Community Detection in Large-scale Bipartite Networks

Xin Liu, Tsuyoshi Murata
Department of Computer Science, Graduate School of Information Science and Engineering
Tokyo Institute of Technology
Tokyo, Japan
e-mail: tsinllew@ai.cs.titech.ac.jp, murata@cs.titech.ac.jp

***Abstract*—Community detection in networks receives much attention recently. Most of the previous works are for unipartite networks composed of only one type of nodes. In real world situations, however, there are many bipartite networks composed of two types of nodes. In this paper, we propose a fast algorithm called LP&BRIM for community detection in large-scale bipartite networks. It is based on a joint strategy of two developed algorithms -- label propagation (LP), a very fast community detection algorithm, and BRIM, an algorithm for generating better community structure by recursively inducing divisions between the two types of nodes in bipartite networks. Through experiments, we demonstrate that this new algorithm successfully finds meaningful community structures in large-scale bipartite networks in reasonable time limit.**

Keywords-community detection; bipartite networks; complex networks; modularity

I. INTRODUCTION

Most of the real world networks typically have community structure -- networks can often be divided into communities, within which the connections are dense, but between which they are sparser [1]. To evaluate the quality of a particular division of a network into communities, Newman introduces a qualitative measure called modularity [8]. Among various kinds of algorithms for detecting communities in networks [4] [10], maximization of modularity is effective and popular.

One of the important classes of networks is bipartite networks (as opposed to the general unipartite networks), where nodes can be divided into two types, such that no two nodes of the same type are adjacent. Examples of such networks are actor-event networks, author-paper networks, etc. Recently, a few researchers investigated the community detection problem in bipartite networks.

In Guimera et al.'s opinion [7], a bipartite network community is composed of nodes of the same type. In an actor-event network, for example, actors form actor communities and events form event communities. An actor community is composed of a group of actors where each pair of them should have a higher frequency to co-participate in some events. Based on this idea, they devise a bipartite modularity and employ simulated annealing for modularity optimization. Consequently, their algorithm aims at detecting communities for nodes of one type at a time.

Barber [5] regards a bipartite network community as a group of densely connected nodes, which is the same as the traditional viewpoint in a unipartite network. Based on this idea, Barber extends the definition of Newman's modularity to be appropriate for bipartite networks and presents a bipartite modularity. He also proposes an algorithm called *adaptive BRIM* for detecting community structure by maximizing this bipartite modularity.

Lehmann et al. [13] extend the *k*-clique community detection algorithm to bipartite networks. The main merits of their method are: (1) it can detect communities of different connection densities; (2) it can detect overlapping communities. The weakness is that it only explores those bipartite network structures whose densities are over the designated threshold, so community memberships of all nodes are not determined.

Murata [14] proposes a bipartite modularity, which gives consistent result as Newman's modularity when applied to unipartite networks. Wakita and Suzuki [15] advance a modified version of Murata's bipartite modularity, which can reflect the multi-facet correspondence among communities. However, effective and efficient algorithm for maximizing those measures is not available.

As there has not been an agreed-on definition for bipartite network communities, community detection in bipartite networks is still an open problem. Since Barber's method can be considered as a means for co-clustering both types of nodes, we accept his perspective as the definition for communities in bipartite network. In this paper, we go along with Barber's idea and propose a new algorithm for community detection in large-scale bipartite networks, which is difficult for *adaptive BRIM* to handle.

For simplicity, we denote the two types of nodes in bipartite networks as red and blue nodes, respectively. The paper is organized as follows: in the next section, the definition of modularity and Barber's bipartite modularity is reviewed. In section III, we give a survey of *BRIM*, *adaptive BRIM*, and label propagation (*LP*) algorithms. Our *LP&BRIM* algorithm is proposed in section IV. Experiments are shown in section V, followed by a conclusion in the last section.

II. BARBER'S BIPARTITE MODULARITY

To evaluate the accuracy of community structures in networks, Newman devises a quantitative measure called modularity [8]. Modularity measures the fraction of the

978-0-7695-3801-3/09 $26.00 © 2009 IEEE
DOI 10.1109/WI-IAT.2009.15

edges in a network that connect nodes within the same community minus the expected value of the same quantity in a null model [3], where the community divisions are the same but connections are made randomly between nodes. Consider a network with n nodes and m edges. Suppose k_i denotes the degree of node i and g_i represents the community to which node i belongs. If $A_{n\times n}$ is the adjacency matrix of the network, and $P_{n\times n}$ is the adjacency matrix of the null model, where $P_{ij}=k_ik_j/2m$ [3], the modularity is expressed as:

$$Q=\frac{1}{2m}\sum_{i\ j\ ,}(A_{ij}-P_{ij})\delta(g_i,g_j)\,. \tag{1}$$

For bipartite networks, let p be the number of red nodes and q the number of blue nodes. Assume k_i, d_j are the degree of red node i and blue node j, and g_i, h_j indicate the community that red node i and blue node j belong to, respectively. Because there is no edge whose ends are both red (blue) nodes, the adjacency matrix of the bipartite network could take a block off-diagonal form as:

$$A=\begin{bmatrix} 0_{p\times p} & \tilde{A}_{p\times q} \\ (\tilde{A}^{T})_{q\times p} & 0_{q\times q} \end{bmatrix}. \tag{2}$$

It is reasonable to require the corresponding null model to take bipartite structure and preclude edges between nodes with the same color, giving:

$$P=\begin{bmatrix} 0_{p\times p} & \tilde{P}_{p\times q} \\ (\tilde{P}^{T})_{q\times p} & 0_{q\times q} \end{bmatrix}, \tag{3}$$

where $\tilde{P}_{ij}=k_id_j/m$ [5]. Suppose $\tilde{B}=\tilde{A}-\tilde{P}$. Then we have:

$$\begin{aligned} Q&=\frac{1}{2m}\sum_{i=1}^{p}\sum_{j=1}^{q}(A_{ij}-P_{ij})\delta(g_i,h_j)\\ &=\frac{1}{m}\sum_{i=1}^{p}\sum_{j=1}^{q}\tilde{B}_{ij}\delta(g_i,h_j)\,. \end{aligned} \tag{4}$$

This is the bipartite modularity proposed by Barber. Without causing confusion, we use "bipartite modularity" to indicate "Barber's bipartite modularity" in the following parts.

III. BRIM, ADAPTIVE BRIM, AND LP

In this section, we give a survey of *BRIM*, *adaptive BRIM*, and *LP* algorithms, which are the basis of the following discussion.

A. BRIM

Let c denote the number of identified communities in a network. The index matrix of a community division, denoted by S, is defined as follows: it has dimensions of $n\times c$, with each row as an index vector of (0,1) element corresponding to a node, such that:

$$S_{ij}=\begin{cases} 1 & \text{if node } i \text{ belongs to community } j, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

In bipartite networks, S can be partitioned so that:

$$S=\begin{bmatrix} R_{p\times c} \\ T_{q\times c} \end{bmatrix}, \tag{6}$$

where R and T are the index matrix for red and blue nodes, respectively. An equivalent expression of bipartite modularity can be given by:

$$Q=\frac{1}{m}\sum_{i=1}^{p}[\sum_{k=1}^{c}R_{ik}(\tilde{B}T)_{ik}]\,. \tag{7}$$

Since each row of R consists of a single 1 with all other elements being 0, maximizing Q becomes simple: we just assign red node i to community k such that $(\tilde{B}T)_{ik}$ is the maximum of the ith row of $\tilde{B}T$. Note that $\tilde{B}T$ is completely determined by the division of blue nodes, it is straightforward to induce the division of red nodes from the division of blue nodes, with the aim of optimizing bipartite modularity. Similarly, we can deduce:

$$Q=\frac{1}{m}\sum_{j=1}^{q}[\sum_{k=1}^{c}T_{jk}(\tilde{B}^{T}R)_{jk}]\,, \tag{8}$$

which indicates how to induce the division of blue nodes from the division of red nodes. Taken together, these two procedures give birth to *BRIM* algorithm [5]: starting from a division of red nodes (we can also start from a division of blue nodes; the start division is called initial division.), we induce the division of blue nodes. From the division of blue nodes, we then induce the division of red nodes, and iterate. In this fashion, bipartite modularity increases until a local maximum is reached and hence a community division is brought.

B. Adaptive BRIM

Based on *BRIM*, Barber puts forward an *adaptive BRIM* algorithm for detecting community structure in bipartite networks [5]. The idea is to adopt a scheme of selecting c as a parameter and randomly assigning red (blue) nodes to c modules as an initial division. For different values of c, we obtain different initial divisions, so different candidate community divisions can be brought from *BRIM* algorithm. Barber assumes that the quality of the candidate community division, measured by bipartite modularity, depends on c in a smooth fashion. Then he adopts a *bi-section* method to search the right value of c, from which the candidate

community division gains greater bipartite modularity than from others. This candidate community division is thus the final community division.

C. LP

Raghavan and Albert propose a *LP* algorithm for detecting communities [6] recently. The most striking feature of *LP* is its less expensive computation than what is possible so far. The basic idea of *LP* is simple: Initially, every node is assigned with a unique label, which represents the community it belongs to. At every step, each node updates its label to a new one which is the most of its neighbors have (the maximal label). If a node has two or more maximal labels, it picks one randomly. In this iterative process, densely connected group of nodes can reach a consensus on a unique label and form a community quickly (Fig. 1).

The node updating process in *LP* can be done in a synchronous or asynchronously way. In synchronous updating [6], node x at the t-th iteration updates its label based on the labels of its neighbors at the (t-1)-th iteration. Suppose node x has k neighbors and $C_x(t)$ denote the label of node x at the t-th iteration. Hence, $C_x(t) = f_{LabelUpdating}(C_{x[1]}(t-1), ...,C_{x[k]}(t-1))$, where function $f_{LabelUpdating}$ returns the label that occurs most frequently among $C_{x[1]}(t-1), ...,C_{x[k]}(t-1)$. However, in bipartite networks, synchronous updating may result in oscillations of labels (Fig. 2). To solve this problem, Raghavan et al. suggest the asynchronous updating (Fig. 3), where $C_x(t) = f_{LabelUpdating}(C_{x[1']}(t), ..., C_{x[m']}(t), C_{x[(m+1)']}(t-1), ..., C_{x[k']}(t-1))$, and $x[1'], ..., x[m']$ are neighbors of x that have already been updated in the current iteration, while $x[(m+1)'], ..., x[k']$ are neighbors that are not yet updated in the current iteration. The order in which all the n nodes in the network are updated at each iteration is chosen randomly.

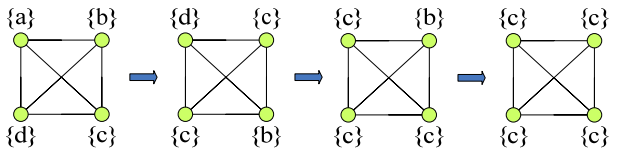

Figure 1. Illustration of label propagation algorithm. A letter in a bracket denotes a label of the corresponding node. In this example, labels are propagated in a synchronous way. After three iterations, this densely connected group reach a consensus on label "c" and form a community.

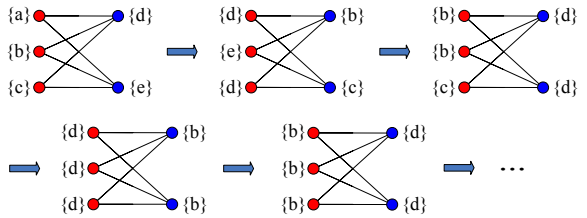

Figure 2. The oscillation phenomenon occurs when LP with synchronous updating scheme is applied to bipartite networks.

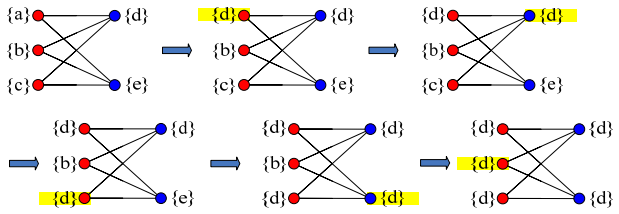

Figure 3. Illustration of the asynchronous updating scheme. The yellow grounding denotes the node whose label is currently being updated. In this example, the label updating sequence of nodes is delicately contrived, so that this connected group reach a consensus on label "d" very quickly.

IV. LP&BRIM

Adaptive BRIM works well in small-scale bipartite networks, but it is not suited to analyze large-scale bipartite networks. This is because the process of searching the right value of c is time-consuming. To solve this problem, we propose *LP&BRIM* algorithm, a joint strategy of *LP* and *BRIM*, which employs *LP* to search an initial division, and then use *BRIM* to find the final community division. Generally, *LP&BRIM* is able to detect community structure in large-scale bipartite networks in an accurate and fast way.

A. Reasons for Combining LP and BRIM

It is not hard to find that the community division found by *BRIM* depends solely on an initial division of the given network. In general, a good initial division motivates *BRIM* to find a better community division, while a bad one makes *BRIM* converge too early and get stuck at a poor local maximum. Hence a good initial division is the key for *BRIM* algorithm to work properly.

Thinking in another way, we can regard *BRIM* as a method for refining an initial division by optimizing bipartite modularity. Fig. 4a describes the flow chart of *adaptive BRIM* algorithm. In order to find a better initial division, we have to try various possible values of c. For different c, *random assignment* generates different initial divisions, which are refined into different candidate community divisions by *BRIM*. Then *bi-section* method is used to search the right value of c and the best candidate community division. Analytically speaking, *adaptive BRIM* is incompetent to handle large-scale networks. The reason is that it is very time-consuming to find the right value of c, as the number of possible values increases linearly with the scale of the network.

Adaptive BRIM adopts *random assignment* as a method for finding an initial division. What about other ways? Actually, there is a trade-off: too bad initial division keeps *BRIM* away from finding better result; yet we have to spend more time to attain a better initial division; however, if an initial division is good enough, there is no need for *BRIM* to refine it.

In large-scale networks, *LP* is a good choice for probing an initial division. The reasons are as follows:

- *LP* is very fast and appropriate for analyzing community structure of large-scale networks.

- Since *LP* is a community detection algorithm, the quality of the initial division probed by it is much better than that by *random assignment*.
- The quality of the initial division probed by *LP*, however, is not perfect, which can be further improved.

Therefore, we combine these two algorithms and propose a new algorithm called *LP&BRIM*, in which *LP* is employed as a means to probe an initial division and *BRIM* is used to refine it (Fig. 4b).

B. Combining LP and BRIM

There are some details about the combination of these two algorithms. In *LP* algorithm, although asynchronous updating can avoid oscillation that occurs in synchronous updating when applied to bipartite networks, it has a side effect: the algorithm becomes much unstable [2]. Since the label updating sequence of nodes is randomly chosen, different updating sequence may lead to different results. To solve this, we propose a new label updating scheme which is more suitable for *LP* to work in bipartite networks.

According to Barber's definition, a bipartite network community is a group of densely connected nodes. Because edges only exist between red and blue nodes, any significant community should contain both red and blue nodes, otherwise there is no edge within this community. Hence the number of communities in a bipartite network is at most equal to the number of red (blue) nodes. In light of this, rather than assigning labels to every node at the start, we assign each red (blue) node with a unique label and keep blue (red) nodes unlabeled. Since a node is required only to know the information about its neighbors when updating its label and each red (blue) node's neighbors are all blue (red) nodes, the updating process can be performed as follows: suppose we start from red nodes, then labels are propagated to blue nodes, later to red nodes again, and iterate (Fig. 5).

The new updating scheme, similar to the fashion of inducing divisions between red and blue nodes in *BRIM*, has some merits. First, it maintains near linear time complexity of the original *LP*, since it exerts no extra computation for label updating. Second, it neatly avoids the oscillation phenomenon. Moreover, note that, the relabeling process for different blue (red) nodes is independent of each other. This means that the label propagation from red (blue) nodes to blue (red) nodes is actually carried out synchronously, from the blue (red) nodes' perspective. Therefore, a parallelized algorithm is readily to be implemented for the new updating scheme, while parallelism is not easy for asynchronous updating scheme since its relabeling process is sequenced. Last, the new label updating scheme completely eliminates the uncertainty factor brought by the randomly chosen label updating sequence in asynchronous updating scheme. Therefore, its result is more stable than that of asynchronous updating scheme. For more information about the new asynchronous updating scheme, please refer to [16].

As *LP* is an iterative algorithm, when to stop it is another crucial point. In [6], Raghavan et al. suggest that the algorithm stops when every node has a label that is the maximal label of its neighbors. This criterion seems reasonable: in most times, the value of bipartite modularity of the corresponding division increases consistently, and reaches a plateau when the stop criterion is satisfied. But in some other cases, the modularity peaks before the stop criterion is satisfied, and afterwards it tends to decrease sharply [2]. Because of this phenomenon, assuming that bipartite modularity is a faithful measure of the quality of community structure, we choose the status corresponding to the highest bipartite modularity as an initial division.

C. Pseudo Code

Suppose the number of red nodes is less than blue nodes. For simplicity, in *LP* part we start label propagation from red nodes; in *BRIM* part we start inducing divisions between red and blue nodes from an initial division for red nodes. The pseudo code of *LP&BRIM* is described in Fig. 8.

D. Time Complexity

The fist part of *LP&BRIM*, *LP* algorithm, has a near linear time complexity [6]. Suppose an initial division found by *LP* contains c communities, and n denotes the number of nodes in the network. In *BRIM* part, when inducing the division of red nodes from the division of blue nodes, we assign each red node i to community k, such that $(\tilde{B}T)_{ik}$ is the maximum of the i-th row of $\tilde{B}T$. This operation involves calculating all elements of $\tilde{B}T$, which requires a time complexity of $O(cn)$. Inducing the division of blue nodes consumes the same running time. As *BRIM* algorithm ends after a certain number of inducing steps (the total number of steps is not a priori clear, but relatively few steps are needed.), its time complexity is $O(cn)$. For c, we do not know its exact value beforehand. But it is in the range of $[1, n]$. As a result, the total time complexity of *LP&BRIM* algorithm is always no more than $O(n^2)$.

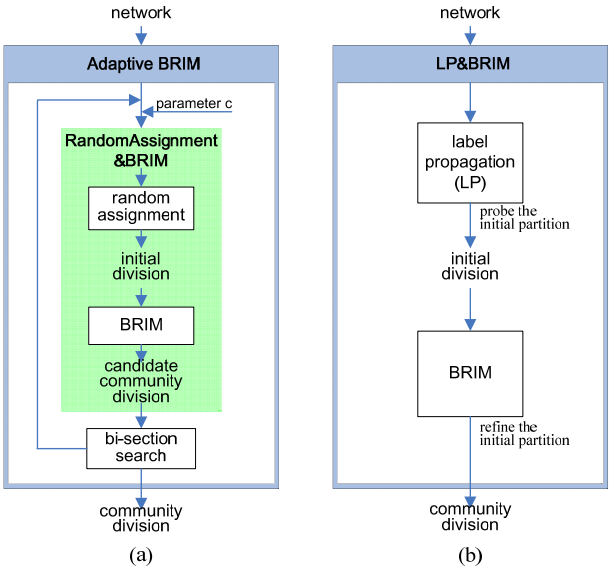

Figure 4. (a) Flow chart of adaptive BRIM algorithm. (b) Flow chart of LP&BRIM algorithm.

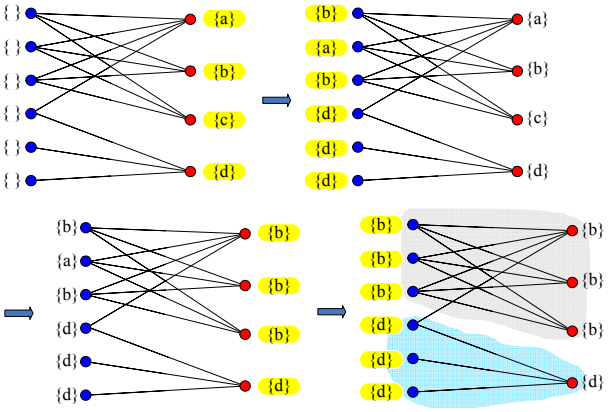

Figure 5. Illustration of the new label updating scheme.

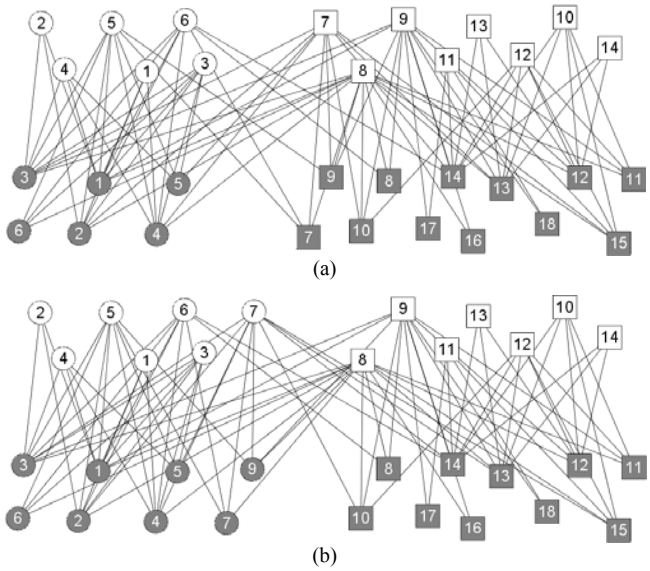

Figure 6. The initial division (a) and the final community division (b) obtained by LP&BRIM in southern women network. Event nodes are represented as open symbols with black labels and woman nodes as filled symbols with white labels. The nodes in the two communities are indicated as circle and rectangle shapes of symbols, respectively.

TABLE I. THE RESULTS OF ADAPTIVE BRIM AND LP&BRIM IN THE AUTHOR-PAPER NETWORK OF ARXIV DATABASE (RUNNING ON A PC WITH INTEL CORE 2 DUAL PENTIUM CPU @ 1.83G).

Algorithm	*adaptive BRIM*	*LP&BRIM*
Number of communities	128	2176
Average community size	312.39	18.38
Bipartite modularity	0.731653	0.783622
Running time(sec.)	189.032	95.140

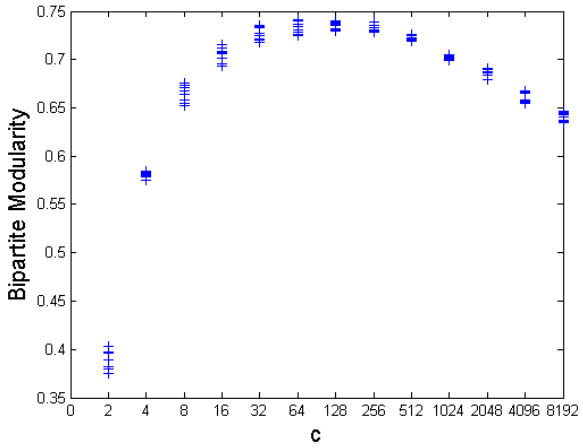

Figure 7. Bipartite modularity of the corresponding division generated by RandomAssignment&BRIM in the author-paper network of arXiv database.

V. EXPERIMENTS

A. *Southern Women Network*

As the first experiment, we consider southern women network [9] to verify the accuracy of *LP&BRIM*. This is because the network has been broadly analyzed by social network researchers and its community structure is known. There are 18 women nodes and 14 events nodes, with 89 edges linking to woman nodes and event nodes if the women attended the corresponding events.

The process of applying *LP&BRIM* to this network is as follows: first, *LP* probes an initial division, which contains two communities: {woman 1-6, event 1-6} and {woman 7-18, event 7-14}(Fig. 6a); then *BRIM* refines the initial division and obtains another two communities: {woman 1-7, 9, event 1-7} and {woman 8, 10-18, event 8-14} (Fig. 6b), with bipartite modularity increasing from 0.299710 to 0.321172. Davis, who collected the network data, has ever used ethnographic knowledge to divide women into communities {1-9} and {9-17} (woman 9 is a secondary member of both communities) [9]. Note that, if we only look at woman nodes, the final community division by *LP&BRIM* agrees with the one proposed by Davis, except for woman 8. Thus *LP&BRIM* is able to identify true community structure in this network.

B. *Author-paper Network of ArXiv Database*

Now we test the performance of *LP&BRIM* against *adaptive BRIM* in large-scale networks. First, we apply these two algorithms to an author-paper network [11], which contains 8638 author nodes and 31348 physics paper (arxiv hep-th) nodes, and 64154 edges which represent the authorship relations between author nodes and paper nodes. Table I shows the results of the experiment.

Bipartite modularity of the division found by *LP&BRIM* is about 0.05 larger than that by *adaptive BRIM*. For *LP&BRIM*, *LP* part found an initial division with bipartite modularity being as large as 0.783405. The initial division is

so good that the improvement by *BRIM* part is very little: *BRIM* terminates only after two inducing steps, with bipartite modularity increased by 0.000217, and the number of communities unchanged.

Note that there is a great difference in the number of communities identified by these two algorithms (128 vs. 2176). The reason is that, during the *bi-section* search for the right value of *c*, bipartite modularity of the corresponding candidate community division found by *RandomAssignment&BRIM* (a subsection in *adaptive BRIM*, see Fig. 4a) drops temporarily as *c* doubles from 64 to 128, thus the algorithm conjectures that the right value of *c* exists in the interval between 64 and 128. The *bi-section* method is based on the premise that the quality of the generated candidate community division, measured by bipartite modularity, depends on *c* smoothly. In fact, the *random assignment* also exerts an influence on the quality of the candidate community division. As the network scale becomes large, that influence becomes so strong that it makes *BRIM* get stuck at a poor local maximum. Consequently, the quality of the candidate community division does not change smoothly with *c* and the *bi-section* method is bound to fail. We run *RandomAssignment&BRIM* by directly assigning parameter *c* to be doubling from 2 until 8192, with each value running for 10 times. Fig. 7 describes the relationship between bipartite modularity of the generated candidate community division and *c*. We can find that for different values of *c* in the range of [16, 512], the corresponding bipartite modularities are very close to each other. It is really hard to use *bi-section* method to identify the most appropriate value for *c*. Therefore, the searched "right" value of *c* can be inaccurate.

We also tried to assign *c* with 2176 (the number of communities detected by *LP&BRIM*) to *RandomAssignment&BRIM*. Bipartite modularity of the obtained candidate community division is 0.690379, much smaller than 0.783622 (the bipartite modularity of the community division by *LP&BRIM*). This verifies that the influence posed by *random assignment* made *BRIM* get stuck at a poor local maximum. Hence it is not a perfect choice to probe the initial division using *random assignment*. Taken all together, *LP&BRIM* performs better than *adaptive BRIM* in this network.

C. User-board Network of Yahoo! Chiebukuro

In order to further compare *LP&BRIM* with *adaptive BRIM*, we generated a bipartite network composed of users and boards from the data of Yahoo! Chiebukuro (Japanese Yahoo! Answers, one of the most popular question-answering forums in Japan, http://chiebukuro.yahoo.co.jp). In this network, a user and a board are linked if this user posts questions or answers to that board. Statistics of the network is shown in table II.

Table III shows the results of *adaptive BRIM* and *LP&BRIM* in this network. For *LP&BRIM algorithm*, *LP* part finds an initial division of 62025 communities in about 1 hour; *BRIM* part then refines the initial division to 52368 communities, with bipartite modularity mounting from 0.239231 to 0.386434.

The *bi-section* method in *adaptive BRIM* algorithm again fails and only finds 6 communities in such a huge network. However, bipartite modularity of the community division by *adaptive BRIM* is about 0.0178 larger than that by *LP&BRIM*. This is because of modularity limit [12]: modularity favors network divisions with groups of communities combined into larger communities when the scale of network is sufficiently large, and thus the comparison of modularities for divisions with totally different numbers of communities is not straightforward and may lead to ambiguities. Through further analysis of the configuration of these communities detected by *LP&BRIM*, we find that 95.34% of the communities' size is no more than 100, and 99.39% no more than 200. Considering the conclusion of [11] that natural communities in large networks should be at very small size scales (up to about 100 nodes), the community division found by *LP&BRIM* is more reasonable than the one by *adaptive BRIM*, although bipartite modularity of the former is slightly smaller than that of the latter.

From the above three experiments, we can find that it is reasonable to combine *LP* and *BRIM* for community detection in large-scale bipartite networks: *LP* can find an initial division very fast; if the quality of the initial division is satisfactory, the optimization process by *BRIM* ends quickly, such as the case in subsection *B*; if the quality of the initial division is not good, *BRIM* can improve it, such as the case in subsection *A* and *C*. As a result, the joint strategy takes advantage of both parts and works very well. On the other hand, *adaptive BRIM* adopts *random assignment* as the scheme to search the initial division. Because *random assignment* is not exact, we have to try various initial divisions(controlled by parameter *c*), use *BRIM* to refine them respectively, and choose the best one. Analytically, this process is time-consuming, and not suited to analyze large-scale bipartite networks. However, through experiments with real large-scale networks, we find that adaptive *BRIM* may even fail to find the reasonable community structure because of the influence brought by *random assignment*. Comparatively speaking, *LP&BRIM* algorithm has abilities to find meaningful community structures in large-scale networks in reasonable time limit, both in terms of bipartite modularity and the configuration of the detected communities.

TABLE II. STATISTICS OF THE USER-BOARD NETWORK OF YAHOO! CHIEBUKURO

Number of red nodes (boards)	3,121,002
Number of blue nodes (users)	206,882
Number of edges	16,460,014
Average degree of red nodes	5.27395
Average degree of blue nodes	79.56233

TABLE III. THE RESULTS OF ADAPTIVE BRIM AND LP&BRIM ALGORITHMS IN THE USER-BOARD NETWORK OF YAHOO! CHIEBUKURO (RUNNING ON A PC WITH INTEL CORE 2 QUAD CPU @ 2.66G).

Algorithm	***adaptive BRIM***	***LP&BRIM***
Number of communities	6	52368
Average community size	554,647	64
Bipartite modularity	0.404188	0.386434
Running time(hr.)	0.54	149.16

VI. CONCLUSION

In this paper, we propose a new algorithm called *LP&BRIM* for community detection in large-scale bipartite networks. It is based on a joint strategy of two developed algorithms -- *LP*, a very fast community detection algorithm, and *BRIM*, an optimization algorithm for generating divisions with better community structures in bipartite networks. Our idea is to employ *LP* to probe the initial division, and then use *BRIM* to improve the initial division. In the first part, *LP* tries to find an initial division in a microcosmic way: communities are formed by self-organization which uses the network structure as its guide. In the second part, *BRIM* tries to refine the initial division in a macroscopical way: it targets at maximizing the global measure of bipartite modularity during the inducing process between red and blue nodes. It is also reasonable to combine these two algorithms: *LP* can find a coarse initial division very fast, while *BRIM* takes a role of refining the initial division. As a result, the joint two parts take advantage of each other and works very well together. Experiments show that *LP&BRIM* is more suitable than the former *adaptive BRIM* for community detection in large-scale bipartite networks, in terms of bipartite modularity and the configuration of the detected communities. In particular, we show that *LP&BRIM* is able to extract meaningful community structures from the user-board network of Yahoo! Chiebukuro, which contains as many as 3,327,884 nodes and 16,460,014 edges.

As the definition of communities in bipartite networks is still an open problem, we look forward to further developments in this area.

ACKNOWLEDGMENT

We gratefully thank Dr. Michael J. Barber (Austrian Institute of Technology GmbH) for valuable suggestions. We also thank Mr. Makoto Okamoto (Yahoo Japan Corporation), Prof. Kikuo Maekawa (The National Institute for Japanese Language), and Prof. Sadaoki Furui (Tokyo Institute of Technology) for allowing us to use the data of Yahoo! Chiebukuro.

REFERENCES

[1] 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, vol. 99, Jun., 2002, pp. 7821-7826.

[2] I. X. Y. Leung, P. Hui, P. Lio, and J. Crowcroft, "Towards real time community detection in large networks," Phys. Rev. E, vol. 79, no. 6, 066107, 2009.

[3] M. E. J. Newman, "Finding community structure in networks using the eigenvectors of matrices," Phys. Rev. E, vol. 74, 036104, Sep. 2006.

[4] L. Danon, J. Duch, A. Diaz-Guilera, and A. Arenas, "Comparing community structure identification," J. Stat. Mech., P09008, Sep. 2005, doi: 10.1088/1742-5468/2005/09/P09008.

[5] M. J. Barber, "Modularity and community detection in bipartite network," Phys. Rev. E, vol. 76, no. 6, 066102, Dec. 2007, doi: 10.1103/PhysRevE.76.066102.

[6] 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, no. 3, 036106, Sep. 2007, doi: 10.1103/PhysRevE.76.036106.

[7] R. Guimera, M. Sales-Pardo, and L. A. N. Amaral, "Module identification in bipartite and directed networks," Phys. Rev. E, vol. 76, no. 3, 036102, Sep. 2007, doi: 10.1103/PhysRevE.76.036102.

[8] M. E. J. Newman, "Modularity and community structure in networks," Proceedings of the National Academy of Sciences of the United States of America, vol. 103, no. 23, Jun., 2006, pp. 8577-8582, doi: 10.1073/pnas.0601602103.

[9] A. Davis, B. B. Gardner, and M. R. Gardner, Deep South, University of Chicago Press, 1941.

[10] S. Fortunato, and C. Castellano, "Community structure in graphs," Dec. 2007, arXiv:0712.2716.

[11] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney, "Community structure in large networks: natural cluster sizes and the absence of large well-defined clusters," 2008, arXiv:0810.1355.

[12] S. Fortunato, and M. Barthelemy, "Resolution limit in community detection," Proceedings of the National Academy of Sciences of the United States of America, vol. 104, no. 1, Jan. 2007, pp.36-41, doi: doi:10.1073/pnas.0605965104.

[13] S. Lehmann, M. Schwartz, and L. K Hansen, "Biclique communities," Phys. Rev. E, vol. 78, no. 1, 016108, 2008, doi: 10.1103/PhysRevE.78.016108.

[14] T. Murata, "Detecting communities from bipartite networks based on bipartite modularities", Proceedings of the 2009 IEEE International Conferene on Social Computing (SocialCom 09), Aug. 2009, in press.

[15] K. Wakita, and K. Suzuki, "Extracting multi-facet community structure from bipartite networks," Proceedings of the International Symposium on Social Intelligence and Networking (SIN 09), Aug. 2009, in press.

[16] X. Liu, T. Murata, "How does label propagation algorithm work in bipartite networks?" Proceedings of the 2009 International Workshop on Intelligent Web Interaction (IWI 09), Sep. 2009, in press.

Algorithm LP&BRIM
input: bipartite network G
output: community division
function:
initiate each red node with a unique label;
while (not every node has a label that is the maximum label of its neighbors)
    propagate labels from red nodes to blue nodes;
    propagate labels from blue nodes to red nodes;
end while
$division_{LP}$ = the division corresponding to the largest Bipartitemodularity in the above while loop;
$division_{red}$ = the division for red nodes in $division_{LP}$;
$division_{blue}$ = the division for blue nodes in $division_{LP}$;
inducingBlueFlag=true;
while (true)
    preBipartitemodularity = Bipartitemodularity($division_{red}$, $division_{blue}$);
    if (inducingBlueFlag == true)
        inducing $division_{blue}$ from $division_{red}$;
    else
        inducing $division_{red}$ from $division_{blue}$;
    newBipartitemodularity = Bipartitemodularity($division_{red}$, $division_{blue}$);
    if (newBipartitemodularity == preBipartitemodularity)
        break;
    else
        preBipartitemodularity = newBipartitemodularity;
        inducingBlueFlag = !inducingBlueFlag;
end while
division = ($division_{red}$, $division_{blue}$);
end function

Figure 8. The pseudo code of LP&BRIM algorithm.