Community Detection Using Restrained Random-Walk Similarity

Makoto Okuda, Shin'ichi Satoh, Yoichi Sato, and Yutaka Kidawara

Abstract—In this paper, we propose a *restrained random-walk similarity method* for detecting the community structures of graphs. The basic premise of our method is that the starting vertices of finite-length random walks are judged to be in the same community if the walkers pass similar sets of vertices. This idea is based on our consideration that a random walker tends to move in the community including the walker's starting vertex for some time after starting the walk. Therefore, the sets of vertices passed by random walkers starting from vertices in the same community must be similar. The idea is reinforced with two conditions. First, we exclude abnormal random walks. Random walks that depart from each vertex are executed many times, and vertices that are rarely passed by the walkers are excluded from the set of vertices that the walkers may pass. Second, we forcibly restrain random walks to an appropriate length. In our method, a random walk is terminated when the walker repeatedly visits vertices that they have already passed. Experiments on real-world networks demonstrate that our method outperforms previous techniques in terms of accuracy.

Index Terms—Community detection, graph, normalized mutual information, random walk

1 Introduction

Rapid progress in computer technology has fueled the analysis of big data across many professional disciplines, such as sociology, biology, and computer science. In this context, community detection technology, which locates subgraphs whose vertices are more tightly connected with each other than with those lying outside the subgraphs, has become increasingly important. Many previous community detection methods have relied on the optimization of network modularity [1], which is a quality index for the partition of a graph into communities [2], [3]. However, such methods have a major drawback, called the *resolution limit* [4], which may prevent them from detecting communities that are relatively small compared with the whole graph. To overcome the problem, some community detection methods have employed the random walk technique [5], which is a mathematical formalization of a path that comprises a succession of random steps [6], [7]. In addition, a large number of methods, including statistical inference approaches, have been developed to identify graph communities more accurately and/or faster [8], [9], [10], [11], [12]. However, no technique that perfectly detects the community structures of large and complex graphs has yet been developed [13], [14].

In this paper, we propose a *restrained random-walk similarity method* for detecting communities of graphs. In our method, the starting vertices of finite-length random walks on a graph are clustered into the same community if the walkers pass similar sets of vertices. The key idea comes from our consideration that the sets of vertices passed by random walkers starting from vertices in the same community must be similar, because the walkers initially tend to move around in that community.

The idea is reinforced with two conditions. The first is the prevention of the incorrect clustering that arises from abnormal random walks, whereby random walkers leave the community including the starting vertex soon after departing. The starting vertices of abnormal random walks are incorrectly clustered because the similarities between the sets of vertices passed by abnormal random walkers and the sets of vertices passed by normal random walkers will be low, even if their starting vertices are in the same community. Therefore, random walks that start from each vertex are executed many times, and vertices that are rarely passed by the walkers are excluded from the set of vertices that the walkers may pass.

The second condition is the forced termination of random walks after an appropriate time. According to our observation, in the first stage, a general random walker frequently visits vertices that it has not passed. In the second stage, the walker repeatedly visits vertices that it has already passed. At this point, the walker is in the community including the starting vertex. In the third stage, the walker finally moves from the initial community to another community. For the proposed restrained random-walk similarity method, random walks would ideally be terminated in the second stage, because the similarities among the sets

- *M. Okuda is with the Universal Communication Research Institute, National Institute of Information and Communications Technology (NICT), Koganei, Tokyo 184-0015, Japan. E-mail: m-okuda@nict.go.jp.*
- *S. Satoh is with the Digital Content and Media Sciences Research Division, National Institute of Informatics (NII), Chiyoda-ku, Tokyo 100-0003, Japan. E-mail: satoh@nii.ac.jp.*
- *Y. Sato is with the Institute of Industrial Science, The University of Tokyo, Meguro-ku, okyo 113-8654, Japan. E-mail: ysato@iis.u-tokyo.ac.jp.*
- *Y. Kidawara is with the Universal Communication Research Institute, National Institute of Information and Communications Technology (NICT), Soraku-gun, Kyoto 619-0289, Japan. E-mail: kidawara@nict.go.jp.*

Manuscript received 15 Nov. 2018; revised 12 June 2019; accepted 26 June 2019. Date of publication 1 July 2019; date of current version 3 Dec. 2020.
(Corresponding author: Makoto Okuda.)
Recommended for acceptance by P. Ravikumar.
Digital Object Identifier no. 10.1109/TPAMI.2019.2926033

of vertices passed by the random walkers departing from vertices in the same community will be maximized. Our method restrains random walks when the number of unique vertices, or the size of the set of vertices, passed by the walkers only increases slightly over several steps.

This second condition is especially effective for graphs including both small and large communities. To successfully cluster vertices in large communities with our method, it is desirable for the number of steps in random walks to be large, because the similarities among the sets of vertices passed by walkers departing from vertices in the same large community will be high. However, when the number of steps in random walks is large, random walkers departing vertices in small communities are likely to move out of their initial communities before finishing their walk. As a result, many vertices in the small clusters will be incorrectly clustered, because the similarities between the sets of vertices passed by walkers departing from vertices in the small communities and in other communities will be high. In such a case, the second condition prevents the random walkers from moving out of their initial small community. As a result, the restrained random-walk similarity method successfully clusters vertices in both small and large communities.

Experiments conducted using synthetic networks indicate that our method is competitive with the best previous method in terms of accuracy, even when the network is large or includes both small and large communities; however, our method is not applicable to networks with a high mixing rate (the proportion of links from one vertex to vertices outside its community). Despite this drawback, our experiments also demonstrate that the proposed method detects communities of real-world networks much more accurately than previous techniques.

The remainder of this paper is organized as follows. Section 2 reviews related studies. Section 3 describes the restrained random-walk similarity method, before Section 4 presents the results of experiments to evaluate our method. Finally, Section 5 presents our conclusions and describes topics for future research.

2 Related Work

Many community detection methods have been proposed. This section presents the more significant studies, divided into categories based on the type of approach. Comprehensive overviews of community detection techniques can be found in previous papers [2], [3].

2.1 Traditional Graph Partitioning

Some early methods clustered a network by determining a minimum cut between two vertices through maximum-flow algorithms [15] based on the *max-flow min-cut* theorem [16]. This theorem states that the maximum amount of flow passing from the source vertex to the sink vertex is equal to the total weight of the edges in the minimum cut, i.e., the smallest total weight of the edges that separate the source from the sink. Other popular methods partitioned a graph by minimizing measures that are affine to the cut size, such as the *conductance* [2], the *ratio cut* [17], or the *normalized cut* [18]. Minimizing the conductance and the normalized cut favors partitions into clusters of approximately equal size in terms of the number of edges. Minimizing the ratio cut tends to produce clusters of almost equal size in terms of the number of vertices. Such features contradict the community structure of many networks. The ratio cut and the normalized cut can be minimized effectively using spectral clustering, which partitions data according to the eigenvalues of the similarity matrix of the data or other matrices derived from it. These graph partitioning techniques have the serious drawback that the number of clusters must be set beforehand, even though it is generally unknown.

2.2 Modularity Optimization

After Newman and Girvan defined the modularity [1], various community detection methods were developed on the basis of modularity optimization. Some of these were greedy methods that seek the global optimum modularity by making a locally optimal choice at each stage [19], [20], [21], [22], [23]. Other methods relied on simulated annealing [24], which is a probabilistic procedure for global optimization [25], [26], [27], [28], extremal optimization [29], which is a heuristic search procedure [30], spectral optimization using the eigenvalues of the modularity matrix [31], [32], [33], [34], [35], or genetic algorithms [36], which are heuristic search and optimization techniques inspired by natural evolution [37]. Newman showed the interesting fact that detecting the community structure by maximizing the modularity is equivalent to normalized-cut graph partitioning when the spectral approach is taken to split a graph into two groups [38]. Even though many community detection methods based on modularity optimization have been developed and analyzed, such methods suffer from the resolution limit.

2.3 Potts Model

Reichardt and Bornholdt showed that the problem of community detection could be mapped onto that of finding the ground state of an infinite-ranged Potts spin glass [8]. The community structure of the network was interpreted as the spin configuration that minimized the energy of the spin glass, with the spin states being the community indexes. Their derived Hamiltonian can recover the modularity defined by Newman and Girvan [1] with appropriate parameter settings. Therefore, finding the spin configuration for which the Hamiltonian is minimal is equivalent to maximizing the modularity under a certain condition.

2.4 Random Walk

Random walk techniques have often been used to detect communities. Pons and Latapy defined distance measures between vertices and between sets of vertices that effectively capture the community structure in a network [39]. The distances are calculated from the probabilities that a random walker will move from one vertex to another in a fixed number of steps. The vertices are then grouped into communities by the agglomerative hierarchical clustering technique [40]. Rosvall and Bergstrom introduced the *map equation* for community detection, which specifies the theoretical limit of how concisely the trajectory of a random walker in a network can be described for a given network partition [41]. The community structure is found by minimizing the map equation over all possible network partitions with some optimization

technique. In this paper, we refer to this method as *Infomap*. Fu et al. proposed *CD-TRandwalk* (a scalable Community Detection method based on Threshold Random walkers) according to some social principles [6]. Their method clusters vertices that have many common neighbors into the same community using the random walk technique. Meo et al. proposed *CONCLUDE* (COmplex Network CLUster DEtection), which aims to combine the accuracy of global methods with the efficiency of local methods [7]. CONCLUDE successfully mitigates the resolution limit by weighting the edges of a graph with random and non-backtracking walks of finite length prior to community detection based on modularity optimization.

Instead of a random walk, the *PageRank* technique is sometimes used for community detection or graph partitioning [42]. When Infomap is applied to a directed graph, the random walk process is modified by introducing a teleportation probability that links every node to every other node, just like the PageRank algorithm, thus guaranteeing a unique steady state distribution [41]. Andersen et al. proposed a local graph partitioning algorithm that finds cuts with small conductance values using a variation of PageRank with a specified starting distribution [43].

Our proposed method also employs the random walk technique. However, the basic concept used to find the community structure is novel. The similarity between vertices that we introduce, i.e., the random walk similarity, is quite different from the distance [39], association degree [6], and proximity [7] between vertices used in previous methods.

2.5 Statistical Inference

The problem of community detection has also been tackled by using statistical inference to fit a generative network model to the data. The stochastic block model is the most common generative model for graphs with communities [44]. Hastings' method finds the community structure by maximizing the probability that the community assignment to vertices is correct according to the *planted partition model*, which is a special case of the stochastic block model, via the technique of *belief propagation* [9], [45]. Even though many real-world networks display broad vertex degree distributions, the classic stochastic block model ignores variations in the vertex degree. Karrer and Newman proposed the *degree-corrected stochastic block model*, which includes an element of heterogeneity in the degrees of the vertices [10]. Newman stated that community detection by likelihood maximization using the degree-corrected stochastic block model is equivalent to both normalized-cut graph partitioning and community detection by modularity maximization when the spectral approach is taken to split a graph into two groups [38]. Gao et al. proposed a computationally feasible algorithm for community detection in stochastic block models [11]. Their method consists of two stages: initialization and refinement. The initial community assignment obtained by some global method such as spectral clustering [17], [18] is refined through penalized local maximum likelihood estimation. The additional penalty term plays a key role in ensuring optimal performance when the community sizes are unequal and when the within-community and/or between-community edge probabilities are unequal. The most important drawback of this type of approach is the need to specify the number of communities beforehand, as this is usually unknown for real-world networks.

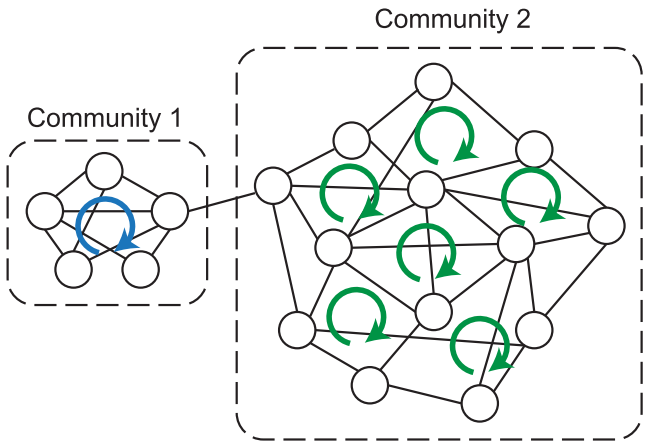


Fig. 1. Graph consisting of two communities. Circles with arrows indicate random walks.

2.6 Other Techniques

Other previous methods that do not fit into the abovementioned categories include approaches employing the *betweenness*, which expresses the importance of the role of the edges in some process, such as a random walk, on a graph [1], *label propagation*, which iteratively updates each vertex's label based on the labels of its neighbors [12], [46], and *nonnegative factorization*, which factorizes a matrix into two matrices with the property that all three matrices have no negative elements [47]. Methods of detecting overlapping communities, which have nodes belonging to multiple clusters, have often been proposed [46], [47], [48]. In this paper, we do not discuss such a community structure.

3 Restrained Random-Walk Similarity Method

The basis of the restrained random-walk similarity method is our recently proposed *random-walk similarity method* [14]. We introduce this previous approach in Section 3.1, and then describe a method of restraining random walks after an appropriate time in Section 3.2.

3.1 Random-Walk Similarity Method

We explain the basic idea of the random-walk similarity method using Fig. 1, which shows a graph consisting of two communities. Random walkers who start from the vertices in Community 1 of Fig. 1 tend to walk around in Community 1 for a period of time, because there are far fewer edges going from the vertices in Community 1 to those in Community 2 than there are among the vertices in Community 1. Similarly, random walkers who start from the vertices in Community 2 tend to walk around in Community 2 for some time. Based on this property of random walks, the random-walk similarity method clusters the starting vertices of random walks into the same communities if, after executing finite-length random walks from all vertices, there is a high degree of similarity among the sets of vertices passed by the walkers.

The detailed procedure is presented in Algorithm 1. In steps 3–12 of Algorithm 1, we execute m-step random walks

starting from all vertices $v_1,\ v_2, \ldots,\ v_N$ of the input graph, and obtain sets $S_1,\ S_2, \ldots,\ S_N$ of vertices passed by the random walkers. Ideally, the set S_i will comprise only vertices in the community including vertex v_i. However, the random walkers sometimes move from the community including the starting vertex to another community at an early stage. Then, the set S_i will include many vertices that are not in the community including vertex v_i. To prevent this, random walks starting from each vertex are repeated p times in steps 4–7, and the only vertices that are frequently passed by the walkers are added to the set S_i in steps 8–11. In steps 13–15, vertices $v_1,\ v_2, \ldots,\ v_N$ are clustered based on the Jaccard similarity coefficients among sets $S_1,\ S_2, \ldots,\ S_N$. Vertices v_i and v_j are grouped into the same cluster if $sim(S_i,\ S_j) \geq th_{sim}$. Even though $sim(S_i,\ S_j) < th_{sim}$, vertices v_i and v_j are grouped into the same cluster when a vertex v_k exists such that $sim(S_i,\ S_k) \geq th_{sim}$ and $sim(S_j,\ S_k) \geq th_{sim}$, and a vertex v_k such that $sim(S_i,\ S_k) \geq th_{sim}$ and a vertex v_l such that $sim(S_j,\ S_l) \geq th_{sim}$ are in the same cluster.

The random-walk similarity method is applicable to weighted directed graphs. When a random walker cannot move to the next vertex in a directed graph before finishing m steps in step 5 of Algorithm 1, the walk terminates at that point. This method does not detect overlapping communities.

Algorithm 1. Random-Walk Similarity Method

1: Input a graph.
2: Let the number of graph vertices be N.
3: **for** $i = 1$ to N **do**
4: **for** $j = 1$ to p **do**
5: Execute m-step random walk starting from vertex v_i.
6: Obtain a set S_{ij} of vertices through which the walker passed.
7: **end for**
8: **if** the number of times vertex v_k $(k = 1,\ 2, \ldots,\ N)$ is included in sets $S_{i1},\ S_{i2}, \ldots,\ S_{ip}$ $< p \times th_{ab}$ $(0 < th_{ab} \leq 1)$ **then**
9: Delete v_k from sets $S_{i1},\ S_{i2}, \ldots,\ S_{ip}$
10: **end if**
11: $S_i \leftarrow S_{i1} \cup S_{i2} \cup \ldots \cup S_{ip}$
12: **end for**
13: Calculate Jaccard similarity coefficients:

$$sim(S_i,\ S_j) \leftarrow \frac{|S_i \cap S_j|}{|S_i \cup S_j|}$$

$$(i = 1,\ 2, \ldots,\ N,\ j = 1,\ 2, \ldots,\ N)$$

14: Generate a community graph whose adjacency matrix is defined as

$$A = \begin{pmatrix} 0 & a_{12} & \ldots & a_{1N} \\ a_{21} & 0 & \ldots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \ldots & 0 \end{pmatrix},$$

$$a_{ij} = \begin{cases} 1 & sim(S_i,\ S_j) \geq th_{sim} \\ 0 & sim(S_i,\ S_j) < th_{sim}\ (0 < th_{sim} \leq 1) \end{cases}$$

15: Output the connected components of the community graph as the clusters of the input graph.

3.2 Restraining a Random Walk

In the random-walk similarity method, it is difficult to determine the number of steps in the random walk (m in Algorithm 1), especially for a graph including both small and large communities, as shown in Fig. 1. When m is small, the probability of a random walker moving from the community including its starting vertex to another community is low, even if the walker's initial community is small. Therefore, $sim(S_i,\ S_j)$ in step 13 of Algorithm 1 becomes high if vertices v_i and v_j are in the same small community. As a result, the random-walk similarity method can detect small communities. However, a random walker who starts from a vertex in a large community, such as Community 2 in Fig. 1, cannot walk around a sufficient number of vertices in that community when m is small. Therefore, even if v_i and v_j are in the same large community, $sim(S_i,\ S_j)$ tends to be low. As a result, large communities are often identified as several different small communities. In contrast, when m is large, $sim(S_i,\ S_j)$ tends to be high if v_i and v_j are in the same large community, resulting in v_i and v_j being successfully clustered into the same community. However, many random walkers departing from vertices in small communities move from their initial communities to other communities before finishing m steps. Then, $sim(S_i,\ S_j)$ tends to be high, even if v_i and v_j are in different communities. As a result, many vertices in small communities are incorrectly clustered and the small communities are sometimes merged into other communities; that is, the random-walk similarity method fails to identify small communities.

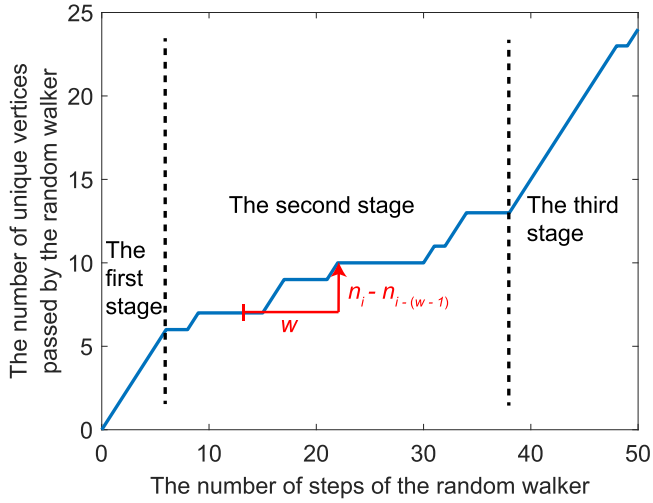


Fig. 2. Relation between the number of steps of the random walker and the number of elements in the set of vertices passed by the walker when the random walk was executed on a network constructed using the Japanese temple/shrine dataset.

To solve these problems, we devised a method of restraining a random walker when (i) the walker has already passed many vertices in the community in which it started its walk and (ii) the walker has not yet moved to another community. The random-walk similarity method in which random walks are restrained by this method successfully clusters vertices in both large and small communities when m is large.

In developing the method of restraining a random walk, we first considered the relation between the number of steps taken by the random walker and the number of unique vertices the walker passed. Fig. 2 illustrates the results for a random walk departing from a vertex in a community with 17 vertices in a network constructed using the Japanese temple/shrine dataset

(Section 4.1.3). For the first six steps, the number of unique vertices passed by the random walker is the same as the number of steps taken by the walker. This is natural, because very few vertices have been passed by the walker in that period. After that, the number of unique vertices passed by the walker gently increases until the walker's 38th step. In that period, the walker repeatedly visits the same vertices, because many vertices in the community had already been passed, including the walker's starting vertex. From the walker's 39th step, the number of unique vertices passed by the walker quickly increases again, because the walker has moved from the initial community to another community whose vertices have not yet been visited. Thus, in the first stage of a walk, a general random walker visits vertices that have not been passed with considerable frequency. In the second stage, the walker repeatedly visits vertices that have already been passed. In the third stage, the walker moves beyond the initial community.

Through the above observations, we finally reached the idea that each random walk should be terminated in the second stage, because this is the point at which the similarities among the sets of vertices passed by the random walkers departing from vertices in the same community will be highest. The procedure is described in Algorithm 2. A random walk should not be terminated as soon as the random walker visits vertices that have already been passed, because the walker sometimes revisits vertices even though it has only passed a few vertices in the initial community. In our method, a random walk is terminated when the increase in the number of unique vertices passed by the walker ($n_i - n_{i-(w-1)}$ in Fig. 2) for the number of steps taken (w in Fig. 2) is less than or equal to the threshold in steps 9–11 of Algorithm 2.

The restrained random-walk similarity method is essentially the random-walk similarity method with the random walks terminated by Algorithm 2. The method is applicable to weighted directed graphs, as is the random-walk similarity method.

Algorithm 2. Restrained Random Walk

1: Input a graph.
2: Let the maximum number of steps in a random walk be m.
3: Let the window of the number of steps in the random walk be w.
4: Let the number of elements in the set of vertices passed by the random walker until step i be n_i.
5: Select a vertex from which the random walk departs.
6: **for** $i = 2$ to m **do**
7: Move the random walker to another vertex selected at random.
8: **if** $i \geq w$ **then**
9: **if** $n_i - n_{i-(w-1)} \leq th_{pass}$ $(0 \leq th_{pass} < w)$ **then**
10: Terminate the random walk
11: **end if**
12: **end if**
13: **end for**

4 Experiments

We evaluate the accuracy of the restrained random-walk similarity method using artificially generated networks and real-world networks obtained from three datasets. We also discuss the processing speed of our method.

TABLE 1
LFR Parameters

Parameter	Description
N	Number of vertices
$\langle k \rangle$	Average degree
k_{max}	Maximum degree
μ	Mixing rate. Each vertex shares a fraction μ of its links with vertices outside its community.
γ	Power law exponent of vertex degree distribution
β	Power law exponent of community size distribution
s_{min}	Minimal community size
s_{max}	Maximal community size

4.1 Datasets

First, we introduce the networks used in the experiments. They are all undirected unweighted graphs.

4.1.1 Lancichinetti–Fortunato–Radicchi Benchmark

To test community detection methods, Lancichinetti et al. developed an algorithm for automatically generating benchmark graphs with ground-truth communities [49], and made its implementation open to the public [50]. Their algorithm is referred to as LFR (Lancichinetti–Fortunato–Radicchi) in this paper. LFR nondeterministically generates graphs characterized by the input parameters presented in Table 1.

First, we generated five graphs using LFR with the parameters $N = 1,000,\ \langle k \rangle = 15,\ k_{max} = 50,\ \mu = 0.1,\ \gamma = 2,\ \beta = 1,\ s_{min} = 20,\ s_{max} = 50$. On the basis of these parameters, we also generated five graphs with different combinations of parameters by changing N, μ, and s_{max}. The combinations of parameters are presented in Fig. 6.

4.1.2 Stanford Network Analysis Project Datasets

Leskovec and Krevl described real-world network datasets on the Stanford Network Analysis Platform (SNAP) [51]. Harenberg et al. used five undirected, unweighted graphs with ground-truth communities in the SNAP datasets to compare various community detection methods [13].

- Amazon: This graph was constructed by crawling the Amazon website. If a product i is frequently co-purchased with product j, the graph contains an undirected edge from i to j. Each product category provided by Amazon defines each ground-truth community.
- DBLP: The DBLP computer science bibliography provides a comprehensive list of research papers in computer science. SNAP provides the co-authorship graph in which two authors are connected if they have published at least one paper together. Authors who published in a certain journal or conference form a community.
- LiveJournal: LiveJournal is a free online blogging community in which users declare friendship with each other. Users can also form a group that other members can join. SNAP provides the graph in which nodes represent users and edges represent

TABLE 2
Summary of the SNAP Graphs

Statistic	Amazon	DBLP	LiveJournal	Orkut	YouTube
Number of vertices	27	298	4,479	525	2,296
Number of edges	82	1,168	143,878	8,709	5,934
Number of communities	3	32	118	5	392
Minimum community size	4	6	9	69	2
Maximum community size	18	26	407	211	29
Mean mixing rate	0.029	0.040	0.018	0.002	0.257

friendship between users. User-defined groups are considered as ground-truth communities.

- Orkut: Orkut is a free online social network in which users form friendships with each other. Orkut also allows users to form a group that other members can join. The graph and its ground-truth communities are defined in the same way as for LiveJournal.
- YouTube: YouTube is a video-sharing website that includes a social network. In the YouTube social network, users form friendships with each other and create groups that others can join. The graph and its ground-truth communities are defined in the same way as for LiveJournal.

Harenberg et al. used only the top 5,000 ground-truth communities for their evaluation; these communities were determined by Yang and Leskovec [52]. In addition, they ranked the top 5,000 ground-truth communities based on internal density, which is the ratio of actual edges to the number of possible edges, and removed the bottom quartile. They also eliminated duplicate communities. They removed all the nodes and incident edges that did not belong to any of the remaining communities from the graphs. The resulting graphs had overlapping ground-truth communities. To test disjoint community detection methods, they found the maximum independent set of disjoint ground-truth communities and removed nodes and their incident edges that did not belong to any of these communities.

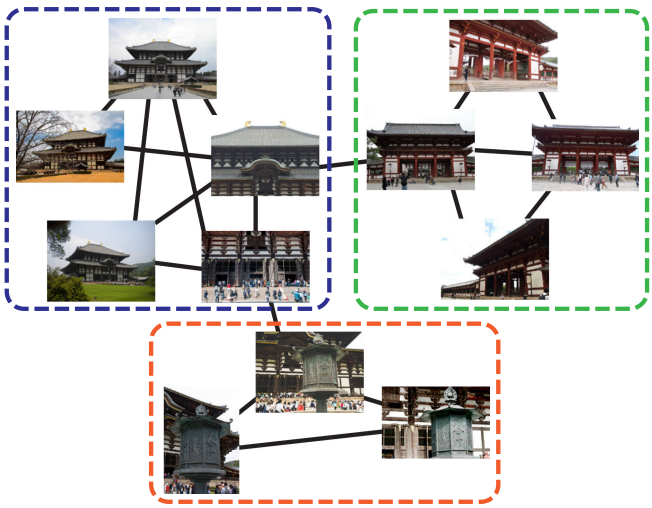


Fig. 3. Image of a match graph constructed to cluster images of tourist attractions by subject. The copyrights of the pictures are held by, from the left, Chang Tai Jyun, yannickdellafaille, rurinoshima, KimonBerlin, Jexweber.fotos, *_*, timothybrennan, RachelH_, ohsarahrose, Lora Sutyagina, RachelC, and nakashi.

In our experiments, we used the largest connected components of the graphs provided by Harenberg [53]. The graph statistics are presented in Table 2.

4.1.3 Japanese Temple/Shrine Image Dataset

We have used match graphs [54] to evaluate the random walk similarity method [14]. A match graph is one in which the vertices are images and the edges represent the existence of matching key points [55] between two images. Fig. 3 shows a match graph constructed from tourist attraction images. Each connected component of a match graph basically consists of images in which a particular subject appears. However, some connected components have images whose primary subjects are different, as shown in Fig. 3, because matching key points are occasionally found between two images, even though the primary subjects of those images are different. This may be because the images have similar textures, as in Fig. 4a, or because one image has a secondary subject that is a primary subject for another image, as shown in Fig. 4b. In such connected components, the edges among images whose primary subjects are the same are dense and the edges among images

(a) Images having similar textures. Left: The Great Buddha Hall of the Todai-ji Temple (© *_*). Right: The Chumon Gate of the Todai-ji Temple (© RachelH_).

(b) Another subject exists behind the primary subject. Left: The Great Buddha Hall of the Todai-ji Temple (© Jexweber.fotos). Right: The gilt-bronze octagonal lantern of the Todai-ji Temple (© timothybrennan).

Fig. 4. Examples of matching SIFT key points [55] between images with different primary subjects.

TABLE 3
Summary of the Connected Components of the Match Graph Constructed from the Japanese Temple/Shrine Image Dataset

Statistic	CC1	CC2
Number of vertices	623	396
Number of edges	7,979	2,102
Number of communities	10	8
Minimum community size	1	1
Maximum community size	500	276
Mean mixing rate	0.037	0.027

TABLE 4
Summary of the Connected Components of Match Graphs Constructed with the Paris 500k Dataset

Statistic	CC3	CC4
Number of vertices	25,067	243
Number of edges	181,631	1,466
Number of communities	76	2
Minimum community size	1	74
Maximum community size	6,162	169
Mean mixing rate	0.052	0.026

whose primary subjects are different are sparse. Therefore, clusters consisting of images whose primary subjects are the same are obtained by applying a community detection method to a connected component consisting of images whose primary subjects are different. The technique of clustering images by subject is important because it is useful in a wide range of applications, including reconstructing 3-D scenes [54] and image recognition [56].

By searching Flickr [57], an online photo management and sharing application, we obtained 4,015 Creative-Commons-licensed images of the Todai-ji Temple using the search term $todaiji$, 3,808 Creative-Commons-licensed images of the Nikko Tosho-gu Shrine using $toshogu$, and 1,102 Creative-Commons-licensed images of the Horyu-ji Temple using $horyuji$. We manually labeled those images whose primary subjects were considered to be popular with the corresponding subject names. This dataset, which we call the $Japanese\ temple/shrine\ dataset$, will be made available to the public in the near future. Based on the work of Agarwal et al. [54], we constructed a match graph using 3,039 images with the primary subject name labels. We used the largest and the second-largest connected components of the match graph to evaluate the community detection methods. We considered the primary subject names of the images to be the ground-truth communities. The statistics of the two connected components (CC1 and CC2) are presented in Table 3.

4.1.4 Paris 500k Dataset

Weyand and Leibe studied popular views of landmark buildings and other objects in image collections [58] and visual landmark recognition [56] with the $Paris\ 500k\ dataset$ [59], which was constructed with 501,356 images of Paris collected from Flickr and Panoramio, an online photo sharing service. We evaluated the community detection methods by applying them to the connected components of the match graph constructed from the Paris 500k dataset.

We obtained the downloadable images of the Paris 500k dataset from Flickr and Panoramio using the download programs developed by Leibe [59], and constructed the match graph using 84,155 images with subject identifiers. We used the largest and the second-largest connected components of the match graph to evaluate the community detection methods. We considered the subject identifiers to be the ground-truth communities. The statistics of the two connected components (CC3 and CC4) are presented in Table 4.

4.2 Parameter Settings

The random-walk similarity method has four parameters: the number of steps in a random walk (m in Algorithm 1), the number of executions of a random walk starting from each vertex (p in Algorithm 1), the threshold for determining whether the random walker accidentally passed a vertex (th_{ab} in Algorithm 1), and the threshold for determining whether sets of vertices passed by random walkers are similar (th_{sim} in Algorithm 1). The restrained random-walk similarity method has two additional parameters: the window of the number of steps in a random walk (w in Algorithm 2) and the threshold for determining whether a random walker is continuously revisiting vertices that have already been passed (th_{pass} in Algorithm 2).

We determined all parameter values, except p, using the pattern search optimization method [60]. We used the normalized mutual information (NMI) [61] between the ground-truth communities and the communities found by the restrained random-walk similarity method as the objective function.

The NMI is based on defining a confusion matrix $\boldsymbol{N}$, where the rows correspond to the real communities and the columns correspond to the detected communities. Each element of $\boldsymbol{N}$, N_{ij}, is the number of vertices that appear in both the real community i and the detected community j. In the field of information theory, the NMI $I(A,B)$, which is a measure of similarity between the partitions, is defined as

$$I(A,B) = \frac{-2\sum_{i=1}^{c_A}\sum_{j=1}^{c_B} N_{ij}\log\left(\frac{N_{ij}N}{N_{i\cdot}N_{\cdot j}}\right)}{\sum_{i=1}^{c_A} N_{i\cdot}\log\left(\frac{N_{i\cdot}}{N}\right)+\sum_{j=1}^{c_B} N_{\cdot j}\log\left(\frac{N_{\cdot j}}{N}\right)}, \tag{1}$$

where c_A is the number of real communities, c_B is the number of detected communities, $N_{i\cdot}$ is the sum over row i of confusion matrix $\boldsymbol{N}$, $N_{\cdot j}$ is the sum over column j, and N is the number of vertices in a graph.

The NMI ranges from 0 to 1. The better a community detection method performs with respect to the ground truth, the higher the value of $I(A,B)$. If the detected communities are identical to the ground-truth communities, then $I(A,B)$ takes its maximum value of 1. If the detected communities are totally independent of the ground-truth communities, for example when the entire graph is found to be one community, $I(A,B)=0$.

When we actually use the restrained random-walk similarity method, the ground-truth communities are unknown. Therefore, in real scenes, we cannot determine the parameters for this method using a pattern search optimization method, as the objective function is the NMI between the ground-truth communities and the communities detected by the restrained random-walk similarity method. However, to evaluate the superior performance of our

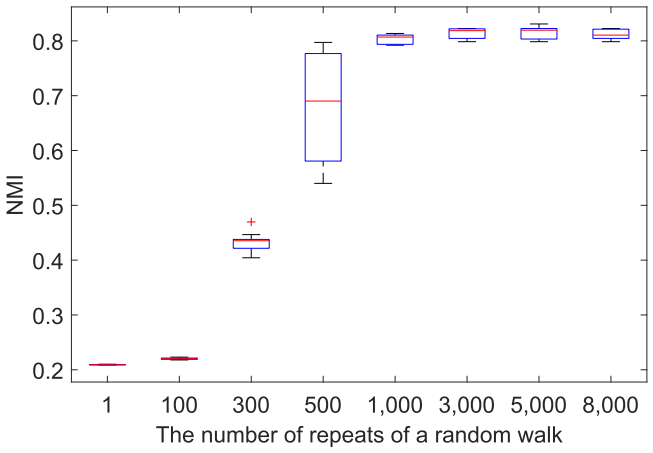


Fig. 5. Accuracy of the restrained random-walk similarity method with respect to the number of repeats of a random walk starting from each vertex (p in Algorithm 1) for CC1 in the match graph constructed with the Japanese temple/shrine dataset (Section 4.1.3). Accuracies are shown in the boxplots [62] of the normalized mutual information (NMI).

community detection method in the experiments, we obtained the optimum parameters using such an optimization method. To ensure equality, we also determined the parameters of previous community detection methods, Spin-glass [8] and CD-TRandwalk [6], using a pattern search optimization method whose objective function was the NMI between the ground-truth communities and the communities detected by each method.

The higher the value of p in Algorithm 1, the better the accuracy of the restrained random-walk similarity method. Fig. 5 presents the relation between the number of executions of random walks p and the NMI when we applied the restrained random-walk similarity method to CC1 of the match graph constructed with the Japanese temple/shrine dataset. The restrained random-walk similarity method is nondeterministic, and so we executed it 10 times for each value of p and present the NMI in the boxplots [62] in Fig. 5. All parameters except p were then determined using pattern search optimization. Fig. 5 indicates that the NMIs were mostly saturated when p was set to 5,000. Thus, in our experiments, p was fixed to 5,000. Fig. 5 also indicates that the condition whereby random walks departing from each vertex are executed p times and exceptional vertices (which are rarely passed by the walkers) are excluded from the set of vertices passed by the walkers (S_i of Algorithm 1) is essential for the restrained random-walk similarity method.

4.3 Accuracy

We compared the accuracy of the restrained random-walk similarity method with that of Reichardt and Bornholdt's method [8], which we call *Spin-glass*, Blondel et al.'s method [23], which we call *Louvain*, Infomap [41], CD-TRandwalk [6], and CONCLUDE [7]. Fan et al. stated that Spin-glass (Section 2.3) and the method relying on modularity maximization with extremal optimization (Section 2.2) outperform the method using the betweenness (Section 2.6) on weighted graphs [63]. The experiments of Lancichinetti and Fortunato showed that Infomap (Section 2.4) outperforms many traditional methods in terms of accuracy and that Louvain (Section 2.2) also performs very well [64]. Furthermore, those methods have a low computational complexity [64]. Fu et al. stated that CD-TRandwalk (Section 2.4) is superior to some previous methods in terms of the quality of community detection and run time; moreover, CD-TRandwalk can adapt to imbalanced networks, which have both small communities with few vertices and large communities with thousands of vertices [6]. CONCLUDE (Section 2.4) successfully mitigates the resolution limit effect [7].

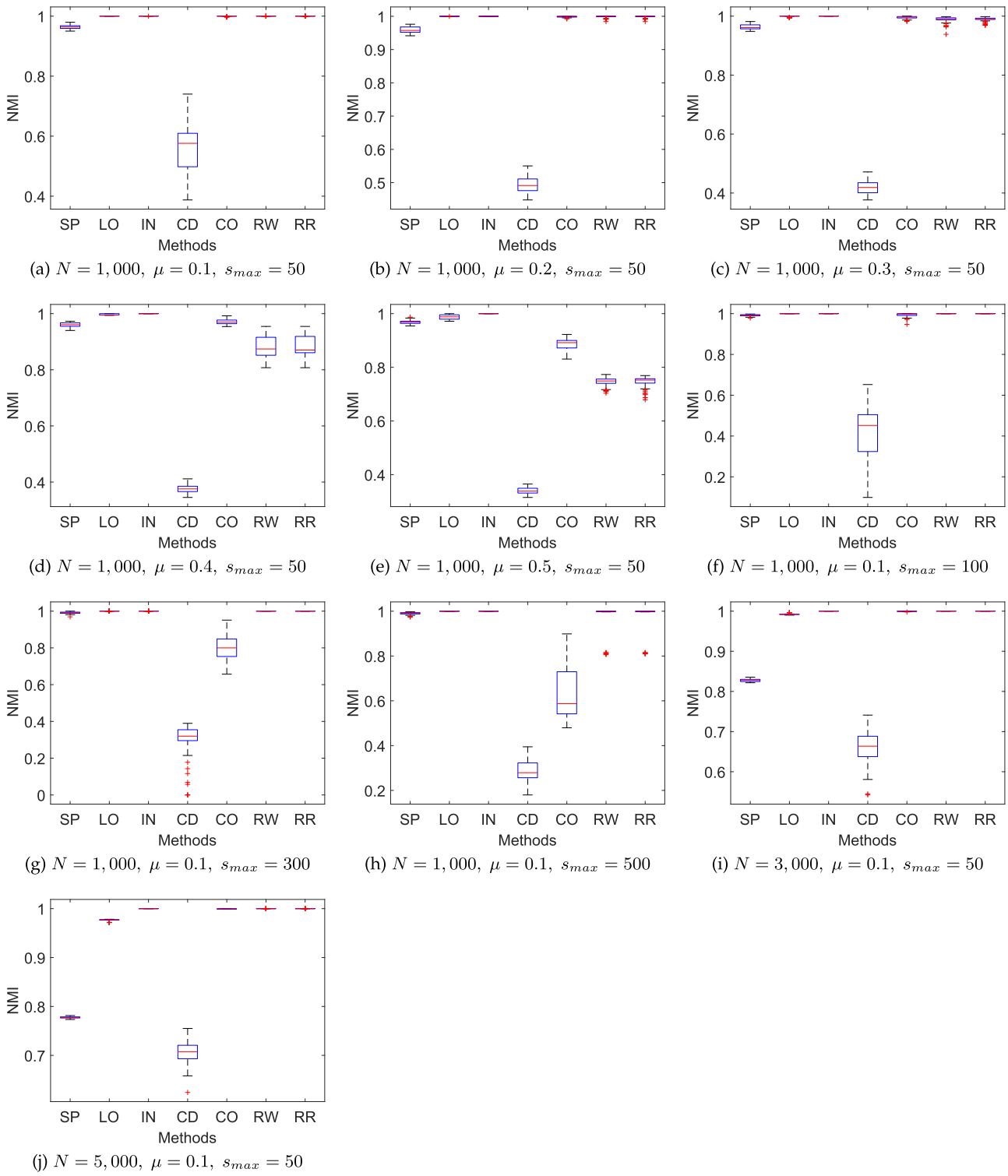
All methods except Louvain are nondeterministic, and so we executed each algorithm 10 times for each network after the parameters had been determined as described in Section 4.2. Figs. 6, 7, 8, and 9 present the NMIs of the communities found by the community detection methods with respect to the ground-truth communities in the boxplots for each network. These figures also show the NMIs of the random-walk similarity method to confirm the effect of restraining the random walks, as described in Section 3.2. The parameters of the random-walk similarity method were also determined using pattern search optimization. As LFR is a nondeterministic algorithm, we generated five networks using LFR with each parameter combination. Therefore, each graph in Fig. 6 presents the results of 50 trials with each community detection method. The test results are as follows.

Fig. 6 presents the experimental results for the LFR networks. They show that the random-walk similarity and the restrained random-walk similarity were very accurate in terms of finding community structures in networks consisting of communities of various sizes (Figs. 6f, 6g, and 6h) and in networks of various sizes (Figs. 6i and 6j). However, our methods did not work well with networks that have high mixing rates μ (Figs. 6d and 6e).

In the random-walk similarity method, the expectation of the number of steps until a walker transfers from the initial community to another community is calculated by the following equations:

$$\begin{aligned} E &= \sum_{k=1}^{\infty} k\mu(1-\mu)^{k-1} \\ &= \sum_{k=1}^{\infty} k(1-r)r^{k-1} \quad (r := 1-\mu) \\ &= (1-r)\sum_{k=1}^{\infty} kr^{k-1} \\ &= (1-r)\frac{1}{(1-r)^2} \\ &= \frac{1}{\mu}, \end{aligned} \tag{2}$$

where k is the current number of steps in the random walk and μ is the mixing rate of LFR (Table 1). When $\mu = 0.5$, random walkers move out of the initial communities on their second step, on average. When random walkers leave the initial communities so early, the similarities among sets of vertices passed by random walkers departing from vertices in the same community will not become high, and therefore the random-walk similarity method does not work well for such networks. Moreover, the method of restraining a random walk using Algorithm 2 is not effective for networks of this type. That is, the restrained random-walk similarity method is not applicable to networks with obscure community structures.



(a) $N = 1,000,\ \mu = 0.1,\ s_{max} = 50$ (b) $N = 1,000,\ \mu = 0.2,\ s_{max} = 50$ (c) $N = 1,000,\ \mu = 0.3,\ s_{max} = 50$

(d) $N = 1,000,\ \mu = 0.4,\ s_{max} = 50$ (e) $N = 1,000,\ \mu = 0.5,\ s_{max} = 50$ (f) $N = 1,000,\ \mu = 0.1,\ s_{max} = 100$

(g) $N = 1,000,\ \mu = 0.1,\ s_{max} = 300$ (h) $N = 1,000,\ \mu = 0.1,\ s_{max} = 500$ (i) $N = 3,000,\ \mu = 0.1,\ s_{max} = 50$

(j) $N = 5,000,\ \mu = 0.1,\ s_{max} = 50$

Fig. 6. Accuracy of the community detection methods for the LFR benchmark graphs (Section 4.1.1). SP: Spin-glass [8], LO: Louvain [23], IN: Infomap [41], CD: CD-TRandwalk [6], CO: CONCLUDE [7], RW: Random-walk Similarity Method (Ours), RR: Restrained Random-walk Similarity Method (Ours); NMI: Normalized Mutual Information; N: Number of vertices, μ: Mixing rate, s_{max}: Maximal community size.

In Fig. 6, we cannot confirm the effectiveness of restraining the random walks, because the NMIs of the random-walk similarity method are close to one, except for those in Figs. 6d and 6e. It seems that the random-walk similarity method can easily find community structures in networks generated by LFR when the mixing rate μ is low.

In the experiments with the LFR networks, Spin-glass and Louvain did not work well with large networks (Figs. 6i and 6j). Infomap worked well overall, but CD-TRandwalk was the worst performer. CONCLUDE was especially weak with networks consisting of various community sizes (Figs. 6g and 6h).

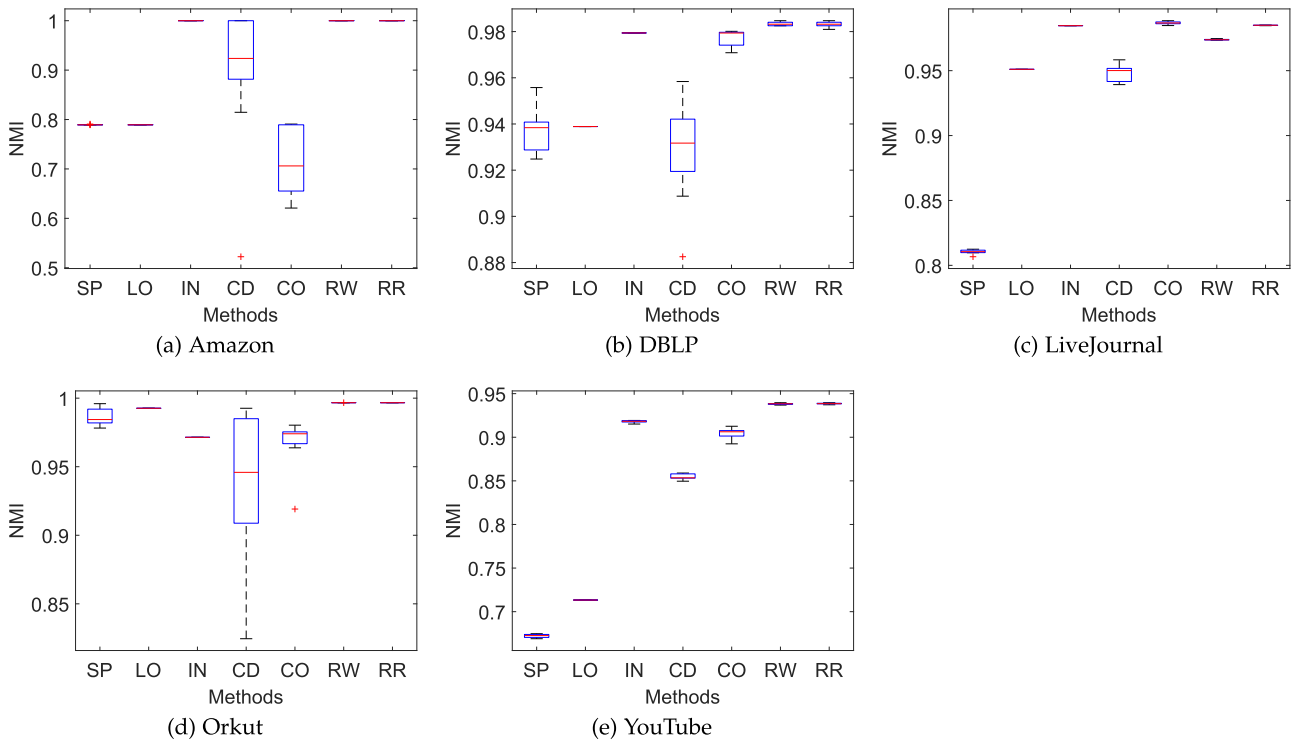


Fig. 7. Accuracy of the community detection methods for the SNAP datasets (Section 4.1.2). SP: Spin-glass [8], LO: Louvain [23], IN: Infomap [41], CD: CD-TRandwalk [6], CO: CONCLUDE [7], RW: Random-walk Similarity Method (Ours), RR: Restrained Random-walk Similarity Method (Ours); NMI: Normalized Mutual Information.

Fig. 7 shows the experimental results using the SNAP networks. Overall, the NMIs of the restrained random-walk similarity method tend to be higher than those of the previous methods. In Fig. 7c, the difference between the median NMI of the random-walk similarity method and that of the restrained random-walk similarity method is 0.011. Thus, restraining the random walks of the random-walk similarity method is an effective approach.

Fig. 8 presents the results of applying the community detection methods to CC1 and CC2 of the match graph constructed from the Japanese temple/shrine image dataset. The restrained random-walk similarity method overwhelmingly outperformed the previous methods in terms of accuracy. Moreover, restraining the random walks was very effective. The difference in median NMI between the random-walk similarity method and the restrained random-walk similarity method was 0.057 for CC1 (Fig. 8a) and 0.100 for CC2 (Fig. 8b).

Fig. 9 presents the results of the tests for CC3 and CC4 of the match graph constructed from the Paris 500k dataset. When we attempted to determine the CD-TRandwalk parameters using the pattern search method, the first trial did not finish within 24 h. Therefore, we stopped calculating the NMIs of CD-TRandwalk for CC3, because the pattern search method must execute CD-TRandwalk many times to determine the optimum parameters and will require an enormous amount of time to complete this procedure.

From Fig. 9a, we can see that the restrained random-walk similarity method outperforms the previous methods.

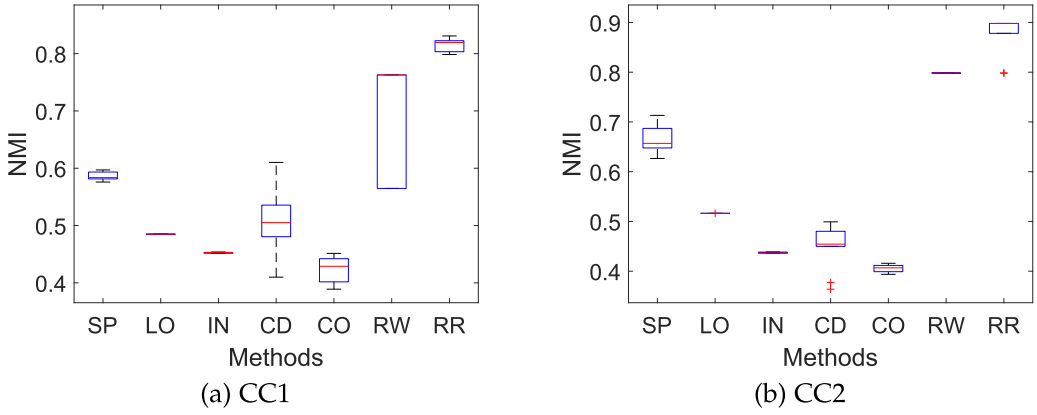


Fig. 8. Accuracy of the community detection methods for the match graph constructed from the images in the Japanese temple/shrine image dataset (Section 4.1.3). SP: Spin-glass [8], LO: Louvain [23], IN: Infomap [41], CD: CD-TRandwalk [6], CO: CONCLUDE [7], RW: Random-walk Similarity Method (Ours), RR: Restrained Random-walk Similarity Method (Ours); NMI: Normalized Mutual Information.

TABLE 5
Analysis of the Communities Obtained by Applying Each Community Detection Method to CC2 in the Match Graph Constructed from the Japanese Temple/Shrine Image Dataset

Community name	Community size	SP	LO	IN	CD	CO	RW	RR
Birushana Buddha statue	276	5	8	19	7	24	1	1
Virupaksa statue	84	1	2	3	5	6	1	1
Nyoirin Kannon status	26	2	1	5	0	7	0	1
Kokuzo Bosatsu statue	4	1	1	2	0	1	0	0
The halo of Birushana Buddha	3	0	0	0	0	0	0	0
Vaisravana statue	1	0	0	0	0	0	0	0
Agyo statue	1	0	0	0	0	0	0	0
Nigatsu-do Hall	1	0	0	0	0	0	0	0

The community name signifies the name of the primary subject of the images in each ground-truth community. The community size represents the number of elements in each ground-truth community. The number of communities detected by each method is presented for every subject. SP: Spin-glass [8], LO: Louvain [23], IN: Infomap [41], CD: CD-TRandwalk [6], CO: CONCLUDE [7], RW: Random-walk Similarity Method (Ours), RR: Restrained Random-walk Similarity Method (Ours).

Additionally, the effect of restraining the random walks is clearly confirmed, with a difference of 0.028 in the median NMI. In Fig. 9b, the median NMI of CD-TRandwalk is higher than that of our method. However, the NMIs of CD-TRandwalk are unstable.

In summary, the restrained random-walk similarity method does not work well with networks in which the average mixing rate μ is high. However, in terms of accuracy, the proposed method overwhelmingly outperforms previous methods on networks in which the average mixing rate is low. Additionally, our method of restraining random walks is effective for some real-world networks. Infomap worked well on the LFR networks, but is clearly inferior to our method when applied to the more complex real-world networks. Fig. 9b shows that the median NMI of CD-TRandwalk is higher than that of the restrained random-walk similarity method. However, our method is far superior to CD-TRandwalk on the other networks.

4.4 Analysis of Detected Communities

Table 5 presents the name of the primary subject of the images in each ground-truth community of CC2 in the match graph built from the Japanese temple/shrine image dataset, the number of elements in each ground-truth community, and the number of communities found by each community detection method for every subject in the trial that obtained each method's fifth highest NMI. Example images from each ground-truth community are shown in Fig. 10.

The previous methods tended to incorrectly detect each community as several different small communities. For example, Spin-glass detected five "Birushana Buddha statue" communities, even though they should ideally have been detected as one community. The restrained random-walk similarity method did not make such a mistake.

The restrained random-walk similarity method detected the "Birushana Buddha statue" community, the "Virupaksa statue" community, and the "Nyoirin Kannon statue" community, which was not found by the random-walk similarity method. In the random-walk similarity method, the pattern search optimization set the number of steps m in each random walk to 102. This value proved appropriate for detecting the "Birushana Buddha statue" community, which has a size of 276, and the "Virupaksa statue" community, which has a size of 84, but was too large to detect the small "Nyoirin Kannon statue" community, which has a size of just 26.

In contrast, m was set to 134 in the restrained random-walk similarity method. Although larger than the value used in the random-walk similarity method, the random walks of the restrained random-walk similarity method were effectively terminated before 134 steps. Fig. 11 shows histograms of the number of steps in restrained random-walks on CC2. The number of steps in random walks departing from vertices in the small

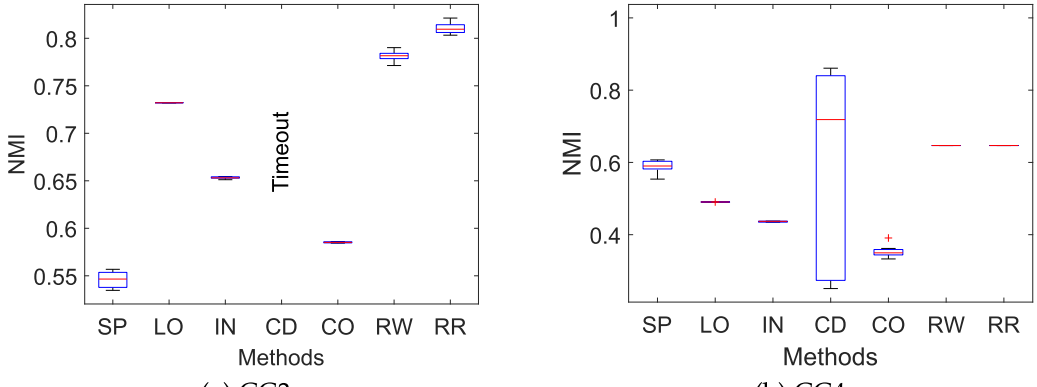
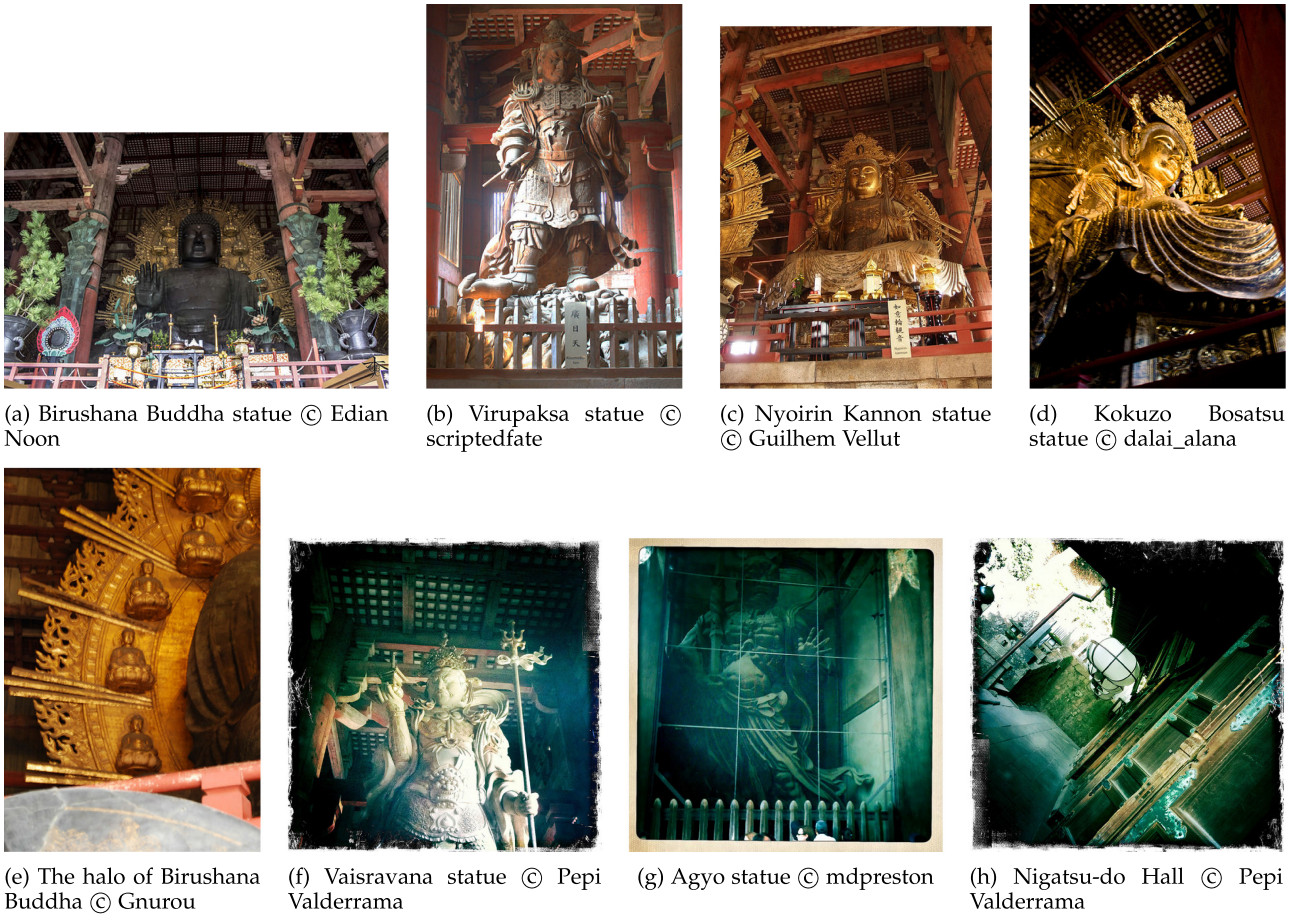


Fig. 9. Accuracy of the community detection methods for the match graph constructed from the images in the Paris 500k dataset (Section 4.1.4). SP: Spin-glass [8], LO: Louvain [23], IN: Infomap [41], CD: CD-TRandwalk [6], CO: CONCLUDE [7], RW: Random-walk Similarity Method (Ours), RR: Restrained Random-walk Similarity Method (Ours); NMI: Normalized Mutual Information.



(a) Birushana Buddha statue © Edian Noon

(b) Virupaksa statue © scriptedfate

(c) Nyoirin Kannon statue © Guilhem Vellut

(d) Kokuzo Bosatsu statue © dalai_alana

(e) The halo of Birushana Buddha © Gnurou

(f) Vaisravana statue © Pepi Valderrama

(g) Agyo statue © mdpreston

(h) Nigatsu-do Hall © Pepi Valderrama

Fig. 10. Example images in the CC2 communities of the match graph obtained from the Japanese temple/shrine image dataset.

"Nyoirin Kannon statue" community (Fig. 11b) was restrained to be much lower than that of random walks departing from vertices in the large "Birushana Buddha statue" community (Fig. 11a). As a result, the restrained random-walk similarity method successfully detected both the large "Birushana Buddha statue" and "Virupaksa statue" communities, and the small "Nyoirin Kannon statue" community. This indicates that our method of restraining the random walkers works well.

However, the restrained random-walk similarity method could not detect tiny communities with sizes of four nodes or less. In a network including small and large communities, such as CC2 in the Japanese temple/shrine match graph, the number of steps m in the random walks must be sufficiently large to successfully detect large communities. However, random walkers that depart from tiny communities soon move out of the initial areas. Then, the similarities between sets of vertices passed by random walkers departing from vertices in tiny communities and other communities tend to be high, and as a result, the vertices in tiny communities are incorporated into other communities. Moreover, the method of restraining random walks does not work well in such tiny communities, because the random walkers stay there for too short a time.

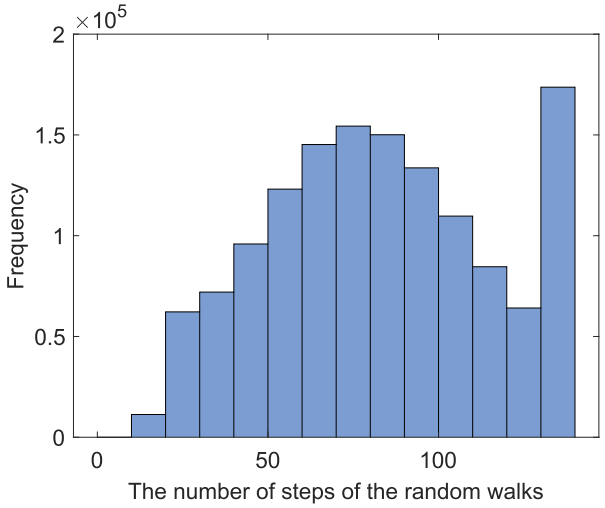
4.5 Processing Speed

Recently, a number of applications in various fields have sought to manipulate large networks. Therefore, it is important to discuss the processing speed of our method.

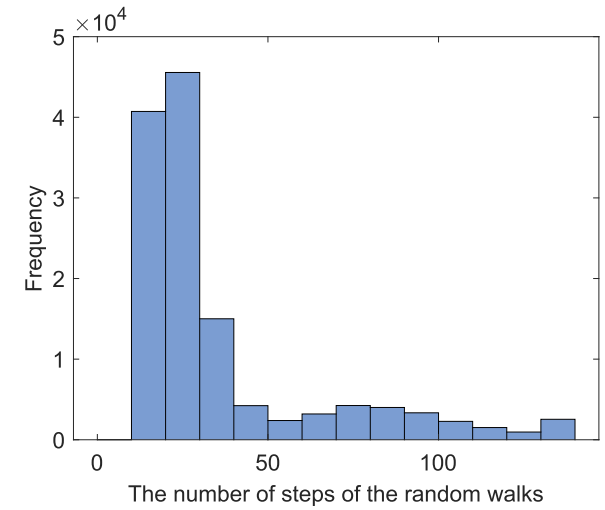
Algorithm 1 requires considerable computation time in steps 3–12 because of the large number of random walks that are executed. The restrained random-walk similarity method executes a maximum m-step random walk Np times. In the case of graph CC1, m = 116, N = 623, and p = 5,000. Therefore, a maximum 116-step random walk is executed 3,115,000 times.

Such a huge number of random walks can be executed completely in parallel. We succeeded in executing the restrained random-walk similarity method at high speed by processing random walks in parallel on a GPU. We used MATLAB to run the restrained random-walk similarity method on an LFR network with 1,000 vertices by sequentially processing the random walks through the CPU (Intel Xeon CPU E5-2699 v4) and by processing the random walks in parallel on a GPU (NVIDIA Quadro P6000). The results in Fig. 12 show that the GPU required approximately 0.5 percent of the runtime of the CPU.

Fig. 13 shows the runtimes of the community detection methods on LFR networks with 1,000, 3,000, and 5,000 vertices. We executed Spin-glass, Louvain, and Infomap with C implementations of igraph [65], [66]. We reproduced and executed CD-TRandwalk using MATLAB. CONCLUDE was executed using a Java implementation provided by E. Ferrara [67]. We implemented and executed the random-walk similarity method and the restrained random-walk similarity method in MATLAB, with the random walks executed in parallel on the GPU. Fig. 13 indicates that Louvain



(a) Restrained random-walks starting from vertices in the "Birushana Buddha statue" community.



(b) Restrained random-walks starting from vertices in the "Nyoirin Kannon statue" community.

Fig. 11. Histograms of the number of steps in restrained random-walks in the CC2 communities of the match graph obtained from the Japanese temple/shrine image dataset.

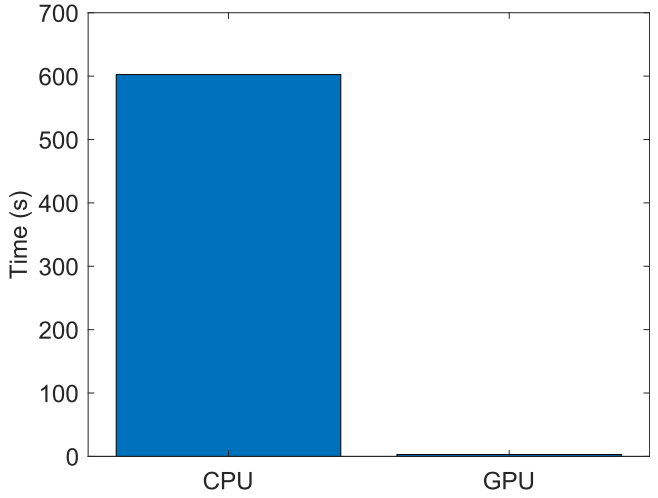


Fig. 12. Runtimes of the restrained random-walk similarity method on the LFR benchmark graph with 1,000 vertices (Section 4.1.1) when the random walks were executed sequentially on the CPU and in parallel on the GPU. Each bar shows the mean of 10 runtimes.

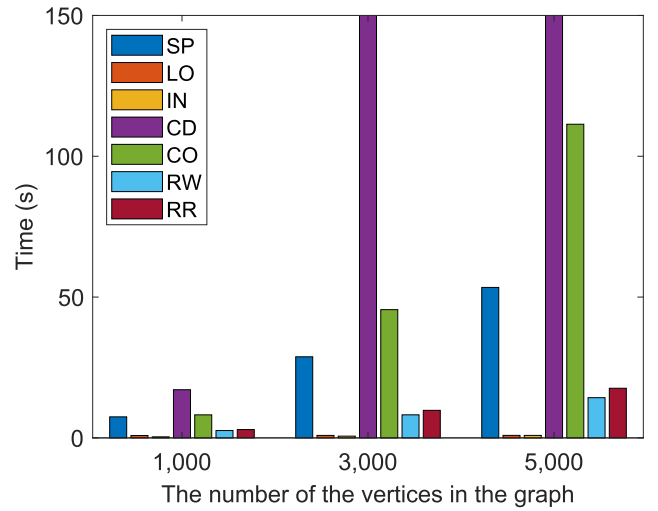


Fig. 13. Runtime comparison of the community detection methods on the LFR benchmark graphs (Section 4.1.1). Each bar shows the mean of 10 runtimes. The runtimes of CD-TRandwalk on the LFR graphs with 3,000 and 5,000 vertices were greater than 150 s. SP: Spin-glass [8], LO: Louvain [23], IN: Infomap [41], CD: CD-TRandwalk [6], CO: CONCLUDE [7], RW: Random-walk Similarity Method (Ours), RR: Restrained Random-walk Similarity Method (Ours).

and Infomap were quite fast, although the random-walk similarity method and the restrained random-walk similarity method significantly outperformed the other methods in terms of processing speed. The random walks in the restrained random-walk similarity method consisted of fewer steps than those in the random-walk similarity method. However, the runtime of the restrained random-walk similarity method was slightly longer than that of the random-walk similarity method because the restrained random-walk similarity method performs additional operations to determine whether the random walks should be terminated at each step.

5 Conclusion and Future Works

We have proposed a novel community detection technique called the restrained random-walk similarity method. Our approach clusters the starting vertices of random walks on a graph based on the similarities among sets of vertices passed by the walkers. Experiments demonstrated that our method is much more accurate than previous methods on graphs with low mixing rates. In particular, our method is overwhelmingly superior in terms of accuracy on real-world networks. The restrained random-walk similarity method executes an enormous number of random walks. However, the experiments also showed that it stands comparison with the previous methods when the random walks are executed in parallel on a GPU.

The restrained random-walk similarity method has six parameters. In this paper, we have not discussed the policy for determining them on a graph whose community structure is unknown. Some relations between the accuracy and these parameters in real-world networks are presented in the appendix, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/2926033. The accuracy of our method does not

seem to be sensitive to variations in each parameter. It appears to work well on any graph when $p = 5,000$ and $th_{ab} = 0.2$. Therefore, we do not think that it is too difficult to set appropriate parameter values in our method. In future work, we will analyze the relation between the parameters of the restrained random-walk similarity method and the structure of networks in more detail and clarify an efficient means of determining the parameters. In addition, we will confirm that the restrained random-walk similarity method finds the community structure of directed weighted graphs more accurately than previous methods.

Acknowledgments

We thank Stuart Jenkinson, PhD, from Edanz Group (www.edanzediting.com/ac) for editing a draft of this manuscript.

References

[1] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Phys. Rev. E*, vol. 69, no. 2, pp. 026 113–026 113–15, 2004.
[2] S. Fortunato, "Community detection in graphs," *Phys. Reports*, vol. 486, pp. 75–174, 2010.
[3] S. Fortunato and D. Hric, "Community detection in networks: A user guide," *Phys. Reports*, vol. 659, pp. 1–44, 2016.
[4] S. Fortunato and M. Barthelemy, "Resolution limit in community detection," in *Proc. National Academy Sci. United States America*, vol. 104, no. 1, 2007, pp. 36–41.
[5] B. D. Hughes, *Random Walks and Random Environments*. Oxford, U. K.: Clarendon Press, 1995.
[6] X. Fu, C. Wang, Z. Wang, and Z. Ming, "Threshold random walks for community structure detection in complex networks," *J. Softw.*, vol. 8, no. 2, pp. 286–295, 2013.
[7] P. D. Meo, E. Ferrara, G. Fiumara, and A. Provetti, "Mixing local and global information for community detection in large networks," *J. Comput. Syst. Sci.*, vol. 80, no. 1, pp. 72–87, 2014.
[8] J. Reichardt and S. Bornholdt, "Statistical mechanics of community detection," *Phys. Rev. E*, vol. 74, no. 1, pp. 016 110–016 110–14, 2006.
[9] M. B. Hastings, "Community detection as an inference problem," *Phys. Rev. E*, vol. 74, no. 3, pp. 035 102–1–035 102-4, 2006.
[10] B. Karrer and M. E. J. Newman, "Stochastic blockmodels and community structure in networks," *Phys. Rev. E*, vol. 83, no. 1, pp. 016 107–1–016 107–10, 2011.
[11] C. Gao, Z. Ma, A. Y. Zhang, and H. H. Zhou, "Achieving optimal misclassification proportion in stochastic block models," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 1980–2024, 2017.
[12] 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, pp. 036 106–1–036 106–11, 2007.
[13] 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," *WIREs Comput. Statist.*, vol. 6, pp. 426–439, 2014.
[14] M. Okuda, S. Satoh, S. Iwasawa, S. Yoshida, Y. Kidawara, and Y. Sato, "Community detection using random-walk similarity and application to image clustering," in *Proc. IEEE Int. Conf. Image Process.*, 2017, pp. 1292–1296.
[15] A. V. Goldberg and R. E. Tarjan, "A new approach to the maximum-flow problem," *J. ACM*, vol. 35, no. 4, pp. 921–940, 1988.
[16] L. R. Ford and D. R. Fulkerson, "Maximal flow through a networks," *Can. J. Math.*, vol. 8, pp. 399–404, 1956.
[17] L. Hagen and A. B. Kahng, "New spectral methods for ratio cut partitioning and clustering," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 11, no. 9, pp. 1074–1085, Sep. 1992.
[18] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.
[19] M. E. J. Newman, "Fast algorithm for detecting community structure in networks," *Phys. Rev. E*, vol. 69, no. 6, pp. 066 133–1–066 133-5, 2004.
[20] A. Clauset, M. E. J. Newman, and C. Moore, "Finding community structure in very large networks," *Phys. Rev. E*, vol. 70, no. 6, pp. 066 111–1–066 111-6, 2004.
[21] L. Danon, A. Díaz-Guilera, and A. Arenas, "The effect of size heterogeneity on community identification in complex networks," *J. Statistical Mech.: Theory Experiment*, vol. 2006, no. 11, p. P11010, 2006.
[22] P. Schuetz and A. Caflisch, "Multistep greedy algorithm identifies community structure in real-world and computer-generated networks," *Phys. Rev. E*, vol. 78, no. 2, pp. 026 112–1–026 112-7, 2008.
[23] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *J. Statistical Mech.: Theory Experiment*, vol. 2008, no. 10, p. P10008, 2008.
[24] S. Kirkpatrick, J. C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Sci.*, vol. 220, no. 4598, pp. 671–680, 1983.
[25] R. Guimerà, M. Sales-Pardo, and L. A. N. Amaral, "Modularity from fluctuations in random graphs and complex networks," *Phys. Rev. E*, vol. 70, no. 2, pp. 025 101–1–025 101-4, 2004.
[26] R. Guimerà and L. A. N. Amaral, "Functional cartography of complex metabolic networks," *Nature*, vol. 433, no. 7028, pp. 895–900, 2005.
[27] C. P. Massen and J. P. K. Doye, "Identifying community within energy landscapes," *Phys. Rev. E*, vol. 71, no. 4, pp. 046 101–1–046 101–12, 2005.
[28] A. Medus, G. A. na, and C. O. Dorso, "Detection of community structures in networks via global optimization," *Physica A: Statistical Mech. Appl.*, vol. 358, pp. 593–604, 2005.
[29] S. Boettcher and A. G. Percus, "Optimization with extremal dynamics," *Phys. Rev. Lett.*, vol. 86, no. 23, pp. 5211–5214, 2001.
[30] J. Duch and A. Arenas, "Community detection in complex networks using extremal optimization," *Phys. Rev. E*, vol. 72, no. 2, pp. 027 104–1–027 104-4, 2005.
[31] Y. Sun, B. Danila, K. Josić, and K. E. Bassler, "Improved community structure detection using a modified fine-tuning strategy," *Europhysics Lett.*, vol. 86, no. 2, pp. 028 004–1–028 004-6, 2009.
[32] G. Wang, Y. Shen, and M. Ouyang, "A vector partitioning approach to detecting community structure in complex networks," *Comput. Math. Appl.*, vol. 55, no. 12, pp. 2746–2752, 2008.
[33] T. Richardson, P. J. Mucha, and M. A. Porter, "Spectral tripartitioning of networks," *Phys. Rev. E*, vol. 80, no. 3, pp. 036 111–1–036 111–10, 2009.
[34] S. White and P. Smyth, "A spectral clustering approach to finding communities in graphs," in *Proc. SIAM Int. Conf. Data Mining*, 2005, pp. 76–84.
[35] J. Ruan and W. Zhang, "An efficient spectral algorithm for network community discovery and its applications to biological and social networks," in *Proc. IEEE Int. Conf. Data Mining*, 2007, pp. 643–648.
[36] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. Cambridge, USA: The MIT Press, 1992.
[37] X. Liu, D. Li, S. Wang, and Z. Tao, "Effective algorithm for detecting community structure in complex networks based on ga and clustering," in *Proc. Int. Conf. Comput. Sci.*, 2007, vol. 4488, pp. 657–664.
[38] M. E. J. Newman, "Spectral methods for community detection and graph partitioning," *Phys. Rev. E*, vol. 88, no. 4, pp. 042 822–1–036 111-10, 2013.
[39] P. Pons and M. Latapy, "Computing communities in large networks using random walks," *J. Graph Algorithms Appl.*, vol. 10, no. 2, pp. 191–218, 2006.
[40] J. J. H. Ward, "Hierarchical grouping to optimize an objective function," *J. Amer. Statistical Assoc.*, vol. 58, no. 301, pp. 236–244, 1963.
[41] M. Rosvall and C. T. Bergstrom, "The map equation," *The Eur. Phys. J. Special Topics*, vol. 178, pp. 13–23, 2009.
[42] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *Comput. Netw. ISDN Syst.*, vol. 30, pp. 107–117, 1998.
[43] R. Andersen, F. Chung, and K. Lang, "Using pagerank to locally partition a graph," *Internet Mathematics*, vol. 4, no. 1, pp. 35–64, 2007.
[44] T. A. B. Snijders and K. Nowicki, "Estimation and prediction for stochastic blockmodels for graphs with latent block structure," *J. Classification*, vol. 14, no. 1, pp. 75–100, 1997.
[45] J. Pearl, "Reverend bayes on inference engines: A distributed hierarchical approach," in *Proc. AAAI National Conf. Artif. Intell.*, 1982, pp. 133–136.
[46] J. Xie and B. K. Szymanski, "Towards linear time overlapping community detection in social networks," in *Proc. Advances Knowl. Discovery Data Mining*, 2012, vol. 7302, pp. 25–36.

[47] J. Yang and J. Leskovec, "Overlapping community detection at scale: A nonnegative matrix factorization approach," in *Proc. ACM Int. Conf. Web Search Data Mining*, 2013, pp. 587–596.
[48] J. J. Whang, D. F. Gleich, and I. S. Dhillon, "Overlapping community detection using neighborhood-inflated seed expansion," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 5, pp. 1272–1284, May 2016.
[49] A. Lancichinetti, S. Fortunatoa, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Phys. Rev. E*, vol. 78, no. 2, pp. 046 110–1–046 110-5, 2008.
[50] S. Fortunato, "inthepress2 - santofortunato," [Online]. Available: https://sites.google.com/site/santofortunato/inthepress2/, Accessed on: May 14, 2018.
[51] J. Leskovec and A. Krevl, "Stanford large network dataset collection," Jun. 2014. [Online]. Available: http://snap.stanford.edu/data/, Accessed on: May 15, 2018.
[52] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," in *Proc. IEEE Int. Conf. Data Mining*, 2012, pp. 745–754.
[53] S. Harenverg, "harenbergsd / community-detection-survey – bitbucket," Jun. 2017. [Online]. Available: https://bitbucket.org/harenbergsd/community-detection-survey/, Accessed on: May 16, 2018.
[54] S. Agarwal, N. Snavely, L. Simon, S. M. Seitz, and R. Szeliski, "Building Rome in a day," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2009, pp. 72–79.
[55] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.
[56] T. Weyand and B. Leibe, "Visual landmark recognition from internet photo collections: A large-scale evaluation," *Comput. Vis. Image Understanding*, vol. 135, pp. 1–15, 2015.
[57] Yahoo! Inc., "Together | flickr," [Online]. Available: https://www.flickr.com, Accessed on: May 21, 2004.
[58] T. Weyand and B. Leibe, "Discovering favorite views of popular places with iconoid shift," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2011, pp. 1132–1139.
[59] B. Leibe, "Computer vision," [Online]. Available: https://www.vision.rwth-aachen.de/page/paris500k, Accessed on: May 22, 2018.
[60] C. Audet and J. J. E. Dennis, "Analysis of generalized pattern searches," *SIAM J. Optimization*, vol. 13, no. 3, pp. 889–903, 2003.
[61] L. Danon, A. Díaz-Guilera, J. Duch, and A. Arenas, "Comparing community structure identification," *J. Statistical Mech.: Theory Experiment*, vol. 2005, no. 9, p. P09008, 2008.
[62] M. Frigge, D. C. Hoaglin, and B. Iglewicz, "Some implementations of the boxplot," *Amer. Statistician*, vol. 43, no. 1, pp. 50–54, 1989.
[63] Y. Fan, M. Li, P. Zhang, J. Wu, and Z. Di, "Accuracy and precision of methods for community identification in weighted networks," *Physica A: Statistical Mech. Appl.*, vol. 377, pp. 363–372, 2007.
[64] A. Lancichinetti and S. Fortunato, "Community detection algorithms: A comparative analysis," *Phys. Rev. E*, vol. 80, no. 5, pp. 056 117–1–056 117–11, 2009.
[65] G. Csardi and T. Nepusz, "The igraph software package for complex network research," *InterJournal Complex Syst.*, vol. 1695, pp. 1–9, 2006.
[66] G. Csardi, "igraph - network analysis software," Jun. 2015. [Online]. Available: http://igraph.org/, Accessed on: Apr. 24, 2018.
[67] E. Ferrara, "Community detection - Emilio Ferrara, Ph.D," [Online]. Available: http://www.emilio.ferrara.name/code/conclude/, Accessed on: Apr. 24, 2018.

Makoto Okuda received the MS degree in electrical and electronic engineering from Kobe University, in 2000. He has worked at Japan Broadcasting Corporation (NHK) since 2000. He has been on secondment to the National Institute of Information and Communications Technology (NICT), Japan, since 2015. He is currently a senior researcher with the Universal Communication Research Institute, NICT. His research interests include machine learning, computer vision, and robotics.

Shin'ichi Satoh received the BE degree in electronics engineering and the ME and PhD degrees in information engineering from the University of Tokyo, in 1987, 1989, and 1992, respectively. He joined the National Center for Science Information Systems (NACSIS), Tokyo, in 1992. He has been a full professor with the National Institute of Informatics (NII), Tokyo, since 2004. He was a visiting scientist at the Robotics Institute, Carnegie Mellon University, from 1995 to 1997. His research interests include image processing, video content analysis, and multimedia databases. Currently, he is leading the video processing project at NII, addressing video analysis, indexing, retrieval, and mining for broadcasted video archives.

Yoichi Sato received the BS degree from the University of Tokyo, in 1990, and the MS and PhD degrees in robotics from the School of Computer Science at Carnegie Mellon University, in 1993 and 1997. He is a professor with the Institute of Industrial Science, The University of Tokyo. His research interests include physics-based vision, reflectance analysis, first-person vision, and gaze sensing and analysis. He has served in several conference organization and journal editorial roles, including the *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *International Journal of Computer Vision*, *Computer Vision and Image Understanding*, ICCV 2021 Program Co-Chair, ACCV 2018 General co-chair, ACCV 2016 Program co-chair, and ECCV 2012 Program co-chair.

Yutaka Kidawara received the PhD degree in engineering from Kobe University, in 1999. He was a researcher at Kobe Steel, Ltd., from 1990 to 2001, then, in 2001, joined the Communication Research Laboratories (currently NICT). After completing a period of temporary transfer to the Cabinet Office, Government of Japan, he has been the Director General of the Universal Communication Research Institute of NICT since April 2011. Concurrently, he is also serving as the Director General of the Advanced Speech Translation Research and Development Promotion Center, NICT. His research interests include credible information analysis, knowledge processing, database systems, and ambient intelligence technologies.