HLPA: A Hybrid Label Propagation Algorithm to Find Communities in Large-scale Networks

Ting Wang*, Xu Qian† and Xiaomeng Wang‡
School of Mechanical Electronic & Information Engineering,
China University of Mining & Technology, Beijing
Beijing, China
Email: *wangting33184@gmail.com, †xuqian@cumtb.edu.cn, ‡xiao_meng_wang@163.com

***Abstract*—Fast detecting communities is challenging in large-scale real-world social networks and an important task in many scientific domains, such as Complex Networks and Social Network Analysis. In this paper, we propose a Hybrid Label Propagation Algorithm (HLPA) for finding communities on large-scale real-world social networks. And we conduct our experiments on real-world social networks datasets and get meaningful community results. Our method can get detection results on large-scale networks significantly fast, due to the following two benefits. The first is that our near linear algorithm HLPA is using a novel hybrid updating scheme, label decaying strategy and different initialization methods on different networks to improve the quality and scalability for detecting communities. And the second is that this is the first attempt implementation at community detection on the lightning-fast cluster computing framework Dpark, which is a Python version of Spark. Through experiment, we compare our algorithm with the state-of-art algorithms, and have confirmed our algorithms' superiority and universality for working on unweighted overlapping community detection of large-scale real-world social networks.**

Keywords
Community Detection, Label Propagation Algorithm, Dpark, HLPA

I. Introduction

Community detection (CD) of real-world social networks has become one of the hottest research fields in Social Network Analysis (SNA) and Complex Networks in recent years, especially analyzing large-scale social network data is getting more complicated and challenging. A general definition of community is that a network divides naturally into clusters of nodes which are densely connected internally and sparsely connected otherwise. Communities in social networks can provide insights into common attributes or interests among people which makes them different from other communities. There are two phases of CD algorithms. One is for non-overlapping or disjoint community detection in which one node can only belong to one community, such as Girvan-Newman algorithm [1]. In reality, network communities are usually overlapping with each other since one node can belong to two or more communities. It leads to the second phase of CD algorithms, overlapping community detection algorithms.

There are a number of local based optimization methods utilizing seed expansion to grow natural communities [2]. LFM [3]recursively expands a randomly selected seed node to a community until the fitness function reaches a local maximum. However, experiments are not repeatable due to randomly node selection. Clique based methods, such as Clique Percolation Method (CPM) was proposed by Palla [4]. It builds up communities as percolation maximum union of k-cliques by finding adjacent cliques that share k-1 nodes. But this kind of method needs to preset value of k which varies in different kind of networks, but the most appropriate value is 4 or 5 [5]. However, due to the clique requirement and the sparseness of real networks, the communities discovered by CFinder[1], an implementation of CPM, are usually of low quality [2]. Apart from that, Lee [6] proposes Greedy Clique Expansion (GCE), a method based on the greedy and iterative expansion of all "seed" cliques of sizes 3 or 4. This method is not time effective due to the long running time inherited from the clique finding process. Our previous work Parallel Hybrid Seed Expansion (PHSE) [7], is a parallel algorithm to get nature community through the local optimization of the fitmess function with a hybrid seed expansion strategy, and can accurately detect highly overlapping communities. Besides, these two methods OSLOM[8] and Infomap [9] show much promise as accurate.

Modularity, defined by Newman [1][10], is the most popular and widely used metric to evaluate community structure of community detection results. There are various modularity based community detection algorithms which use modularity as a key fitness indicator [11]. However, modularity optimization has been proved to be resolution limit [12], especially for larger systems and smaller communities.

Label propagation algorithm (LPA) [13] is an extremely fast community detection method and is widely used in large scale networks. However, each node can only belong to one community in LPA. COPRA [14] is an extension of LPA, it can find overlapping communities. But it allows each node belonged to v communities at the most and the parameter v is vertex-independent. Xie *et al.* propose a Speaker-listener Label Propagation Algorithm (SLPA) [15][16], which uses memory list to store labels from each iteration, so it can detect the overlapping communities.

Scalability has become an important and desirable feature of community detection methods due to increasing populations of social services. [17] summarize the known implementations

[1]http://www.cfinder.org/

include: LPA using Hadoop MapReduce, SLPA using MPI, the Louvain method using Apache Giraph [18] and many others.

DPark[2] is a Python version implementation of Spark, is a Mesos based cluster computing framework similar to MapReduce but more flexible and supporting iterative computation. Dpark revolves around the concept of a resilient distributed dataset (RDD), which is a fault-tolerant collection of elements that can be operated on in parallel. Dpark has the feature of functional programming, including two types of operations on RDD: *Transformations* and *Actions*. Transformations create a new dataset from an existing one. All transformations in Dpark are lazy, in that they do not compute their results right away. The transformations are only computed when an action, which return a value to the driver program after running a computation on the dataset, requires a result to be returned to the driver program.

Many methods [2][19] have been proposed to find the overlapping communities. Unfortunately, most of them are not easy to parallel and are time consuming when analyzing large-scale networks. To overcome these limitations, we propose a Hybrid Label Propagation Algorithm (HLPA), a near linear algorithm to detecting overlapping communities using hybrid updating scheme on large-scale real-world social networks with distribution computing framework Dpark. This paper improves the state-of-the-art by making three contributions. First, to the best of our knowledge, we are the first group to implement a parallel community detection method with *Dpark*. Second, it introduces a *hybrid updating scheme* to make the algorithm more stable and scalable, and *different initialization* methods for detecting communities on directed, undirected and bipartite networks. Third, it can prevent to produce a "monster" community and several small communities by *label decaying strategy*.

The rest of the paper is organized as follows. In Section II, the proposed HLPA algorithm are presented. In Section III, we present the community detection experiment on real networks, and its results and evaluation. In Section IV, the previous work on label propagation based algorithms is elaborated. Finally, we conclude with future work in Section V.

II. Algorithm

A. Hybrid Updating Scheme

Parallelism and scalability have become very important when the social network is growing larger and larger, or real-time requirement of analyzing the social network data.

LPA based methods iteratively update each node's label based on its neighbors labels. There are two kinds of updating scheme: synchronous and asynchronous. Suppose, $C_x(t)$ represents the label of node x at t^{th} iteration, and node x has k neighbors. In synchronous updating way, node x at t^{th} iteration updates its label based on the labels of its neighbors at $(t-1)^{th}$ iteration, where, $C_x(t) = f_{SynUpdating}(C_{x_1}(t-1),...,C_{x_k}(t-1))$, shown in Figure 1. However, synchronous updating may lead to oscillation situation of nodes' labels in bipartite or nearly bipartite structure networks. So, Raghavan *et al.* [13] suggest the asynchronous updating strategy to randomly update nodes in one iteration, where, $C_x(t) = f_{AsynUpdating}(C_{x_{i1}}(t),...,C_{x_{im}}(t),C_{x_{i(m+1)}}(t-1),...,C_{x_{ik}}(t-1))$, shown in Figure 2. However, the asynchronous scheme may detect the very different division results at each run. The stability and quality issue of detection results need to be settled. As for SLPA, it updates the list of labels in one iteration will be reflected in the subsequent iterations. Therefore, the nodes cannot be processed completely independent of each other. Each node is a neighbor of some other nodes, therefore, if the lists of labels of its neighbors are updated, it will receive a label randomly picked from the updated list of labels [20].

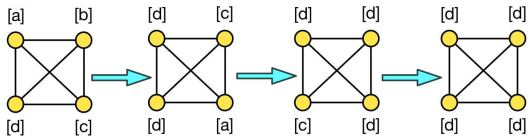

Fig. 1: Illustration of the Label Propagation with synchronous updating.

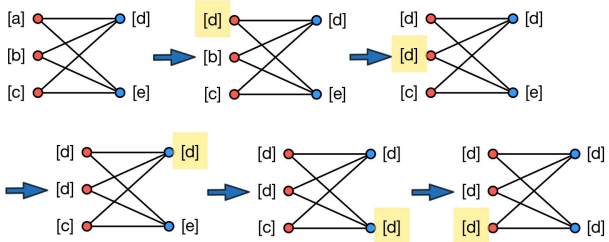

Fig. 2: Illustration of the Label Propagation with asynchronous updating.

As we know, edges only exist between the bi-side nodes in bipartite network. To expand to different networks, we denote two side nodes of one pair edge in networks as $left$ nodes and *right* nodes for instructions. For example, if the network is directed, we denote source nodes as $left$ nodes, and target nodes as *right* nodes. Because the observation of the number of communities is at most equal to the number of large side nodes, we only need to assign one side node with a unique label at the initialization stage. In our HLPA method, we design a hybrid updating scheme to updating labels at each iteration. Suppose we initialize the label of *right* nodes, then the updated labels from *right* nodes propagated to $left$ nodes, then iterate back and forth until reach the maximal iteration. When propagating labels from one side nodes to the other, the updating process is independent from each side nodes. In Figure 3, it shows the illustration of HLPA. At the first iteration, we only initialize the *right* nodes with unique labels, and then update the $left$ nodes' label list, each neighbor node of each $left$ node selects one label from its label list with probability, and each $left$ node choose the popular label updating to its label list. At the next iterations, it will iterate updating the both side nodes' label list until reach the maximal iteration. Then, get the memberships of each node by the most observing labels in its label memory list. At last, we get

[2]https://github.com/douban/dpark

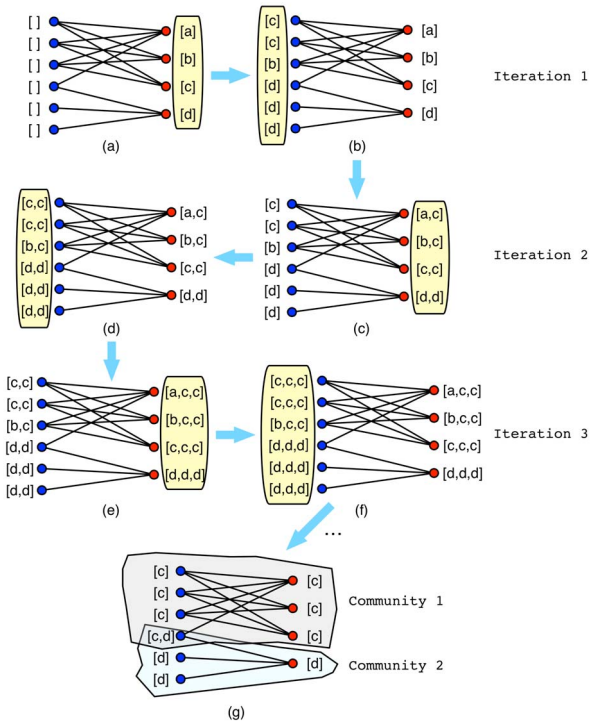

Fig. 3: Illustration of the HLPA

two communities with overlap in the example. Such makes the updating labels of one side nodes synchronous and easily implement paralleled version. At the same time, HLPA is not an entire synchronous updating scheme, the design can prevent the oscillation well.

B. Label Decaying Strategy

Due to the nature that label propagation is an epidemic process, a label can spread too far from its origin and produce a "monster" community, it may have more than half the nodes of the large network. To avoid the monster community, we take a label decaying strategy. After each iteration, we calculate the sizes of each communities, and if it is large enough, the label of this community will add to a cut down label lists (IGNORE), and the labels in IGNORE will not be propagated at the next iteration. The advantages of this strategy is, the first, it will prevent the growing of the big communities and avoid the monster communities, and the second, the small communities can still iterate until fully growth.

C. HLPA

Hybrid Label Propagation Algorithm (HLPA) is an overlapping community detection algorithm on large-scale real-world social networks based on SLPA, shown in Algorithm 1. Firstly, we novelly use hybrid updating scheme to enhance the efficiency of labels propagation process and avoid the oscillation phenomenon in bipartite or nearly bipartite networks. Secondly, we implement HLPA with Dpark and benefit from Resilient Distributed Datasets (RDDs) data structure to achieve parallelism easily.

Algorithm 1: HLPA-Hybrid Label Propagation Algorithm

Input: T: the user defined maximum iteration; r: post-processing threshold; $IGNORE$: cut down labels list

Output: $\mathscr{C}$: communities set

1 Initialization: Organize the graph as two side structure with the connection between, $left$ represent the left nodes set and *right* represent the right nodes set. Suppose we initialize the label of *right*. (Using different strategies on different kinds of networks.)
2 Hybrid updating labels scheme: Iterate until the maximum iteration T is reached
3 1) Synchronously propagate label from *right* to $left$:
4 a. Randomly select a label in each *right* node's label list but not in $IGNORE$.
5 b. Each node at $left$ get the updated labels list from its neighbors previously selected label at *right*
6 c. Choose the most popular label as the updated label (if there are more than one popular labels, randomly choose one) and add the updated label to $left$ labels list.
7 2) Synchronously propagate label from $left$ to *right*, same as step one.
8 3) Get communities with the same label and update $IGNORE$.
9 Post-processing:
10 Get the majority labels in each labels list with the probability over the threshold r, and output the final communities $\mathscr{C}$.

1) Implementation with Dpark: Three Dpark implementation algorithms, Directed-HLPA, Undirected-HLPA, and Bipartite-HLPA, targeted on three kinds of networks, directed, undirected, and bipartite networks are elaborating detailed in this section.

Directed-HLPA: The HLPA can support the directed network input. We set the directed edge as a pair, and arrange the source nodes which links the others as $left$ nodes, and the target nodes as *right* nodes. We can get the information of the neighborhood who has influence on the others. We always believe the target nodes (right nodes) have the impact on the source nodes (left nodes). So, we first initialize the all nodes' labels. In each iteration, we update left side nodes' labels based on the right side neighbor information and right side nodes' labels based on the left side information. To get the overlapping detection results, a memory list is set for each node to store the labels in each updated iteration.

Undirected-HLPA: For detecting communities on undirected networks, first, copy all reversed edges appended to the origin edges, and remove the duplicate edges. The rest progress is like *Directed-HLPA*, iteratively update the nodes' label list, and to get final community detection results. We deal with the iteration process only through propagating the left side nodes based on the last iteration labels information in the nodes' memory list.

Bipartite-HLPA: First, we initialize the labels of one side (with larger size) of nodes marked as *right*, and the other side as $left$.

Second, propagate right nodes' labels to the left ones, that is to say, update left nodes' label memory list. The updating process is using flag $isLeftFlag$ to control the direction of propagation, if it is $True$ then propagate right nodes' labels to the left ones, otherwise, if flag $isLeftFlag == False$, propagating blue nodes' label to right ones.

Finally, get community detection results, and due to the structure of bipartite network, we need to filter the communities whose has only one side nodes.

2) Post-processing: If the probability of a label in the label list is less than threshold $r \in [0, 0.5]$, the label should be removed. After thresholding, the nodes having the same label will group as one community. Corresponding to the fact that one node may belong to multiple communities. When $r \geq 0.5$, HLPA outputs disjoint communities.

All these improvements, such as hybrid updating scheme, different initialization methods and label decaying strategy, along with the Dpark parallel implementation are making HLPA more effective and efficient. Next section will give the experiment results and evaluation of our method on real networks.

III. Experiment

We tested the detection quality and efficiency of HLPA on benchmark real networks and large-scale real networks.

A. Evaluation on Benchmark Real Networks

We conduct our experiments on four well known benchmark real networks, including Zachary's karate club network (*karate*)[21], American College Football network ($football$)[22], Political blogs network (polblogs)[23] and Southern Women network (*women*)[24]. The detailed information of each network is shown in Tabel. I.

We take Normalized Mutual Information (NMI) metric to evaluate the communities results detected by different algorithms with the ground truth. We compare our results against Label Propagation based methods, LPA, SLPA and our previous work iSLPA [25], which is also a Label Propagation algorithm without label decaying strategy. The NMI evaluation of detection results are shown in Table. II.

In the case of the Zachary's karate, American College Football and Southern Women networks, the results of the HLPA are better than those of LPA, SLPA, and iSLPA without decaying strategy. And, in the case of the Political Blogs network, LPA and SLPA do not support the directed networks detecting.

B. Evaluation on Large-scale Real Networks

Douban[3] is a Chinese SNS website. Its users create or tag content related to film, books, music, and recent events and activities, and share with others. Douban can help its users get recommendations about books and movies based on the user behaviors and comments about items. In this study, we chose several datasets on the Douban site. The first dataset is a small network of friends who has broadcast in public, which has 9,613 vertices and 20,000 edges, originally it is a directed network marked as *Directed-1*. An undirected network is created from the original directed network marked as *Undirected-1*. The second dataset is a larger friends network which has 190,641 vertices and 8,901,291 edges, marked as *Directed-2*, and the undirected projection network is marked as *Undirected-2*. The third group of datasets is user-movie dataset, that is the users and the movies they cited on Douban site. They are a small bipartite user-movie network marked as *Bipartite-1* which has 475 users and 1,257 movies and 10,000 edges, a middle bipartite user-movie network marked as *Bipartite-2* which has 22,578 users and 12,128 movies and 892,638 edges, and a large bipartite user-movie network marked as *Bipartite-3* which has 3,044,092 users and 135,968 movies and 171,685,859 edges, and a projected user-user undirected network from user-movie bipartite network marked as *Undirected-3* which has 1,183,830 vertices and 118,382,984 edges.

Table III gives description of the experiments including the vertex, edges and average degrees K of different social network datasets, and the detection results of HLPA algorithm including number of communities and time consuming. In our experiment, we set $iteration \geq 20$ and r varies from 0.01 to 0.1 for HLPA, and the performance with the optimal parameter is reported for all algorithms with tunable parameters. The results show that it only takes 37.12 minutes to get the communities detection results when running HLPA algorithm on a 3 million users, 0.13 million movies and 1.7 billion edges large-scale bipartite network. The running time of HLPA is faster than SLPA, later renamed to GANXiS[4], on the these networks. Among the overlapping community detection algorithms [26], HLPA has the fastest run-time with Dpark on Douban's networks. Other methods mentioned in the introduction, such as LFM, GCE, CFinder, was not able to run on the largest unbipartite networks (*Undirected-3*) within the time frame of 4 hours. These algorithm are very computationally expensive and thus may not be suitable for detecting large-scale networks.

Detailed analysis on Directed-2 network:

Since the network Douban does not provide a standard result which can be used to compare, we analyze the results of *Directed-2* through different angles. The proposed method detected 72,363 communities of different sizes in this network. Table IV lists the description of the top five real communities detected. Consequently, the community detection result obtained from Douban via HLPA is sound and accurate.

IV. Related Work

This section summarizes three label propagation based algorithms.

A. Label Propagation Algorithm (LPA)

Label propagation algorithm (LPA) [14] is an extremely fast community detection method and is widely used in large scale networks. Also, LPA is scale independent for community

[3]http://www.douban.com

[4]https://sites.google.com/site/communitydetectionslpa/

Attr.	*karate*	*football*	*polblogs*	*women*
# type of graphs	undirected	undirected	directed	bipartite
# of nodes	34	115	1222	women:18, events:14
# of edges	78	613	16782	89
# of communities	2	12	2	2
# our communities	2	11	138	2

TABLE I: The statistics of real networks.

Datasets	LPA	SLPA	Our iSLPA	Our HLPA
karate	0.584	0.511	0.742	**0.837**
football	0.624	0.617	0.742	**0.766**
political blogs	-	-	0.196	**0.235**
southern women	0.184	0.424	0.438	**0.560**

TABLE II: The NMI results of LPA, SLPA, Our iSLPA and Our HLPA on real networks.

Data	Vertexes	Edges	K	Num.	HLPA	SLPA	Description
Directed-1	9,613	20,000	2.08	175	4.73min	-	friends-small
Undirected-1	9,613	20,000	2.08	252	6.85min	8.61min	friends-small
Directed-2	190,641	8,901,291	46.69	72,363	7.87min	-	friends-large
Undirected-2	190,641	8,901,291	46.69	19,016	8.93min	11.21min	friends-large
Bipartite-1	U:475 M:1,257	10,000	7.96	29	13.75min	17.78min	user-movie-small
Bipartite-2	U:22,578 M:12,128	892,638	39.54	240	14.74min	19.15min	user-movie-middle
Bipartite-3	U:3,044,092 M:135,968	171,685,859	56.40	11,226	37.12min	48.37min	user-movie-large
Undirected-3	1,183,830	118,382,984	100	22	24.50min	30.67min	projected user-user

TABLE III: Experiment results of community detection on large-scale real-world social networks

Comm. Num.	Nodes	Aver. in-degree	Aver. out-degree	Description
1	98,649	74	61	famous person
2	14,852	88	73	book reviewers
3	2,093	341	66	authors
4	2,005	85	78	architectures
5	1628	102	79	financial person

TABLE IV: Description of top 5 communities on Directed-2.

detection because it doesn't involve modularity optimization [11]. The idea of LPA is simple. There is a label for each node in a network and communities are defined as the sets of nodes with same labels. Initially, every node is assigned with a unique label. At every step, each node updates its label to a new one which is most of its neighbors shares. If a node has two or more maximal labels, it picks one randomly. The stop criteria is until every node has a label that is the maximum label of its neighbors. In this fashion, densely connected group of nodes can reach a consensus on a unique label and form a community quickly [27]. However, each node can only belong to one community in LPA.

B. Community Overlap PRopagation Algorithm (COPRA)

As an extension of LPA, the COPRA [14] is able to find the overlapping communities. At each iteration, it synchronously updates each node's *belonging coefficients* by averaging the coefficients over all its neighbors, until community members reach a consensus on their community membership. In its results, each vertex can belong to v communities at the most. However, the parameter v is vertex-independent. It is hard for COPRA to adapt the situation of some vertices with a small number of community memberships and some others with a large number of community memberships.

C. Speaker-Listener Propagation Algorithm (SLPA)

Xie *et al.* propose a Speaker-listener Label Propagation Algorithm (SLPA) [15][16], it is a mimic to information propagation process, liking people's preference to spread most popular discussed information. Unlike the other algorithms, SLPA will not forget a node's information of labels gained in previous iterations, by using memory list to store labels from each iteration. The membership strength is interpreted by the probability of observing a label in a nodes's memory list, when a node observes more the same labels, the more likely it will spread this label to other nodes. This memory list design makes it able to detect the overlapping communities.

V. Conclusion

This paper presents an improved and scalable community detection algorithm HLPA implemented with distribution computing framework Dpark, by using a novel hybrid label updating scheme, label decaying strategy and different initialization methods on different networks. Comparison with the

previous label propagation based algorithms, HLPA has shown relatively high accuracy on benchmark real networks and has competitively performance on the large real networks.

In future work, we plan to apply HLPA after the dimension reduction of heterogeneous social networks to get meaningful community structure. We also plan to extend HLPA to weighted networks community detection.

REFERENCES

[1] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Phys. Rev. E*, vol. 69, no. 2, p. 026113, Feb. 2004.

[2] A. Lancichinetti and S. Fortunato, "Community detection algorithms: a comparative analysis," 2009, cite arxiv:0908.1062.

[3] A. Lancichinetti, S. Fortunato, and J. Kertész, "Detecting the overlapping and hierarchical community structure in complex networks," *New J. Phys.*, vol. 11, no. 3, Mar. 2009.

[4] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *Nature*, vol. 435, no. 7043, pp. 814–818, June 2005.

[5] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Phys. Rev. E*, vol. 78, no. 4, p. 046110, Oct. 2008.

[6] C. Lee, F. Reid, A. McDaid, and N. Hurley, "Detecting highly overlapping community structure by greedy clique expansion," in *Workshop on Social Network Mining and Analysis*, 2010, cite arxiv:1002.1827.

[7] T. Wang, X. Qian, and H. Xu, "An improved parallel hybrid seed expansion (PHSE) method for detecting highly overlapping communities in social networks," in *Advanced Data Mining and Applications, 9th International Conference, ADMA 2013, Hangzhou, China, December 14-16, 2013, Proceedings, Part I*, 2013, pp. 385–396.

[8] A. Lancichinetti, F. Radicchi, J. J. Ramasco, and S. Fortunato, "Finding statistically significant communities in networks," *CoRR*, vol. abs/1012.2363, 2010.

[9] M. Rosvall and C. T. Bergstrom, "Maps of random walks on complex networks reveal community structure," *Proceedings of the National Academy of Sciences*, vol. 105, no. 4, pp. 1118–1123, 2008.

[10] M. E. Newman, "Modularity and community structure in networks," *Proc Natl Acad Sci U S A*, vol. 103, no. 23, pp. 8577–8582, Jun. 2006.

[11] I. X. Y. Leung, P. Hui, P. Liò, and J. Crowcroft, "Towards real-time community detection in large networks," *Physical Review E*, vol. 79, no. 6, pp. 066 107+, Jun. 2009.

[12] S. Fortunato and M. Barthelemy, "Resolution limit in community detection," *Proceedings of the National Academy of Sciences*, Jan. 2007.

[13] U. N. Raghavan, R. Albert, and S. Kumara, "Near linear time algorithm to detect community structures in large-scale networks," *Phys. Rev. E*, vol. 76, p. 036106, Sep 2007.

[14] S. Gregory, "Finding overlapping communities in networks by label propagation," *New Journal of Physics*, vol. 12, no. 10, pp. 103 018+, Oct. 2010.

[15] J. Xie, B. K. Szymanski, and X. Liu, "Slpa: Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process," *CoRR*, vol. abs/1109.5720, 2011.

[16] J. Xie and B. K. Szymanski, "Towards linear time overlapping community detection in social networks." in *PAKDD (2)*, ser. Lecture Notes in Computer Science, P.-N. Tan, S. Chawla, C. K. Ho, and J. Bailey, Eds., vol. 7302. Springer, 2012, pp. 25–36.

[17] N. Buzun, A. Korshunov, V. Avanesov, I. Filonenko, I. Kozlov, D. Turdakov, and H. Kim, "Egolp: Fast and distributed community detection in billion-node social networks," in *Data Mining Workshop (ICDMW), 2014 IEEE International Conference on*, Dec 2014, pp. 533–540.

[18] V. Blondel, J. Guillaume, R. Lambiotte, and E. Mech, "Fast unfolding of communities in large networks," *J. Stat. Mech*, p. P10008, 2008.

[19] J. Xie, S. Kelley, and B. K. Szymanski, "Overlapping community detection in networks: The state-of-the-art and comparative study." *ACM Comput. Surv.*, vol. 45, no. 4, p. 43, 2013.

[20] K. Kuzmin, S. Y. Shah, and B. K. Szymanski, "Parallel overlapping community detection with slpa." in *SocialCom*. IEEE, 2013, pp. 204–212.

[21] W. Zachary, "An information flow model for conflict and fission in small groups," *Journal of Anthropological Research*, vol. 33, pp. 452–473, 1977.

[22] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.

[23] L. A. Adamic and N. Glance, "The political blogosphere and the 2004 U.S. election: divided they blog," in *Proceedings of the 3rd International Workshop on Link Discovery*, ser. LinkKDD '05, New York, NY, USA, 2005, pp. 36–43.

[24] A. Davis, B. Gardner, and M. Gardner, *Deep South: A Social Anthropological Study of Caste and Class*. Chicago, IL: University of Chicago Press, 1941.

[25] T. Wang, X. Qian, and X. Wang, "Label propagation based community detection algorithm with dpark," in *Computational Social Networks: 4th International Conference, CSoNet 2015, Beijing, China, August 4-6, 2015, Proceedings*, ser. Lecture Notes in Computer Science. Springer International Publishing, 2015, pp. 116–127.

[26] S. Harenberg, G. Bello, L. Gjeltema, S. Ranshous, J. Harlalka, R. Seay, K. Padmanabhan, and N. Samatova, "Community detection in large-scale networks: a survey and empirical evaluation," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 6, no. 6, pp. 426–439, 2014.

[27] X. Liu and T. Murata, "Community detection in large-scale bipartite networks." in *Web Intelligence*. IEEE, 2009, pp. 50–57.