework for massive data processing. Google proposed a distributed parallel programming model, MapReduce [4], which is one of core computing modes of cloud computing. As for huge tags, using the MapReduce model to extract the tags relationship is very adaptable. In this paper, we proposed a novel method based on MapReduce model to extract the social tags cooccurrence relations.

2 Related Work

2.1 Social Tagging System

Social tagging system is an environment where web users using tags to annotate web resources. The system consists of web users, web resources and social tags, and includes a relation set [5]. The model can be characterized by a four-tuple $F=(U,T,R,A)$, in which U is a finite set of web users, T is a finite set of social tags, R is a finite set of web resources, and $A \subseteq U \times T \times R$ is a ternary relation set. The element $a=(u,t,r) \in A$ means that user a use tag t to annotate resource r , the model of social tagging system is shown in Fig.1.

The tag cooccurrence analysis is an important way to reflect the semantic relation of tags. Michlmayr and Cayzer [6] proposed that if two tags are used in combination (cooccur) by a certain user for annotating a certain bookmark, then there was some kind of semantic relationship between them. Szomszor et al. [7] showed that the non-trivial nature of cooccurrence relationships between tags might be considered as a kind of semantic relationship between tags, measured by means of relative cooccurrence between tags, known as Jaccard coefficient. Kipp and Campbell [8] used co-word analysis to elicit patterns from tags used in the social Bookmaking service delicious, they discovered that the number and the using frequency of tags are obeyed a power-law distribution, which means that only a few tags were used very frequently and much more tags were used very infrequently. Begelman, Keller and Smadja [9]explore the use of tag clustering techniques, proposing an algorithm that assign similarity based on cooccurrence, and groups tags based on spectral clustering.

2.2 MapReduce

MapReduce is a programming model [10] and an associated implementation proposed by Google for processing and generating large data sets. It provides an efficient framework for processing large datasets in a parallel manner. The Google File System [11] that underlies MapReduce provides efficient and reliable distributed data storage required for applications involving large datasets.

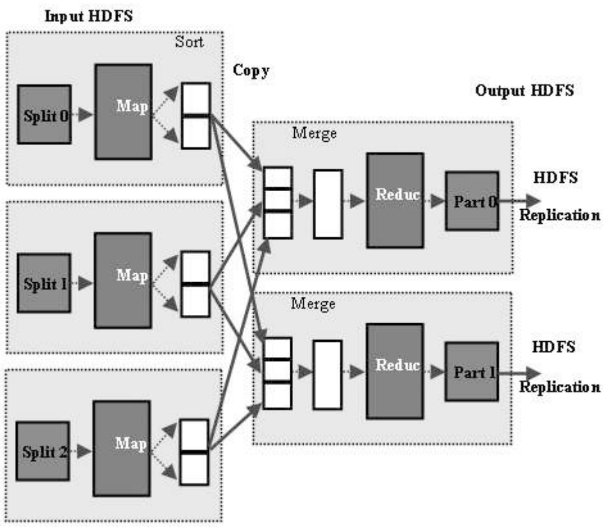

Fig. 2: Model of MapReduce programming.

The basic function of the MapReduce model is to iterate over the input, compute key/value pairs from each part of the input, group all intermediate values by key, then iterate over the resulting groups and finally reduce each group. The model efficiently supports parallelism. The programmer can abstract from the issues of distributed and parallel programming, MapReduce implementation deals with issues such as load balancing, network performance, fault tolerance etc. The Apache Hadoop project [12] is the most popular and widely used open-source implementation of Googles MapReduce writing in java for reliable, scalable, distributed computing. The MapReduce model includes two separate functions: Map and Reduce. Map function takes an input pair and produces a set of intermediate key/value pairs. Reduce function accepts an intermediate key I and a set of values for that key. It merges these values to form a possibly smaller set of values. Typically, just zero or one output value is produced per Reduce invocation. The Hadoop implementation of the MapReduce model is shown in Fig.2, in which HDFS means Hadoop distributed file system [13].

3 Analysis of Tag Cooccurrence Relationship

In the social tagging system, tag cooccurrence has three situations. We will discuss these situations and find out which one is most valuable to reflect the semantic relation of tags. Before analyzing, we will give some symbols to represent some concepts firstly.

$T_{u_n}^T=(t_s|t_s \in T, u_n \in U)$: Which is a tag set of users $u_n, T_{u_n}^T \subset T$.

$T_{r_m}^T = (t_s | t_s \in T, r_m \in R)$: Which is a tag set of resources $r_m, T_{r_m}^T \subset T$.

$T_{u_n,r_m}^T = (t_s | t_s \in T, u_n \in U, r_m \in R)$: Which is a tag set where user u_n use tags to annotate resources $r_m, T_{u_n,r_m}^T \subset T$.

Therefore, there are three cooccurrence forms:

If two tags t_i and t_j in the same tag set $T_{r_m}^T$, we call t_i and t_j co-occur for the same resource.

If two tags t_i and t_j in the same tag set $T_{u_n}^T$, we call t_i and t_j co-occur for the same user.

If two tags t_i and t_j in the same tag set T_{u_n,r_m}^T , we call t_i and t_j cooccur for the same user and the same resource simultaneously.

As for the first situation, two tags appear in the same resources tag set, according to this situation to calculate the frequency occurrence of the two tags in all resources. It shows that these two tags have some semantic relationship, but each resources tag set collected according to many users annotating data. If the users understanding and preference to the same resource are different, they will choose different tags to annotate, so this cooccurrence situation ignores the users annotating behavior and will produce some noise to the extracting of the tags semantic relationship.

As for the second situation, two tags appear in the same users tag set. It shows that these two tags have some semantic relationships about the users expression, but this tag set just reflects the users annotating behaviors, does not reflect the relationship between resources, so this cooccurrence situation ignores the resources relationship and will also produce some noise to the extracting of the tags semantic relationship.

As for the third situation, two tags are used by one user to annotate one resource. There also exist three status, we will using some examples to explain.

Example 1: User u_1 uses tags 'apache' and 'program' to annotate resource r_1 , and user u_2 also uses tags 'apache' and 'program' to annotate resource r_1 , which means that these two tags reflect the different users unifying understanding to the same resources, so these two tags have very strongly semantic relationship.

Example 2: User u_1 uses tags 'apache' and 'program' to annotate resources r_1 and r_2 simultaneously, which means that these two tags reflect the same users unifying understanding to the different resources, which also can reflect that these two resources have same characters, so these two tags have very strongly semantic relationship.

Example 3: User u_1 uses tags 'apache' and 'program' to annotate resource r_1 , and user u_2 uses tags 'apache' and 'program' to annotate resource r_2 , which means that these two tags reflect the different users unifying understanding to the different resources, so these two tags have very strongly semantic relationship.

Therefore, in the three situations of tag cooccurrence, the third situation keeps the ternary relation between user, resource and tag completely, and makes these two tags have the semantic correlation. We will use this situation to extract the tag cooccurrence network based on MapReduce.

4 Extraction Method of Tag Cooccurrence

4.1 Definitions of Tag Cooccurrence

The social tagging system includes three primary entities, according to the analysis of the tag cooccurrence, we find that two tags used by one user to annotate one resource could keep the most strongly semantic relationship; therefore, we will extract this cooccurrence relationship in the huge annotated tag data, and build the cooccurrence network. Firstly, we give some formal definitions.

Definition 1. Tag cooccurrence. D is a given dataset, $D = (U, R, T)$, U is a user set, R is a resource set, and T is a tag set, $d_x = (u_i, r_j, t_k) \in D$ is a item in D,$d \subseteq U \times R \times T$,there are two items $d_x = (u_i, r_j, t_k)$ and $d_y = (u_l, r_m, t_n)$, if $u_i = u_l$,$r_j = r_m$, and $t_k \neq t_n$, then t_k cooccur t_n, using symbol $CO(t_k, t_n)$ to denote, and the cooccurrence time to this two items is 1. Compute all cooccurrence times of t_k and t_n in dataset D, denote as $|CO(t_k, t_n)|$.

Definition 2. Tag cooccurrence intensity. To a given tag pair (t_k, t_n), using $\alpha(t_k, t_n)$ to denote the intensity of this two tags, using $|t_k|$ to denote all the items in dataset D where tag is t_k , the $\alpha(t_k, t_n)$ can be defined as formula (1):

$$\alpha(t_k, t_n) = \frac{|CO(t_k, t_n)|}{(|t_k| + |t_n|) - |CO(t_k, t_n)|} \quad (1)$$

Definition 3. Tag cooccurrence network. Using undirected weighted graph $G(V, E, W)$ to denote, where V is a node set of tags, E is a edge set, $E \subseteq V \times V$, W is a weight function on the set E ,$\forall e(t_k, t_n) \in E$, $W(e(t_k, t_n)) = \alpha(t_k, t_n)$.

4.2 Data Pretreatment

At present, there are many annotated datasets collected from social tagging systems in Internet. The datasets are stored as a text file and those data forms are record items stored random. For example, Olaf Görlitz, Sergej Sizov and Steffen Staab [14] collected the annotated data in Delicious from 2006 to 2007, and formed a dataset, in which there are 532924 users, 17262480 resources, 2481698 tags, and 140126586 record items. The dataset is around 1.1GB in size (compressed). Although the annotated data in social tagging system is massive, the tag cooccurrence relation may be sparser in the whole data space, therefore, it is very hard to calculate any two tags cooccurrence relation in a single computer environment. Even though the dataset is split into several sub-datasets in a distributed environment, extracting tag cooccurrence relation from every sub-dataset also would cause the loss of the tag cooccurrence relation for the random stored record items. To solve these problems, the data should be pre-processed to adapt the MapReduce model. when the data are gathered, we collect all annotated data of one user's and store them in one line of a text file. The form of each line is u_i : $\langle\langle r_1, t_1 t_2 t_5 \ldots t_m\rangle\langle r_2, t_3 t_5 \ldots t_n\rangle \ldots \langle r_i, t_2 t_3 \ldots t_k\rangle \ldots\rangle$.Thus, each line in this text file represents one users annotated behavior. If the dataset is acquired from Internet, we transform it in the same way, in which the data stored can be easily extracted the tag cooccurrence relations, and can be split and merged in the MapReduce model effectively without losing cooccurrence relations .

4.3 Extraction method

The steps of using Hadoop parallel platform to extract social tag cooccurrence relation and calculate its relation are as follow. Firstly, with the help of MapReduce HDFS file system, we assign the huge dataset to n computing nodes, and make each node have one data block and make each data block have k lines data, namely, each block have k users annotated data. Secondly, each node reads k data items line by line and does 'map' operating. Thirdly, use Hadoop platform to merge and sort these 'key/value' pairs. Lastly, send all nodes' 'key/value' pairs to one or more nodes for doing 'reduce' operating. In the 'reduce' stage, all information of the same tag cooccurrence pair will be collected by a 'reduce' node, therefore, the weight of tag cooccurrence pair can be calculated by using formula (1). The model using MapReduce to extract the tag co-occurrence relation is shown in Fig.3.

The algorithm of class MAPPER is shown in table 1. In the stage of 'map', the input of each node is the user id u_i and the users annotated data $d_{r_{j=x\to y}}$. Firstly, 'map' function will create an association container H ; then to each tag pair including tag pair of itself exists in , 'map' function will accumulate appearance frequency in H; at last, output each element from H. The output of 'key' is a two-tuple of tag $d_r j$ pair, and the output of 'value' is the appearance frequency of tag cooccurrence pair.

Table 1: Algorithm of MAPPER class

```
Class MAPPER
1:class MAPPER
2:method MAP(user $u_i$,data $d_{r_{j=x\to y}}$)
3:$H \leftarrow$ new ASSOCIATIVEARRAY
4:for each tag pair $\langle t_k, t_n\rangle \in d_{r_j}$ do
/*where $t_k$ can equal $t_n$ , it means that $k = n$. */
5:$H\{\langle t_k, t_n\rangle\} \leftarrow H\{\langle t_k, t_n\rangle\} + 1$
6:for all tag pair $\langle t_k, t_n\rangle \in H$ do
7:EMIT $(tuple\langle t_k, t_n\rangle$, count $H\{\langle t_k, t_n\rangle\})$
```

The algorithm of class REDUCER is shown in table 2. In the stage of 'reduce', the 'key' received by 'reduce' node is a two-tuple of tag cooccurrence pair, the 'value' is a list of tag cooccurrence frequency. The initial function

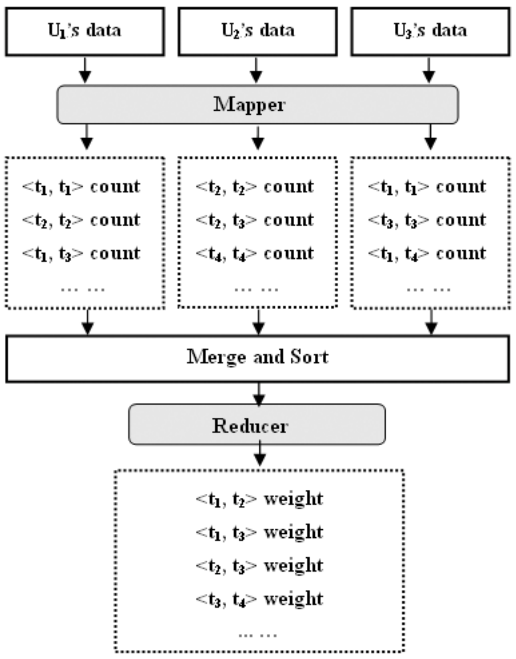

Fig. 3: MapReduce model of extracting tag co-occurrence relation.

Table 2: Algorithm of REDUCER class

```
Class REDUCER
1:class REDUCER
2:method INITIALIZE
3:$p \leftarrow$ new POSTINGARRAY
4:method REDUCE$(tuple\langle t_k, t_n\rangle, count[\langle t_k, t_n\rangle, \ldots])$
5:for each tag pair tuple $\langle t_k, t_n\rangle$ do
6:$P\{\langle t_k, t_n\rangle\} \leftarrow$
    $\frac{\sum_{s=1}^{l} count[\langle t_k, t_n\rangle]_s}{\sum_{e=1}^{f} count[\langle t_k, t_k\rangle]_e + \sum_{g=1}^{h} count[\langle t_n, t_n\rangle]_g - \sum_{s=1}^{l} count[\langle t_k, t_n\rangle]_s}$
/*where $t_k$ can equal $t_n$ , it means that $k = n$. */
7:for each tag pair tuple $\langle t_k, t_n\rangle$ where $t_k$ is not equal $t_n$ do
8:EMIT $(tuple\langle t_k, t_n\rangle$, weight $P\{\langle t_k, t_n\rangle\})$
```

will create a post container P, then the reduce function will calculate each tag cooccurrence pairs weight by using formula (1) and put it into P, and output the elements from P . Here, the same tag pair of itself will not be output.

4.4 Structure of Tag Cooccurrence Network

For the tag cooccurrence relation in the whole tag space is sparse, if the tag cooccurrence relation network is stored as a matrix structure, too much storage space will be

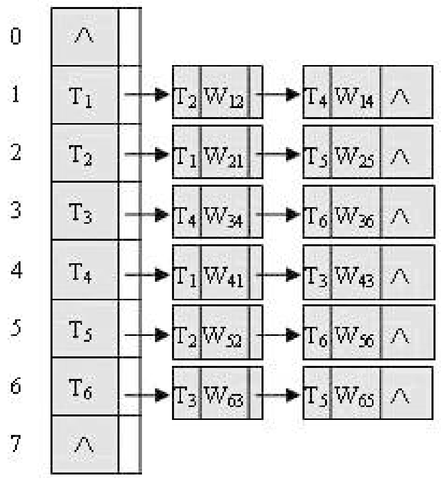

Fig. 4: Hash structure of tag cooccurrence relation network.

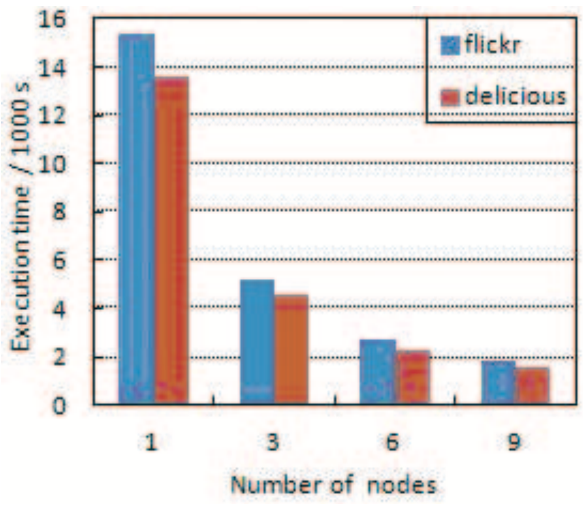

Fig. 5: Scalability experiment on the same data.

waste. To avoid this status, we store the weighted tag cooccurrence relation network in a hash table. Firstly, one tags co-occurrence tags and their weights are stored in a linear list. Then the hash table for storing all tag co-occurrence relations is constructed by assembling each linear list. Using hasp table to store tag co-occurrence relation network can save storage space greatly and can output the tag cooccurrence network conveniently. The hash structure of tag co-occurrence relation network is shown in Fig.4.

5 Experimental Analysis

5.1 Experimental Environment and Data

To verify our methods effectiveness, a cloud computing platform is configurated by using Hadoop 0.20.2. Each computing node uses a PC with the configuration: Intel Xeon 5160 dual-core processor, 4GB RAM; Network environment is 100M-LAN. The Java programming language is used to realize Mapper and Reducer algorithm. The environmental data are two small data sets 'delicious' and 'flickr', which comes from Internet [15]. The data sets are shown in table 3.

Table 3: Experimental Data sets

name	users	tags	resources	tag assignments
delicious	1476	59275	109242	683665
flickr	1476	72671	166423	892378

5.2 Result Analysis

We use scalability to test our method in this paper. The scalability includes two main indicators, one is 'Speedup' and the other is 'Scaleup'. Speedup tests the execution ability to the same data when the computing nodes are increased. Speedup tests the execution ability when the computing nodes and the data are increased at the same rate.

Experiment 1: Select 1, 3, 6 and 9 nodes to compute singly. Each node distributes the data averagely. Take the execution time on one node as the fast time of the single computer environment.$Speedup(p) = \frac{T_1}{T_p}$, where P is the number of nodes, T_1 is the execution time on one node, T_p is the execution time on P nodes [16].

As shown in Fig.5, extraction time of the same scale label cooccurrence decreases with the node increment, and the variation relation is linear. In Fig.6, a good speedup is shown. The results show the method has a good scalability.

Experiment 2: one, three and six nodes are selected to conduct the scalability test. 246 records, 738 records and 1476 records selected from the two preprocessed dataset distributes 1 node, 3 nodes and 6 nodes respectively. $Scaleup(D, p) = \frac{T_{D_1}}{T_{D_p}}$,where D is the data set, p is node number, T_{D_1} is the execution time for D on one node, T_{D_p} is the execution time for D on p nodes [16].

As shown in Fig.7, when node and data scale are in direct proportion, extraction time of the label cooccurrence keep a stable level. There is a good scaleup in Fig.8. So the test result shows that the method have a good scalability.

6 Conclusions

There is a main task for the research on the social tagging system, which is extracting the tag cooccurrence

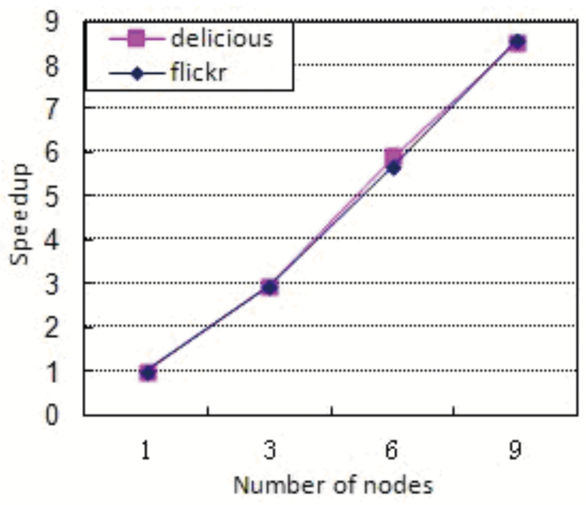

Fig. 6: Speedup.

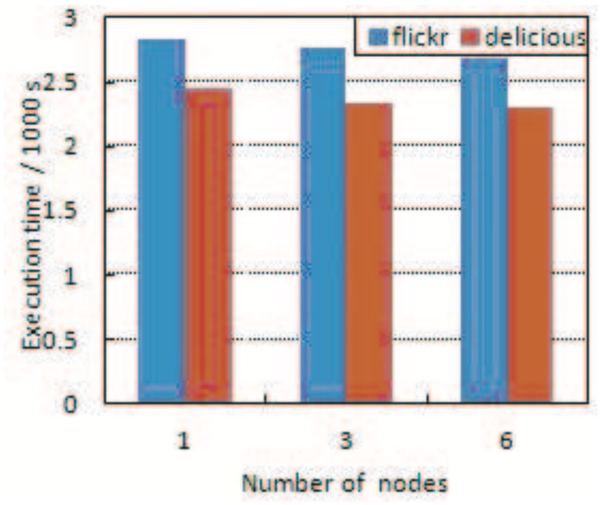

Fig. 7: Scalability experiment on the same increasing rate.

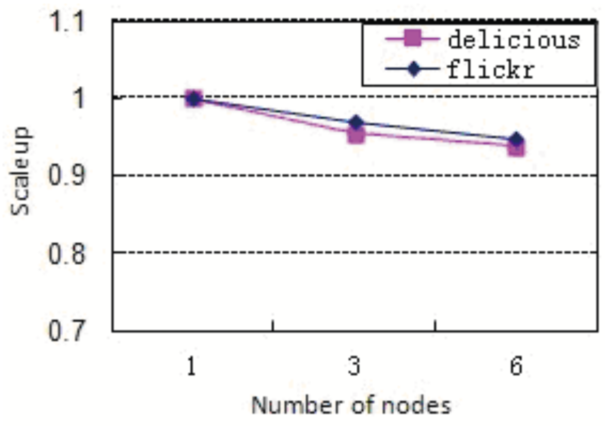

Fig. 8: Scaleup.

relationship from the massive annotating data. There is a simpler and more efficient method for dealing with this huge annotating data, which is a parallel computing method. Therefore, we proposed a novel parallel method to extract the tag cooccurrence relation in this paper. The work of this paper is very useful for the application extending of the social tagging system. In the future work, we will design a parallel clustering algorithm to cluster the tags from the tag cooccurrence realtionship, so we can quickly find the context environment, and provide a working basis for the tag recommendation.

Acknowledgement

This work was supported by the National Natural Science Foundation of China (Grant No. 51001004), the key project of Educational department of Anhui Province in China (Grant No. kj2012A087) and the Huainan Science and Technology Program of China (Grant No. 2012A00913). The authors are grateful to the anonymous referee for a careful checking of the details and for helpful comments that improved this paper.

References

[1] Thomas Vander Wal, Folksonomy (2004), http://www.vanderwal.net/folkso-nomy.html.
[2] Clay Shirky, Folksonomy (2004), http://many.corante.com/archives/2004/08/25/folksonom-y.php.
[3] Adam Mathes, FolksonomiesCcooperative classification and communication through shared metadata(2005). http://www.adammathes.com/academic/computer-mediated communication/folksonomie-s.html.
[4] Dean J, Ghemawat S, MapReduce: Simplified data processing on large clusters. Communications of the ACM, **51**, 107-113(2008).
[5] Hotho A, Jäschkes R, Schmitz C, Stumme G, Information retrieval in folksonomies: search and ranking, The Semantic web: research and applications, Springer, Heidelberg, **4011**, 411-426(2006).
[6] Michlmayr E, Cayzer S, Learning user profiles from tagging data and leveraging them for personalized information access, Proceedings of the Workshop on Tagging and Metadata for Social Information Organization, 16th International World Wide Web Conference, (2007).
[7] Martin Szomszor, Ciro Cattuto, Harith Alani, Kieron OHara, Andrea Baldassarri, Vittorio Loreto, and Vito D. P. Servedio, Folksonomies, the semantic web, and movie recommendation, 4th European Semantic Web Conference, Bridging the Gap between Semantic Web and Web 2.0, Innsbruck, Austria, 71-84 (2007).

[8] Margaret E. I. Kipp, D. Grant Campbell, Patterns and Inconsistencies in Collaborative Tagging Systems: An Examination of Tagging Practices, Proceedings of the American Society for Information Science and Technology, **43**, 1-18 (2006).
[9] G. Begelman, P. Keller, and F. Smadja. Automated Tag Clustering: Improving search and exploration in the tag space, Proceedings of the Collaborative Web TaggingWorkshop at WWW, **6**, (2006).
[10] R. Lämmel. Googles mapreduce programming model-revisited, Sci. Comput. Program, **70**, 1-30 (2008).
[11] S. Ghemawat, H. Gobioff, S. Leung. The google file system, SIGOPS Oper. Syst. Rev., **37**, 29-43 (2003).
[12] Apache Hadoop, Available: http://hadoop.apache.org/.
[13] White, Hadoop: The Definitive Guide, first ed., O Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA, (2009).
[14] Olaf Görlitz, Sergej Sizov, Steffen Staab. PINTS: Peer-to-Peer Infrastructure for Tagging Systems. In: Proceedings of the Seventh International Workshop on Peer-to-Peer Systems (IPTPS). Feb, Tampa Bay, USA, (2008).
[15] http://wis.ewi.tudelft.nl/icwe2011/um/, (2011).
[16] X. Xu, J. Jäger, H. P. Kriegel, A fast parallel clustering algorithm for large spatial databases, Data Mining and Knowledge Discovery, **3**, 263-290 (1999).

Xiangqian Wang received the MS degree in computer application technology from Anhui University of Science and Technology in 2006. He is currently a associate professor in Anhui University of Science and Technology, PHD candidate. His research interests are in the areas of management science and engineering, coal mining technology, compute application.

Huizong Li received the MS degree in computer application technology from Anhui University of Science and Technology in 2006. He is currently an associate professor in Anhui University of Science and Technology, PHD candidate. His research interests are in the areas of management science and engineering, knowledge management, social tagging system.

Yerong He received the MS degree in management science and engineering from Anhui University of Science and Technology in 2007. She is currently an associate professor in Huainan Normal University. Her research interests are risk assessment and management.