Parallelizing Label Propagation for Overlapping Community Detection

Naiyue Chen
Key Laboratory of Communication and Information Systems, Beijing Municipal Commission of Education
Beijing jiaotong University
Beijing, China
13111007@bjtu.edu.cn

Yun Liu
Key Laboratory of Communication and Information Systems, Beijing Municipal Commission of Education
Beijing jiaotong University
Beijing, China
liuyun@bjtu.edu.cn

Junjun Cheng
China Information Technology Security Evaluation Center
Beijing, China
chengjj@itsec.gov.cn

Qing Liu
China Electric Power Construction Limited by Share Ltd
Beijing, China
liuqing@powerchina.cn

Abstract*—Community detection is one of the most important ways that reflect the structure and mechanism beneath the social network. The overlapping communities are more in line with the reality of social network. In the society, the phenomenon of some members shared membership of different communities reflects as overlapping communities in the network. Facing big data network, it is a challenging and computationally complex problem to detect overlapping communities. In this paper, we proposed highly scalable variants of a community detection algorithm with parallelized called Label Propagation with nodes Confidence (PLPAC). We introduced MapReduce to parallelize the algorithm to process the big data and guarantee the efficient of community detection. We implemented the algorithm on real network and artificial network to evaluate the accuracy and speedup of the proposed algorithm. Experiments results on many test datasets illustrated that the improved label propagation method outperforms some existing methods in terms of accuracy and time efficiency.

***Keywords*—community detection; label propagation; parallel computation**

I. INTRODUCTION

Social network has become an indispensable part of present society. Community represents significant property of real word social network as it reflects the relationship between the users. Analyzing network structure and detecting community of people also play an important part in research on social network. Detecting network community structure is of very important theoretical significance and practical value for analyzing network topology structure, network function and predicting network behavior, and has been widely used in terrorist organizations, organizational structure management and some other fields.

Many community detection algorithms have been proposed in the literatures to identity complex community structures in social network, such as the Girvan Newman algorithm [1], some other algorithms based on label propagation algorithm [2] and optimization algorithm. After improvement and optimization, these algorithms further reduce time complexity degrees. However, facing big scale social network, the calculation time is still too large to detect community efficiently. The data volume produced by social network is growing with an enormous rate, such as the Microblogging [3]. Therefore, the existing traditional community mining algorithms have great limitations including low computing power, high computational time, and the bad division result of the community with high quality.

Community detection is similar to traditional clustering or graph partitioning problems. Thus, several effective clustering or graph partitioning algorithms have been applied in community detection. The Kernighan–Lin algorithm aims to minimize the difference between intra-edges and inter-edges to detect communities [5]. However, these early algorithms cannot detect large network efficiently because of their high time complexity. A promising algorithm, called the label propagation algorithm (LPA), was proposed recently [6]. This algorithm is particularly suitable for large social networks with complex communities because of various reasons [7]. Although LPA is suitable for large network, it cannot find overlapping communities and the division results are of highly randomness. Then COPRA extends the LPA and becomes another classical method to detect overlapping communities [8]. In addition, several other algorithms also have been designed to overcome the limitations of the LPA algorithm. For example, SLPA [8], and BMLPA [9] alleviate the problem of monster communities by introducing an extra parameter to control the number of labels that a vertex can hold.

Facing big data analysis, the parallel processing method arises at the historic moment. The MapReduce [10, 11] can be used to achieve distributed clustering, and has a good scalability and fault tolerance to satisfy the needs of the rapid growth of data. However, in the distributed framework, the clustering algorithm must be operated in a distributed way. Many existing algorithms are not distributed and cannot be easily represented as a single MapReduce process.

In this paper, based on the label propagation classical model, we not only improve the classical method, but also accommodate the algorithm to parallelize in big data network. This paper proposes a fast way to adaptive big and confuse network, we utilize MapReduce to distributed compute the labels of nodes in the network. We use synchronization and no data replication. In the LPA, it adopts the asynchronization update because of oscillation of labels. If the network has very large-scale and real time update, the LPA is not suitable by asynchronization update way. Meanwhile it is not suitable for distribute dynamic complex network. To avoid the random of LPA we propose a new processing of label propagation with considering the relationship between the nodes. In this paper, the parallel label propagation can combine asynchronization and synchronization update way by using MapReduce. It can save time cost and avoid the label oscillation.

The organizational structure of the rest of the paper is as follows: section 2 introduces related work of the previous studies about label propagation algorithm. Section 3 introduces the preliminary knowledge about parallelized manage. Section 4 proposes a parallelize label propagation algorithm to detect the network structure. Section 5 shows the experiment and the comparing results with other algorithms. Finally, in section 6 we discuss the conclusions and the future work.

II. RELATED WORK

Community detection by label propagation belongs to the class of local move heuristics. In previous work, label propagation algorithm is the most common method to detect the community structure caused it has approximate linear time complexity [12]. However, LPA just can find non-overlapping community, so as an extend in COPRA [13], each node updates its labels and the belonging coefficients averaged out from the coefficients of all its neighbors in a synchronous manner. SLPA is a general speaker–listener based information propagation process [14]. It spreads label information between nodes according to pairwise interaction rules. In the SLPA, each node has a memory space to store the received information. The probability of observing a label in the memory of a node is perceived as the membership strength [15]. Compared with the existing label propagation methods, our algorithm introduces the concept of confidence to denote the importance of each neighbor in the label updating process.

Hadoop is a distributed system infrastructure, the distributed storage and distributed computing is the core of distributed system. The most fundamental objective of distributed system design is split the large-scale task into many small tasks [16], and then assign the small tasks to each node with parallel processing, finally generated the results from each processors as the final result. MapReduce is the mainly programming model of implementation the Hadoop architecture. MapReduce [16], as a parallel programming model, is good at dealing with large data and large calculation. The simple MapReduce has three parts: Map function, Reduce function and the main function. If make traditional community detection algorithm parallel with MapReduce programming model and make a good use of cluster computing advantage to handle big users' data, the execution time of the algorithm will be shortening [17].

III. PARALLELIZING LABEL PROPAGATION

Label propagation algorithm has approximate linear time complexity and is very suitable for large network community detecting. As the number of users on social networking now reached hundreds of millions, using the classical algorithm is of high computing complex. If we employ the distributed computing algorithm (i.e., the computing process of the algorithm is distributed) to process the data, the execution time of community detection algorithm is much more shortened, and the efficiency is also improved significantly. MapReduce, as one of the mainstream parallel computing programming models, is very suitable for processing large-scale data sets. Therefore, it is one of the effective methods to solve the problem of the efficiency in community detecting algorithm.

A. Data preprocessing

Paralleling community detection algorithm means make data sets distribute into each machine averagely, the algorithm is calculated on each machine, the calculation process of machine independent, input data sets are also independent and eventually computed on each machine results together to get the results. In the previous discussion, using Hadoop of MapReduce programming model algorithm parallelization is better choice, so here we will be utilized Hadoop to parallel the data sets.

As discussion of related work, in the process of synchronous updating, there is a potential oscillation problem, which leads hard to the convergence of the algorithm. In an asynchronous update, if a node updates its label value during an iteration, the value must be immediately fed back to all the nodes in the network. This increases the high coupling between the data sets and is contrary to the principles and design of the MapReduce original intention.

As each step of MapReduce is the process of Map and Reduce, if we design the MapReduce algorithm in accordance with this situation, the process of Map calculates the node update value and the process of Reduce is feeding back the update value to all neighbor nodes. As a label value update is required to feed back to all of the same neighbor nodes, and it need a Reduce process. If the network contains enormous users and at each node asynchronous label propagation need process of Reduce, it will need so many Reduce process and reduce the efficiency of the algorithm.

In order to solve this problem, we first need to reduce the coupling between the input data sets. In this paper we will build a data set called NodeModel which contains a node and its neighbor nodes. If we consider the network as consist of NodeModel, not just nodes and edges. The input set of

Mapreduce is the NodeModel data sets. We can distribute the input sets averagely to each machine to compute the label value at the same iteration step. As the caused oscillation and asynchronous update cost lots of time, we combine synchronous update with asynchronous update to evolve the update method. In this paper, the network was divided into n subsets equally. We utilize multi thread to process a part of the network, feedback the results to the rest of the subsets. This process will be iterated until the whole network completed. This method solve the oscillation caused by synchronous and the problem of high coupling of dataset caused by asynchronous. The structure of large social networking is generally sparse [18], in sparse networks, most of the nodes are not connected by each other, but the majority of nodes can be reached another node through a small number of steps. Therefore, the above method combing synchronous with asynchronous update is feasible.

In the step of Map, it managed a part of dataset, the result feedback to others datasets in the processing of Reduce. Then the output of the Reduce will be as the input to the next Map processing. This processing will be repeat until the whole datasets have been computed and it means one iteration execution is completed. The iteration will be continuing until the label value of user nodes reaches the required convergence condition. The figure 1 shows the structure of the processing by MapReduce.

B. Label propagation with the confidence between nodes

In the processing of label propagation, the most important link is the computing of label value. In classical label propagation, the node chose the maximum of the label value, if there are same label value of its neighbors, it will pick up a label randomly. In this paper, we consider the relationship between nodes as an important indicator to compute the label value. In the network, the nodes all in the same community have strong relationship and influence to each other. In the social network each node corresponds to a real user, in the real society, the influence between users are different. As different friend has diverse influence to their friends, so we should consider the confidence between the nodes by the processing of the choice of label propagation.

Definition 1 The confidence of the node v to its neighbor node u. Its neighbors are defined as

$$\theta_u(v) = \frac{sim(u,v)}{\sum_{i \in N(u)} sim(u,i)} \tag{1}$$

Where N(u) represents the set of neighbors of node u and sim(u, v) represent the similarity between nodes u and v. Here, we use the Jaccard similarity function [19]

$$sim(i,j) = \frac{|\Gamma(i) \cap \Gamma(j)|}{|\Gamma(i) \cup \Gamma(j)|} \tag{2}$$

$$\Gamma(x) = \mathrm{N}(x) \cup \{x\} \tag{3}$$

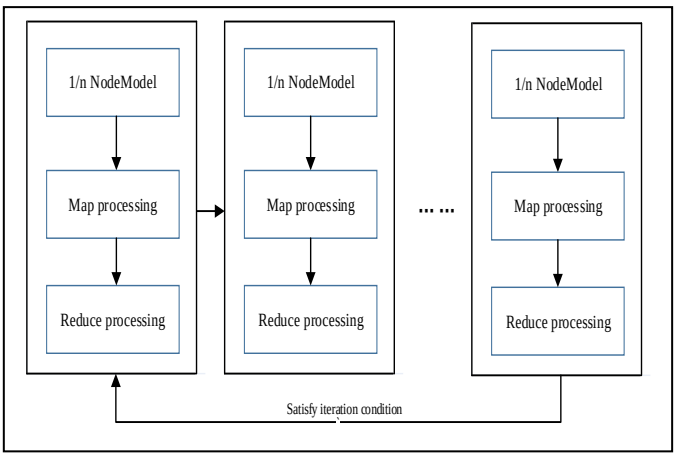

Fig. 1. the structure of the processing by MapReduce

For a pair of nodes, the confidence measures the intensity of their connection. For example, as shown in Fig.2, the confidences of node 2 to its neighbors are $\theta_1(2)=0.5$, $\theta_3(2)=0.35$, $\theta_4(2)=0.198$. It can be seen that the relationship between node 2 and its neighbor node 1 is the strongest, and the relationship between node 2 and its neighbor node 3 is stronger than that of node 4.

This paper proposes the algorithm called LPA-C to consider the conference between users to detect the overlapping community. As each node has a sequence of its neighbor node ID, we create a sequence corresponding to the confidence between the node and its neighbor node. When the labels propagate, the labels were sent with the confidence value of this neighbor node with target node. We add the confidence value from neighbor nodes which owned same label sent to target node. $\varepsilon_l(v)$ means whether the node v has the label l. If it has this label, the value of $\varepsilon_l(v)$ is 1, otherwise, the value is 0.

$$w_l(i) = \sum_{v \in N(i)} \theta_v(i) * \varepsilon_l(v) \tag{4}$$

Then we introduce inflation operation φ_{in} on conference to control the overlapping rate, within which is the parameter taking real-number values. After applying φ_{in} on the labels of node i, the belonging coefficient rises to the in-th power. The inflation operator φ_{in} is defined as

$$\varphi_{in} w_i(i) = \frac{w_i(i)^{in}}{\sum_{i \in N(i)} w_i(i)^{in}} \tag{5}$$

The inflation operation φ_{in} is also a normalized method and can be considered as the label weight to the node. If the label weight is bigger than the threshold, we will keep the label to the node. In this paper the threshold is set as the reciprocal of the node degree. Based on the above

definitions, the label updating process is described in the following. First, compute the confidence value of each node

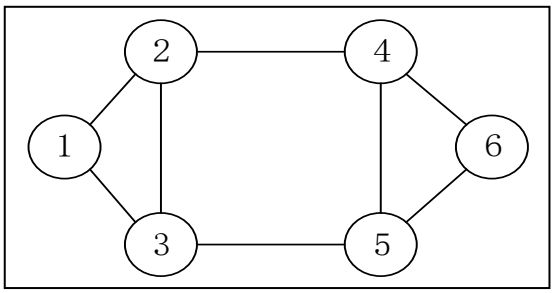

Fig. 2. The simple network

with its neighbors. Second, for any target node u, each neighbor of u sends its labels and the corresponding confidence value to u. Third, after u receives the labels and the confidence value from its neighbors, we add each confidence value by its corresponding confidence to output the new confidence value. Fourth, we normalize each new confidence value from its neighbors via Eq. (5), and choose the label which confidence value is bigger than the threshold.

C. Parallel community detection with label propagation

The label propagation algorithm is a sequential linear time algorithm for detecting communities. In Hadoop based parallelized LPAC, we split the network into n partitions of nodes to be processed on p processors. Each processor gets its allocation of nodes that are contains user-id and recreates network induced by local node by creating duplicates of nodes that are allocated to other processors but have an edge whose other node is the local one. This paper improved the way of label update method.

In the Map processing, the Map function is used to update user ID's new label value. Map function processes the prepared data in each row in the iteration phase. It deals with 1/n data by the main functions and configuration of the design until all data are processed. In this processing, Map function will compute the label weight by formula (5), and hold the label which the label weight is bigger than threshold. And then assign the new labels to the label variable of the user ID and save the record order to be called in Reduce processing.

In the Reduce processing, the Reduce function is used to update the label value of users' neighbors. The input of the Reduce function is the output of the Map function. It updates the label value based on the user ID and its new label value. The format of output is <key (user ID), value (new and old label value, neighbor id, the label value of neighbor)>. If the ratio between the number of the nodes that keep the same new and old labels and the total number of all node meets the specified value, the iteration ends, otherwise the iteration continues. This processing of parallelize label propagation with node confidence called PLPAC as showed in Fig.3.

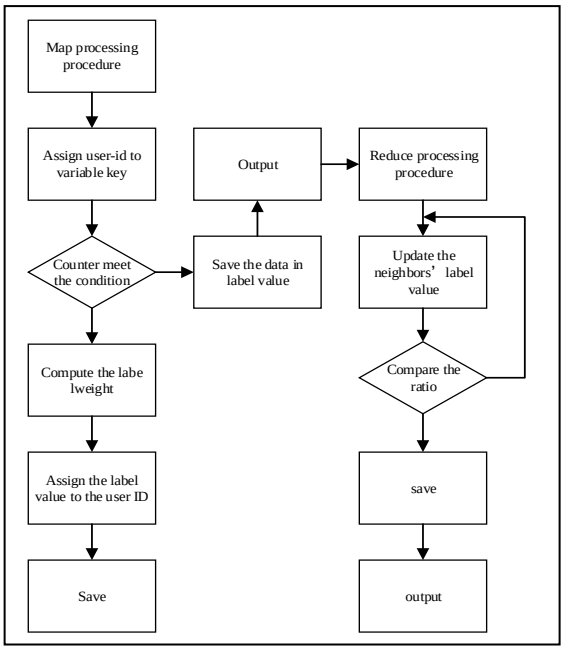

Fig. 3. The processing of PLPAC

IV. PFORMANCE EVALUATION OF THE PARALLELIZING LABEL PROPAGATION

In this section, we first describe the experimental environment and simulation dataset. Then we describe and analysis the experiments that we performed using parallel LPA-C.

A. Framework and setting

The language of choice for all implementations is Java according to the JDK 1.6 standard, allowing us to use object-oriented and functional programming concepts while also compiling to native code.

The Hadoop cluster environment is used in this experiment which consists of 10 machines, a typical master slave mode, (Master-Slaves) structure. The cluster consists of a master node (Master) and four slave nodes (Slave). In the master-slave structure, the main nodes are generally responsible for cluster management, task scheduling and load balancing, and the slave node performs calculation and storage tasks from the main node. For representative experiments we average quality and speed values over multiple runs in order to compensate for fluctuations. Table 1 provides information on the multicore platform used for all experiments.

TABLE 1

Environmental category	**Describe**
Hardware	Intel (R) Xeon () CPU (R), 4G memory
CPU	Intel(R) Xeon(R) E5-2620v3 @ 2.40GHz, 64 threads
Development environment	Eclipse 32, 64bit java version 1.6.0_02

B. *Datasets*

This paper performed experiments on a variety of graphs from different categories of real-world and synthetic data sets. We can use NMI [20] (Normalized Mutual Information) to measure the performance of parallel LPA-C with other algorithms for the known community structure network. For some real networks, there is no known community structure at present, so this paper will use the EQ function to evaluate the results.

$$NMI(X,Y)=\frac{2I(X,Y)}{H(X)+H(Y)}$$
$$I(X,Y)=\sum_{y\in Y}\sum_{x\in X}p(x,y)\log\left(\frac{p(x,y)}{p(x)p(y)}\right) \quad (6)$$
$$H(X)=-\sum_{i=1}^{n}p(x_i)\log p(x_i)$$

In the real network part, we use some classical dataset to test and verify the performance of this algorithm, and compare with other algorithms. In the artificial network, we can experiment the efficiency of the proposed algorithm. The details of the experimental data sets were showed in Table 2.

TABLE 2

Network	Vertices	Edges	Description
Karate [21]	34	78	Zachary's karate club
Dolphins [22]	62	159	Dolphins social network
Football [23]	115	613	Football American College football
Netsci [24]	1589	2742	Network scientists
Artificial network [25]	100k to 5M	Mu=0.1 to 0.8	LFR

C. *Experments*

The experimental results for artificial network with varying sizes are presented in Fig. 4. As shown in Fig. 4(a), the algorithm accuracy of the LPA-C is consistently better than other algorithms on the artificial network. This result demonstrates the effectiveness of node confidence value and label selection through the NMI values. We can find that the performance of COPRA and PCOPRA is not identical and neither is the performance of SLPA and PSLPA. There is different label propagation update way between the classical algorithm (like, COPRA, SLPA) and the parallelized algorithms. The synchronization mechanism is necessary for designing the parallel steps of the algorithms. On the contrary, COPRA and SLPA update the labels asynchronously.

In the Fig. 4(b) shows how running time varies with increasing network scale. Clearly, the total running time includes the time spent on communication between processors and time spent on execution of the algorithm itself. It is obvious that the time cost of all the algorithms increases nearly linearly with network size. When the number of network nodes in thousands of counting, the speedup of the other algorithms caused by parallel computation is not evident. It possible caused by the time spent on data processing is comparable with the time spent on cluster administration and communication. However, when the number of network nodes in millions of counting, the speedup of the parallelized algorithms shows the advantage than classical algorithms. Parallel computation becomes remarkable when network scale increases beyond the capability of a single-machine algorithm. SLPA and PSLPA run faster than PGLPA and PCOPRA because the speak and listen strategy is simpler than the label updating strategies used in PLPAC. PSLPA exhibits better scalability than SLPA, which is largely due to the parallel speak and listen scheme. Although the run time of PLPAC is higher than PSLPA, the high NMI value shows the algorithm this paper proposed can detect the better communities.

The experimental results for the real networks are presented in Fig. 5. The Fig. 5 shows the performances of the algorithms on the real networks are considerably different from those on the artificial networks. The networks known the structure can utilize NMI to show the performance of community detection by those algorithms. The NMI value is higher, the structure divided is close to the real structure.

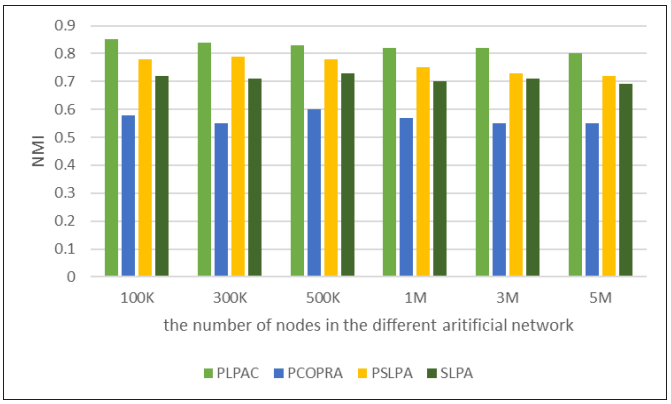

Fig. 4(a). The NMI value in artificial network

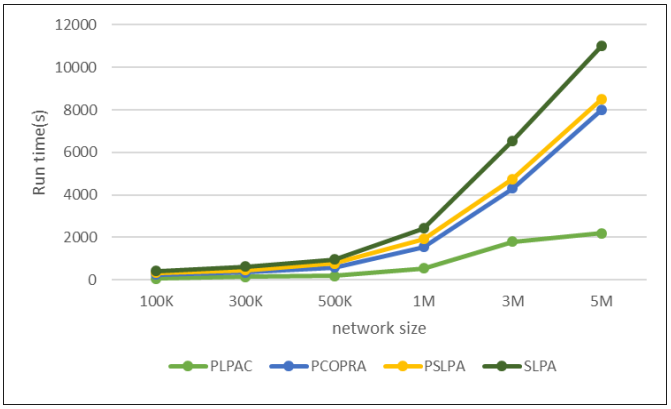

Fig. 4(b). The run time in artificial network

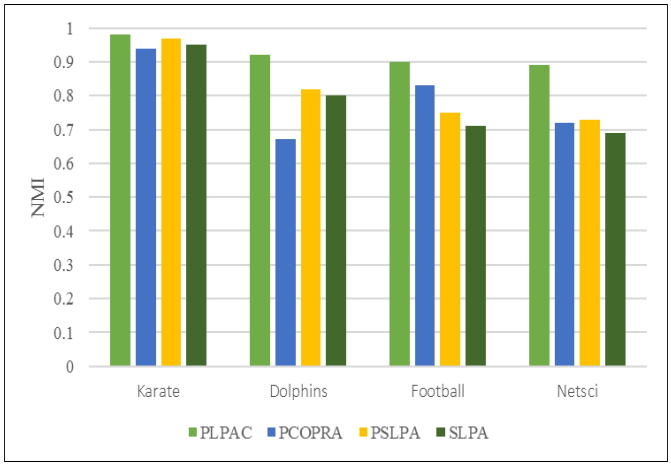

Fig. 5. The NMI value in real network

At the end of each run, we calculated the total execution time and speedup using formula shown in (7), efficiency according to (8).

$$Speedup = \frac{T_1}{T_n} \tag{7}$$

Where, T_1 means the running time on the single machine, the T_n means the running time on the cluster.

$$Efficiency = \frac{Speedup}{p} \tag{8}$$

Experiments are conducted on the 1 M network with a varying number of machines to evaluate the effect of cluster scale on the performance. Fig. 6 shows that a boost on running speed caused by adding machines to the cluster is evident. As the number of processors increase, the growth rate of speedup is decay. Because the time cost by communication and management among the machines will rise with more machines. Therefore, we should balance the number of the processors to detect the accurate community structure in the shortest time.

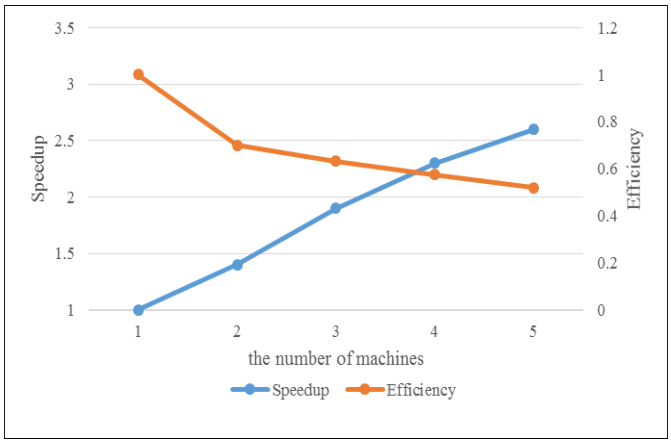

Fig. 6. Speedup and efficiency for a network

V. Conclusion and Future Work

In this paper, we proposed a paralleling label propagation algorithm with node confidence to detect communities. In addition, we evaluated the performance of a multi-threaded parallel implementation of label propagation algorithm and showed that using modern multiprocessor can significantly reduce the time required to analyze the structure of different networks and output communities. We found that with the rise of the numbers of the processors, the rate of speedup reduces slowly. This can be explained that there is more and more communication time spent on the processors should be considered. Our parallelized algorithm PLPAC implementation was proven that it can detect the communities in big data network with high accuracy. Compared with other algorithms, simulation result shows that our algorithm can correctly identify overlapping community structures from real data, and the improved label propagation with node confidence is very effective. Besides, the speedups on various datasets and different numbers of machines are satisfactory.

In our future work, we plan to raise more number of the processors and evaluate the experimental performance. In addition, we will explore other parallel programming paradigms to compare their performance with our parallel approach.

Acknowledgment

This work has been supported by the Fundamental Research Funds for the Central Universities 2016YJS029 and the National Natural Science Foundation of China under Grant 61401015, 61271408. Academic Discipline and Postgraduate Education Project of Beijing Municipal Commission of Education

References

[1] *Girvan M, Newman M E J. Community structure in social and biological networks[J]. Proceedings of the National Academy of Sciences of the United States of America, 2002,99(12): 7821-7826*

[2] *Raghavan U N, Albert R, Kumara S. Near linear time algorithm to detect community structures in large-scale networks[J].Physical Review E, 2007, 76(4): 046106*

[3] *Pan X, Yang J, Qiu X. A multi-label model to predict undisclosed attributes in microblogging[C]// International Conference on Behavioral, Economic and Socio-Cultural Computing. IEEE, 2015.*

[4] *S. Fortunato, Community detection in graphs, Phys. Rep. 486 (2010) 75–174.*

[5] *B.W. Kernighan, S. Lin, An efficient heuristic procedure for partitioning graphs,Bell. Syst. Tech. J. 49 (1970) 291–407.*

[6] *U. Raghavan, R. Albert, S. Kumara, Near linear time algorithm to detect community structures in large-scale networks, Phys. Rev. E. 76 (2007) 12.*

[7] *Qishan Zhang, Qirong Qiu, Wenzhong Guo, et al. A social community detection algorithm based on parallel grey label propagation[J]. Computer Networks.*

[8] *S. Gregory, Finding overlapping communities in networks by label propagation, New J. Phys. 12 (2010) 104018*

[9] *Z.H. Wu, Y.F. Lin, S. Gregory, H.Y. Wan, S.F. Tian, Balanced multi-label propagation for overlapping community detection in social networks, J. Comput. Sci.Technol. 27 (2012) 468–479.*

[10] *Dean J, Ghemawat S. MapReduce: Simplified Data Processing on Large Cluster[J]. Communications of the ACM, 2005, 51(1): 107-114.*

[11] *Lee K, Lee Y, Choi H. Parallel Data Processing with Map-Reduce: A Survey[J]. ACM SIGMOD Record, 2011, 40(4): 11-20.*

[12] *Ugander J, Backstrom L. Balanced label propagation for partitioning massive graphs[C]// ACM International Conference on Web Search and Data Mining. 2014:507-516.*

[13] *Gregory S. Finding overlapping communities in networks by label propagation[J]. New Journal of Physics, 2009, 12(10):2011-2024.*

[14] *Xie J R, Szymanski B K and Liu X M 2011 Proceedings of the 11th International Conference on Data Mining Workshops, December 11–14,2011 Canada, pp. 444–449*

[15] *Sun He-Li, Huang Jian-Bin, Tian Yong-Qiang, et al. Detecting overlapping communities in networks via dominant label propagation[J]. Chinese Physics B, 2015, 24(1):551-559.*

[16] *Dean J, Ghemawat S. MapReduce: Simplified Data Processing on Large Clusters.[J]. Communications of the Acm, 2008, 51(1):107-114.*

[17] *Dean J, Ghemawat S. MapReduce: A Flexible Data Processing Tool[J]. Communications of the Acm, 2010, 54(1):72-77.*

[18] A Lancichinetti,S Fortunato.Community detection algorithms:a comparative analysis[J].Physical ReviewE,2009

[19] *Rousseau R. Jaccard similarity leads to the Marczewski-Steinhaus topology for information retrieval[J]. Information Processing & Management, 1998, 44(1):87-94.*

[20] *Rousseau R. Jaccard similarity leads to the Marczewski-Steinhaus topology for information retrieval[J]. Information Processing & Management, 1998, 44(1):87-94.*

[21] *Zachary W W. An information flow model for conflict and fission in small groups[J]. Journal of Anthropological Research, 1997, 44(4): 452-474.*

[22] *M. Girvan and M.E.J. Newman. Community structure in social and biological networks. Proceedings of the National Academy of Sciences of the United States of America, 99(12):7821–7826, 2002.*

[23] *Lusseau D, Boisseau S K, et al. The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations. Behavioral Ecology and Sociobiology, 2004, 54: 496-405.*

[24] *Lusseau D, Boisseau S K, et al. The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations. Behavioral Ecology and Sociobiology, 2004, 54: 496-405.*

[25] *L. Danon, A. Diaz-Guilera, J. Duch, and A. Arenas. Comparing community structure identification. Journal of Statistical Mechanics: Theory and Experiment,2005:P09008, 2005.*