Constrained Local Graph Clustering by Colored Random Walk

Yaowei Yan
yxy230@ist.psu.edu
Pennsylvania State University

Yuchen Bian
yub31@ist.psu.edu
Pennsylvania State University

Dongsheng Luo
dul262@ist.psu.edu
Pennsylvania State University

Dongwon Lee
dlee@ist.psu.edu
Pennsylvania State University

Xiang Zhang
xzhang@ist.psu.edu
Pennsylvania State University

ABSTRACT

Detecting local graph clusters is an important problem in big graph analysis. Given seed nodes in a graph, local clustering aims at finding subgraphs around the seed nodes, which consist of nodes highly relevant to the seed nodes. However, existing local clustering methods either allow only a single seed node, or assume all seed nodes are from the same cluster, which is not true in many real applications. Moreover, the assumption that all seed nodes are in a single cluster fails to use the crucial information of relations between seed nodes. In this paper, we propose a method to take advantage of such relationship. With prior knowledge of the community membership of the seed nodes, the method labels seed nodes in the same (different) community by the same (different) color. To further use this information, we introduce a color-based random walk mechanism, where colors are propagated from the seed nodes to every node in the graph. By the interaction of identical and distinct colors, we can enclose the supervision of seed nodes into the random walk process. We also propose a heuristic strategy to speed up the algorithm by more than 2 orders of magnitude. Experimental evaluations reveal that our clustering method outperforms state-of-the-art approaches by a large margin.

CCS CONCEPTS

• **Mathematics of computing** → **Graph algorithms**; • **Information systems** → **Clustering**; • **Theory of computation** → **Random walks and Markov chains**.

KEYWORDS

community detection; local graph clustering; random walk; non-Markovian

ACM Reference Format:
Yaowei Yan, Yuchen Bian, Dongsheng Luo, Dongwon Lee, and Xiang Zhang. 2019. Constrained Local Graph Clustering by Colored Random Walk. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3308558.3313719

WWW '19, May 13–17, 2019, San Francisco, CA, USA

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313719

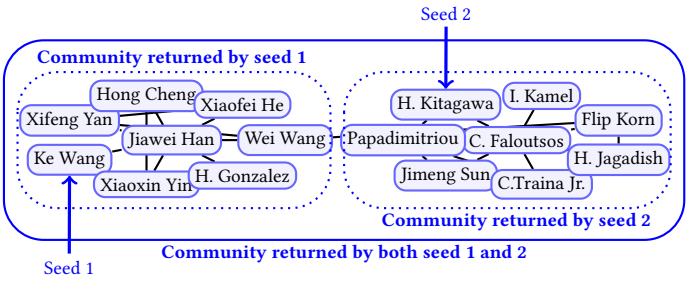


Figure 1: Example of an academic social network. All nodes are data science researchers and form two subgroups by their collaborations. Query from each side returns a single subgroup (in dotted boxes). The joint query returns two subgroups together (in the solid box).

1 INTRODUCTION

The graph/network is a natural representation of real-world relations. Graph clustering (or community detection) is a fundamental problem and is the base of many applications, such as online recommendation, medical diagnosis, social network and biological network analysis [11, 14, 15, 18]. The clustering of a graph aims to find groups of nodes that are closely related to each other in each group. Unlike global graph clustering which retrieves all clusters/communities from a graph, local graph clustering focuses on the subgraph within the neighborhood of the seed/query nodes and is applicable to very large data [17]. With the increasing size of networks, recently local clustering has attracted lots of research interest [6, 8, 9, 22].

The seed nodes of local graph clustering can be single or multiple. Existing local graph clustering methods do not accept multiple seed nodes (e.g., Random Walk with Restart [19]), or simply assume that all these seeds are from the same cluster (e.g., Personalized PageRank [1]). We argue that knowing the relations of seed nodes (in the same community or not) are important, as the network data may be noisy or lossy to be clustered accurately. Furthermore, it is a relatively easy task to label the seed nodes by users or domain experts, since the number of seed nodes is small.

Knowing seeds nodes are in the same community can help to merge divided subgroups to a whole community. Fig. 1 is part of a collaboration network used by a conference program committee (PC) member recommendation system. Given that Dr. Ke Wang and Dr. Kitagawa are prospectively favorable PC members, we want the system to recommend more researchers as candidates.

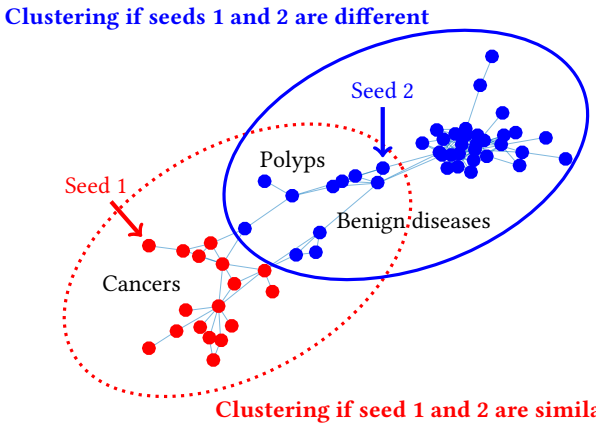


Figure 2: Example of a stomach disease similarity network. Vertices are diseases and edges are similarities between diseases. Red nodes are malignant diseases and blue nodes are benign ones. The dotted red circle contains diseases detected by a single querying seed 1. The solid blue circle encloses clustering result by knowing seed 1 and seed 2 are from different communities.

However, local clustering from either Dr. Wang or Dr. Kitagawa returns their own collaboration subgroup, which is not appropriate due to diversity concerns. On the other hand, assuming Dr. Wang and Dr. Kitagawa are in the same research community and querying jointly from both of them may return a larger community with diverse researchers relevant to them. The final recommendation can be decided by the proximity score of each researcher with respect to the two query seeds.

More importantly, knowing that seed nodes are different can separate a dense subgraph into semantically meaningful communities. Fig. 2 is a network of stomach diseases where red nodes are malignant tumors and blue nodes denote benign stomach diseases such as gastritis and ulcer. By conventional local clustering method, either query from seed 1 or from both seed 1 and 2 will enclose polyps into group of malignant diseases, which is not correct. On the contrary, with domain knowledge that seed 2 is benign and different from seed 1, stomach polyps can be well divided from malignant tumors by our proposed method. This helps the automatic medical diagnosis system to correctly identify a specific disease [13].

Generally, existing local graph clustering methods cannot take advantage of seed node labels, and prefer well-structured networks [8]. For networks without clear community structures, the performance of such methods is limited. In this paper, we introduce a method to leverage supervision of labeled seed node. If a clustering task has multiple seed nodes, we assume that the relationships of seed nodes are already known: we know that which seeds should be in the same community, and which should not. We name this local graph clustering problem with labeled seed nodes as constrained local graph clustering.

To solve the constrained local graph clustering problem, we propose a colored random walk (CRW) algorithm, which is motivated by random walk based local graph clustering [1]. In addition to using random walkers to explore a network, we label seed nodes and random walkers by colors, where seed nodes believed to be in the same community are marked with identical color. The algorithm launches colored random walkers for each color starting from the seed nodes with the corresponding color.

Unlike previous random walk methods [3, 20] where walkers change nothing to the graph, the colored random walker will leave colors to every node it visited. The colors on each node will affect the transition probability of subsequent walkers. In general, a walker is more likely to visit nodes with the same color. In other words, walkers are attracted by nodes with the same color and repulsed by nodes with different colors. The attraction and repulsion are affected by the amount of colors accumulated on each node. By this means, communities with different color of seed nodes are divided, even in unclear network structures.

For local graph clustering with single seed node, the proposed colored random method still outperforms the traditional approaches. This is due to the attraction of walkers by the community members which are assigned with the same color, especially by the central influential nodes with large degrees in each community. These nodes cumulate a higher amount of color by being visited more frequently (Fig. 9), and the cumulative color, in turn, prevents the random walkers from fleeing their community. Due to the same reason, our method is robust to the position of the seed nodes. No matter where is the seed node, the influential nodes remain the same, and so do the corresponding communities. This is different with the previous approaches, in which the performance degrades if seed nodes are close to cluster boundaries. Please refer to Section 5 for more details.

We formulate the colored random walk algorithm by power iterations, and introduce a heuristic speedup strategy to reduce the running time. By the nature of local graph clustering, nodes far away from seeds are not important. The strategy can speed up the searching process by more than 100 times while increasing the local clustering accuracy at the same time. Experiments show that the proposed algorithm outperforms the state-of-the-art baseline methods on both accuracy and speed.

The rest of the paper is organized as follows. Related works are discussed in Section 2. Section 3 explains the detail of our method. In Section 4, extensive experimental results are presented and analyzed. Section 5 provides two case studies for deeper insight of the proposed method, and finally Section 6 draws the conclusion.

2 RELATED WORK

The current local graph clustering methods have no restriction on seed nodes. Random Walk with Restart [20] uses Markovian random walkers to explore the networks. With certain probability, the walkers jump back to seed nodes. The local clusters are cut from nodes frequently visited by the walkers. Heat Kernel Diffusion [8] instead adopts values diffused from seed nodes to evaluate how close the nodes are to the seeds. Query-biased Densest Connect Subgraph method [22] weights nodes by random walk. Then the local community is selected to minimize a predefined goodness function. All these methods assume seed nodes are from a single cluster.

Semi-supervised learning, including semi-supervised clustering with constrains, gained success as they do not require a large amount of human annotations [4]. In semi-supervised learning, the labels are propagated from labeled seed instances by random-walk-like mechanisms through the similarity graph, and then a classifier is trained to categorize all other instances [24, 25]. However, traditional semi-supervised learning focuses more on classification than clustering, and is a heavy framework since it considers both local and global consistency during the learning process.

The colored random walk is also relevant to the Vertex-reinforced Random Walk (VRW), which views graph nodes as urns [16]. Each time the random walker visits a node, it puts a ball with its color into the corresponding urn. The further visiting probability of the random walker is reinforced by the numbers and colors of balls in each node. Though it is proved that single-color VRW is guaranteed to converge to a distribution [2], the converged distribution is not unique due to the randomness of the walker, which diminishes the performance of VRW in local graph clustering. Moreover, the theories of multi-color VRW are still underdeveloped.

3 OUR PROPOSAL

Local graph clustering requires proximity measurement to decide which nodes should be in a local cluster. Random walk based approaches such as Random Walk with Restart (a.k.a. Personalized PageRank) are commonly used to evaluate the node proximity in graphs [20]. Compared with other methods, random walk takes advantage of local network structure and reports better performance [9]. In this work, we extend the random walk method to colored random walk to solve the problem of local graph clustering with constrains.

3.1 Preliminaries

The local graph clustering targets on a graph $G = (V; E)$, where V is the node set and E is the edge set. The adjacency matrix of graph G is denoted by $\mathbf{A}$, whose element $\mathbf{A}_{ij}$ is the edge weight between node V_i and node V_j. $\mathbf{P}$ is the transposed column-stochastic transition matrix with entries $\mathbf{P}_{ji} = \mathbf{A}_{ij} / \sum_j \mathbf{A}_{ij}$. The main symbols and their definitions can be found in Table 1.

The core function of local graph clustering is to measure the proximity of each node with the seed node(s). In Random Walk with Restart (RWR), a random walker starts from the seed node q, and randomly walks to a neighbor node according to edge weights. At each time point $(t + 1)$, the walker continues walking with probability α or returns to the query node q with probability $(1 - \alpha)$. The proximity score of node p with respect to node q is defined as the converged probability that the walker visits node p:

$$\mathbf{c}^{(t+1)} = \alpha \mathbf{P} \mathbf{c}^{(t)} + (1 - \alpha) \mathbf{s} \tag{1}$$

where $\mathbf{s}$ is the column vector of the initial distribution with q^{th} entry as 1 and all other entries 0, and $\mathbf{c}$ is the current distribution vector whose p^{th} entry is the visiting probability of node p.

RWR can be extended to accommodate multiple seed nodes. For n seed nodes, the initial vector $\mathbf{s}$ consists of entries with value $1/n$ for each seed node. However, if seed nodes are from multiple clusters, RWR cannot use such information. One may run RWR multiple times, each for a single seed node, yet this approach fails to take full advantage the prior knowledge of cluster membership of the seed nodes. For example, if two seed nodes are known from different clusters, walkers of each seed node should avoid walking into the other seed's cluster. We give the formal definition of constrained local graph clustering as follows.

Table 1: Main symbols

Symbol	Definition
$G = (V; E)$	graph G with node set V and edge set E
$\mathbf{A}; \mathbf{P}$	adjacency matrix and transition matrix
$\mathbf{c}; \mathbf{s}$	current and initial color distributions
$\alpha; \theta$	forward probability and color threshold
$\psi(t)$	decaying factor at time t
$\lambda_1; \lambda_2$	attraction and repulsion coefficient

Definition 3.1 (Constrained local graph clustering). Given graph $G = (V; E)$ and seed node set with K different colors, the constrained local graph clustering aims to find K local communities containing seed nodes of each color.

3.2 The Colored Random Walk

To solve the constrained local graph clustering problem, we propose a Colored Random Walk (CRW) method in this section, which takes advantage of seed node labels. Seed nodes of the same cluster are assigned with identical color, and seed nodes from different clusters have different colors. For each color, the algorithm sends out a colored walker from the corresponding seed nodes to explore the network. During the exploration, the walker leaves a certain amount of colored values to the visited nodes, similar to the RWR method in Eq. 1.

The main difference between CRW and conventional RWR is that the movement of walkers is affected by existing colors of nodes. Generally, a walker is more likely to visit nodes with identical colors, and is less likely to visit nodes with distinct colors. Since color values are changed at each time point t, the transition matrix $\mathbf{P}$ should be updated accordingly.

For a constrained local graph clustering task with K colors, the CRW algorithm maintains K transition matrix $\mathbf{P}^{(k;t)}$ for color k at time t. Then the Eq. 1 expands to a new power iteration formula

$$\mathbf{c}^{(k;t+1)} = \alpha \mathbf{P}^{(k;t)} \mathbf{c}^{(k;t)} + (1 - \alpha) \mathbf{s}^{(k)} \tag{2}$$

where $\mathbf{c}^{(k;t)}$ is the distribution of color k at time t, and $\mathbf{s}^{(k)}$ is the initial vector of color k, respectively.

3.3 Transition Matrix Reinforcement

The transition matrix $\mathbf{P}^{(k;t)}$ in Eq. 2 is based on the original transition matrix $\mathbf{P}$, and is reinforced by all current color distributions $\mathbf{c}^{(`;t)}$, $1 \quad ` \quad K$. The reinforcement is done by

$$\mathbf{P}^{(k;t)} = \mathbf{P} \quad (1 + \mathbf{R}^{(k;t)}) \tag{3}$$

where operator $\quad$ is the Hadamard (entry-wise) product of two matrices with the same size [7]. By Hadamard product, we avoid adding new edges to graph G, and the reinforcement is done by weight adjustment of existing edges. In the Eq. 3, '1' is used to keep the original transition matrix $\mathbf{P}$ as part of the reinforced transition

matrix $\mathbf{P}^{(k;t)}$. $\mathbf{R}$ is a coefficient matrix defined by its entry

$$R_{ij}^{(k;t)} = \sum_{\ell} c_i^{(\ell;t)} \lambda_{k\ell}; \quad \lambda_{k\ell} = \begin{cases} \lambda_1 & k = \ell \\ \lambda_2 & k \neq \ell \end{cases} \tag{4}$$

In Eq. 4, if the factor $\lambda_{k\ell}$ is defined as positive when color k equals color ℓ, and negative otherwise. This definition increases the likelihood of a walker going to the node with identical color, and decreases the likelihood of walking to the node with different colors. The mechanism can be understood as that, a walker is attracted by nodes with the same color, and repulsed by nodes with different colors. Here we denote by λ_1 the attraction coefficient and by λ_2 the repulsion coefficient.

By Eq. 3 and 4, the entries of $\mathbf{P}^{(k;t)}$ are possibly negative. To make $\mathbf{P}^{(k;t)}$ as a transition matrix meaningful, it is normalized before used in Eq. 2. The normalization includes two parts. First, the negative entries of $\mathbf{P}^{(k;t)}$ are set to be zero: $P_{ij}^{(k;t)} \leftarrow 0$ if $P_{ij}^{(k;t)} < 0$, and then each entry of the matrix is divided by the sum of its column to make the matrix column stochastic: $P_{ij}^{(k;t)} \leftarrow P_{ij}^{(k;t)} / \sum_i P_{ij}^{(k;t)}$.

3.4 Convergence

Unlike the traditional random walk methods which have stationary transition matrix $\mathbf{P}$, the transition matrix $\mathbf{P}^{(k;t)}$ of colored random walk varies at each time t. To guarantee the convergence of Eq. 2, we introduce a decaying mechanism changing $\mathbf{P}^{(k;t)}$ to $\mathbf{P}^{(k;t)}$ (Eq. 5), where the value of function $\psi(t)$ approaches to 0 when t increases. Then the colored random walk is adapted to Eq. 6.

$$\mathbf{P}^{(k;t)} = \begin{cases} \mathbf{P} & t = 0 \\ \psi(t)\ \mathbf{P}^{(k;t)} + (1 - \psi(t))\ \mathbf{P}^{(k;t-1)} & t \geq 1 \end{cases} \tag{5}$$

$$\mathbf{c}^{(k;t+1)} = \alpha\ \mathbf{P}^{(k;t)}\ \mathbf{c}^{(k;t)} + (1 - \alpha)\ \mathbf{s}^{(k)} \tag{6}$$

Here the definition of $\mathbf{P}$ remains the same with Eq. 3 and Eq. 4. Intuitively, when $\psi(t) \to 0$, $\mathbf{P}^{k;t} \to \mathbf{P}^{k;t-1}$, meaning that $\mathbf{P}^{k;t}$ converges. With a converged $\mathbf{P}^{k;t}$, the proximity score vector $\mathbf{c}^{(k;t)}$ converges, similar to the conventional random walk. The formal proof of convergence of $\mathbf{c}^{(k;t)}$ is given by Theorem 3.2.

Theorem 3.2 (Convergence of colored random walk). *Both $\mathbf{P}^{(k;t)}$ and $\mathbf{c}^{(k;t)}$ in Eq. 5 and Eq. 6 converge if $\psi(t) = \Omega(\alpha^t)$.*

Proof. To ease the description, we combine distributions, reinforced transition matrices, and initial vectors of all color at time t by

$$\mathbf{c}^{(t)} = \begin{bmatrix} \mathbf{c}^{(1;t)} \\ \mathbf{c}^{(2;t)} \\ \vdots \\ \mathbf{c}^{(K;t)} \end{bmatrix}; \mathbf{P}^{(t)} = \begin{bmatrix} \mathbf{P}^{(1;t)} & & & 0 \\ & \mathbf{P}^{(2;t)} & & \\ & & \ddots & \\ 0 & & & \mathbf{P}^{(K;t)} \end{bmatrix}; \mathbf{s} = \begin{bmatrix} \mathbf{s}^{(1)} \\ \mathbf{s}^{(2)} \\ \vdots \\ \mathbf{s}^{(K)} \end{bmatrix}$$

Evidently, Eq. 7 equals to Eq. 6 for all color $1 \leq k \leq K$. Similarly we define $\mathbf{P}^{(t)}$ as diagonal combination of all $\mathbf{P}^{(t;k)}$.

$$\mathbf{c}^{(t+1)} = \alpha\ \mathbf{P}^{(t)}\ \mathbf{c}^{(t)} + (1 - \alpha)\ \mathbf{s} \tag{7}$$

For convergence of decayed transition matrix $\mathbf{P}^{(t)}$, it is easy to know that for $t \geq 1$,

$$\|\mathbf{P}^{(t)} - \mathbf{P}^{(t-1)}\|_1 = \psi(t)\|\mathbf{P}^{(t)} - \mathbf{P}^{(t-1)}\|_1 \leq 2\psi(t)$$

Algorithm 1: The Colored Random Walk

Input : $\mathbf{P}, \mathbf{S}, \alpha, \lambda_1, \lambda_2, T, \psi(t)$
Output: $\mathbf{c}$

```
K ← |S|;
for k ← 1 to K do
    s^(k) ← 0;
    for each x ∈ S[k] do s_x^(k) ← 1/|S[k]| ;
s ← [s^(1); s^(2); ...; s^(K)];
c ← s;
P^(0) ← I_K ⊗ P;
P ← P^(0);
Λ ← I_N ⊗ λ_1;
Set all non-diagonal entries of Λ as λ_2;
Λ ← Λ ⊗ I_K;
for t ← 0 to T do
    c ← α P c + (1 − α)s;
    r ← Λ c;
    R ← r 1^T_{N×K};
    P̂ ← P^(0) ⊙ (1 + R);
    for each i and j do P̂_ij ← 0 for P̂_ij < 0;
    for each i and j do P̂_ij ← P̂_ij / Σ_x P̂_xj;
    P ← ψ(t) P̂ + (1 − ψ(t)) P;
```

For convergence of the color distribution $\mathbf{c}^{(t)}$, we define $\Delta^{(t)} = \|\mathbf{c}^{(t)} - \mathbf{c}^{(t-1)}\|_1$. Then

$$\begin{aligned}
\Delta^{(t+1)} &= \|\mathbf{c}^{(t+1)} - \mathbf{c}^{(t)}\|_1 = \|\alpha\mathbf{P}^{(t+1)}\mathbf{c}^{(t)} - \alpha\mathbf{P}^{(t)}\mathbf{c}^{(t-1)}\|_1 \\
&= \alpha\|\mathbf{P}^{(t+1)}\mathbf{c}^{(t)} - \mathbf{P}^{(t)}\mathbf{c}^{(t-1)}\|_1 \\
&= \alpha\|\mathbf{P}^{(t+1)}\mathbf{c}^{(t)} - \mathbf{P}^{(t+1)}\mathbf{c}^{(t-1)} + \mathbf{P}^{(t+1)}\mathbf{c}^{(t-1)} - \mathbf{P}^{(t)}\mathbf{c}^{(t-1)}\|_1 \\
&= \alpha\|\mathbf{P}^{(t+1)}(\mathbf{c}^{(t)} - \mathbf{c}^{(t-1)}) + (\mathbf{P}^{t+1} - \mathbf{P}^{t})\mathbf{c}^{t-1}\|_1 \\
&\leq \alpha\|\mathbf{P}^{(t+1)}(\mathbf{c}^{(t)} - \mathbf{c}^{(t-1)})\|_1 + \alpha\|(\mathbf{P}^{(t+1)} - \mathbf{P}^{(t)})\mathbf{c}^{t-1}\|_1 \\
&\leq \alpha\|\mathbf{P}^{(t+1)}\|_1\|\mathbf{c}^{(t)} - \mathbf{c}^{(t-1)}\|_1 + \alpha\|\mathbf{P}^{(t+1)} - \mathbf{P}^{(t)}\|_1\|\mathbf{c}^{t-1}\|_1 \\
&\leq \alpha\|\mathbf{c}^{(t)} - \mathbf{c}^{(t-1)}\|_1 + \alpha\|\mathbf{P}^{(t+1)} - \mathbf{P}^{(t)}\|_1 \\
&\leq \alpha\Delta^{(t)} + 2\alpha\psi(t+1) \leq \alpha^2\Delta^{(t-1)} + 2\alpha^2\psi(t) + 2\alpha\psi(t+1) \\
&\leq \alpha^t\Delta^{(1)} + 2\sum_{\tau=1}^{t}\alpha^\tau\psi(t-\tau+2)
\end{aligned}$$

To make $\Delta^{(t+1)}$ converge to 0, a sufficient condition is that $\psi(t) = \Omega(\alpha^t)$, which means $0 \leq c\psi(t) \leq \alpha^t$ for any $t > t_0$ and $c > 0$. Given $\psi(t) \leq 2\alpha^{t-1}$ for $t > t_0$,

$$\begin{aligned}
\Delta^{(t+1)} &\leq 2\alpha^t + 2\sum_{\tau=t+2-t_0}^{t}\alpha^\tau + 2\sum_{\tau=1}^{t+1-t_0}\alpha^{t+1} \\
&< 2\alpha^t + 2(t_0 - 1)\alpha^{t-t_0+2} + 2(t - t_0 + 1)\alpha^{t+1} \\
&< 2t_0\alpha^{t-t_0+2} + 2(t - t_0 + 1)\alpha^{t+1} \to 0
\end{aligned}$$

□

3.5 The Algorithm

By Eq. 7, we can formulate the colored random walk as shown in Algorithm 1. The input of the algorithm includes original transition matrix $\mathbf{P}$, seed node sets $\mathbf{S}$ of all colors, penalty factor α, attraction coefficient λ_1 and repulsion coefficient λ_2, together with the number of iterations T and the decaying faction $\psi(t)$. The output of the algorithm is the color distribution vector $\mathbf{c}$ for all colors. The details of the algorithm are as follows.

Lines 1 – 11 initialize the algorithm. Specifically, in line 1 we get the number of colors K by the number of seed node sets in $\mathbf{S}$. Lines 3 and 4 prepare for the initial color vectors $\mathbf{s}^{(k)}$ for each color k. From line 5 to line 8 we combine initial distributions and transition matrix of all colors together by $\mathbf{s}$ and $\mathbf{P}^{(0)}$, as explained in Section 3.4, and initialize the color distribution vector $\mathbf{c}$ and the modified transition matrix $\mathbf{P}$.

To calculate the color reinforcement for each node, we create a $N \quad N$ factor matrix Λ (lines 9 to 11), which originally has the value of λ_1 on all diagonal entries and λ_2 on all non-diagonal entries. The size of matrix Λ is then amplified (line 11) by Kronecker Product [7] with an identity matrix $\mathbf{I}_K$, to multiply with vector $\mathbf{c}$ (line 14), which is a matrix-form equivalence to the operation in Eq. 4.

The for loop from line 12 to 19 is the power iteration process. We first generate the color distribution for next iteration (line 13), and then generate the reinforcement vector $\mathbf{r}$ (line 14). Vector $\mathbf{r}$ is repeated to form a matrix $\mathbf{R}$ (line 15), which means that the reinforcement is fair to every node. After generating new reinforced matrix $\hat{\mathbf{P}}$ (line 16), we remove negative entries (line 17) and normalize it to column stochastic (line 18), and update the decayed transition matrix for the next iteration (line 19).

The power iteration part of the algorithm is critical in analyzing the time complexity, where all operations including calculations of color distribution (line 13), reinforcement (line 14) and decayed transition matrix (line 19) take $O(K|E|)$, if we base the operations on sparse matrices. Therefore, the overall time complexity is $O(TK|E|)$. As empirically reported in [9], $T = 10$ is sufficient for the convergence of distribution $\mathbf{c}$.

With the distribution $\mathbf{c}$ for each color, we rank nodes by their colors and cut local communities by sweeping from the top-rank node and find a cluster with minimum conductance [1].

3.6 Speedup Strategy

Time complexity of the proposed colored random walk algorithm is proportional to the number of edges $|E|$, implying that the algorithm searches the entire graph for local clusters. This is undesirable as the task of local clustering focuses only on the nodes close to the seeds.

In this section, we propose a localized variant of Algorithm 1, with the basic idea that random walkers only search the influential nodes around the seed nodes. Unlike Algorithm 1 where the walkers go as far as T-hops from the seeds, we only consider nodes with colors above a given tolerance threshold θ.

As the random process in Eq. 2 can be viewed as the diffusion of colors from each node to its neighbors, the key point of Algorithm 2 is that only colors with values above a threshold θ will diffuse to next iteration. The intuition behind this is that small color amount will not affect colors of next iteration very much. When a walker reaches a node with color value less than θ, it stops there and does not go to the node's neighbors, which prevents the process from searching too many further nodes. We also neglect the decaying function $\psi(t)$ and reinforce the node on demand (with color value greater than θ) to accelerate the algorithm.

Algorithm 2: The Localized Colored Random Walk

```
   Input  : P, S, α, λ1, λ2, T, θ
   Output : c
1  for each color k do
2      c^(k)  0;
3      for each node i ∈ S[k] do c_i^(k)  1/|S[k]| ;
4  for t  1 to T do
5      ĉ^(k)  0;
6      for each color k do
7          for each c_i^(k) > θ do
8              p  0;
9              for each neighbor j of node i do
10                 p_j  P_ij^(k;t);
11                 for each color ℓ do
12                     if ℓ = k then p_j  p_j + λ1 c_j^(ℓ) ;
13                     else p_j  p_j  λ2 c_j^(k) ;
14                 if p_j < 0 then p_j  0 ;
15             for each p_j > 0 do ĉ_j^(k)  ĉ_j^(k) + α c_i^(k) p_j / Σ p ;
16         for each seed node i ∈ S[k] do
17             ĉ_i^(k)  ĉ_i^(k) + (1  α)/|S[k]|;
18     for each color k do c^(k)  ĉ^(k) ;
```

In Algorithm 2, lines 1 – 3 initialize the color distribution according to seed nodes, and lines 4 – 18 simulate the power iteration in a heuristic way. In this heuristic algorithm, we remove the process of decaying, as it is shown that the result of experiments without decaying is similar. Specifically, at each iteration (line 4) the current color scores are in $\mathbf{c}^{(k)}$ for color k, and the new scores for next iteration are calculated and stored in $\hat{\mathbf{c}}^{(k)}$. Note that, $\mathbf{c}^{(k)}$ is changed to $\hat{\mathbf{c}}^{(k)}$ after the scores of all colors are updated (line 18). Note that the color scores are updated simultaneously.

By this strategy, though color values diminish at each iteration, the summation of colors remains its majority value. The difference turns out to be less than 1%.

4 EXPERIMENTAL VALIDATION

Extensive experiments are conducted on both synthetic and real-world data to evaluate the effectiveness and efficiency of the proposed algorithms. All experiments are performed on a server with Redhat OS, Xeon 2.6GHz CPU, and 64G memory. The core functions are implemented by C++.

Each experimental result is gathered on average of 10,000 trials. The querying seeds are randomly selected. The accuracy is measured by F1-score, which is defined as $2pr/(p + r)$, where p and r

are precision and recall, respectively.

4.1 Non-overlapping Clusters

We use the following networks with non-overlapping ground truth clusters to test the effectiveness of our proposed method. The clusters are large enough to test the searching ability of the colored random algorithm algorithm.

Disease networks. The networks contain a disease similarity network [13] with 9,721 diseases. For this network, there are two sets of clustering ground truth for the disease network, with 17 classes and 47 classes, respectively. We use both sets of ground truth, and denote the experiments by Disease 1 and Disease 2, accordingly. Disease 3 is a phenotype similarity network with 5,080 diseases in 20 categories [21].

Gene network. The gene network represents the functional relationship between 8,503 genes [21] with 200 pathway labels.

Author network. This network comprises 20,111 authors from 4 research areas: Data Mining, Data Base, Information Retrieval and Machine Learning. The edge of this network is the number of papers co-published by the authors.

Conference network. This is a full graph comprises 20 main conferences from the same 4 areas with the author network. The edges are cosine similarity of keywords from the publication of the conferences. We use research areas as the ground truth for both author network and conference network.

Since no previous method is designed for the constrained local graph clustering problem, we compare the proposed colored random walk (CRW) method with the following five state-of-the-art local clustering methods, which support multiple seed node querying: Random Walk with Restart (RWR) [20] uses Eq. 1 to calculate the proximity score of each node; Heat Kernel Diffusion (HKD) [8] measures node proximity by heat kernel network diffusion; Query-biased Densest Connect subgraph (QDC) [22] assigns weights to nodes using RWR scores and finds the query biased densest subgraph; Vertex-reinforced Random Walk (VRW) [12] reinforces the next-hop visiting probability by previous visiting counts. For our method CRW, we use the localized colored random walk (Algorithm 2) with $\alpha = 0{:}9, \lambda_1 = 1000, \lambda_2 = 10, \theta = 10^{-5}$.

Effectiveness evaluations are conducted on the selected networks. Since the accuracy difference between our two algorithms is small, we present the results of Algorithm 2 only. Table 2 shows the result of single seed node querying. By column "CRW" we only use one seed node with a single color, so there is no repulsion of different colors, which serves as a baseline of our method. From the results in Table 2, it can be seen that even the basic CRW can outperform the state-of-the-art methods in terms of clustering accuracy.

In Table 2 we add an extra column of CRW_1^1, which randomly picks up two seed nodes of different colors to examine the impact of color repulsion. Comparison of CRW and CRW_1^1 shows that resistance of distinct colors helps to improve the clustering accuracy significantly. On all datasets CRW_1^1 achieves the best performance.

Table 3 shows the result of 2-seed-node query on the same 6 datasets. For each method except CWR_2^2 we use two seed nodes from the same cluster. CWR_2^2 is similar to CWR_1^1 by choosing 2 colors, each having two seeds. Though initially VRW is not designed for multiple seed nodes, we make it work by adding extra random walkers that are reinforced by rules of the original VRW. Comparing Table 3 with Table 2, the conclusion is that almost all methods improve clustering accuracy by having more seeds. Furthermore, CWR gains more improvement on average. Again CRW_1^1 and CRW_2^2 have the best performance among all chosen methods.

Table 2: 1-query accuracy for non-overlapping clusters

F1-score	RWR	HKD	QDC	VRW	CRW	CRW_1^1
Disease 1	0.33	0.27	0.11	0.08	**0.34**	**0.37**
Disease 2	0.42	0.48	0.32	0.25	**0.49**	**0.53**
Disease 3	0.24	0.30	0.14	0.13	**0.30**	**0.38**
Gene	0.05	0.06	0.09	0.15	**0.16**	**0.22**
Author	0.27	0.03	0.01	0.01	**0.30**	**0.40**
Conference	0.42	0.38	0.40	0.57	**0.66**	**0.71**

Table 3: 2-query accuracy for non-overlapping clusters

F1-score	RWR	HKD	QDC	VRW	CRW	CRW_2^2
Disease 1	0.40	0.30	0.11	0.10	**0.41**	**0.44**
Disease 2	0.43	0.54	0.38	0.29	**0.57**	**0.61**
Disease 3	0.26	0.33	0.19	0.10	**0.38**	**0.39**
Gene	0.05	0.07	0.08	0.17	**0.21**	**0.35**
Author	0.32	0.02	0.01	0.01	**0.35**	**0.41**
Conference	0.43	0.45	0.40	0.57	**0.89**	**0.91**

4.2 Overlapping Clusters

Intuitively colored random walk works with non-overlapping clusters. However, due to its color-agglomerating nature, it also works for detecting overlapping clusters. Table 4 lists the statistics 4 of networks with overlapping ground truth communities (Amazon, DBLP, YouTube, LiveJournal). These datasets are publicly available at http://snap.stanford.edu. For the quality of clusters, we select clusters with size larger than 10, and within 3σ range of the average cluster size. The last column of Table 4 shows the mean and standard deviation (σ) of the chosen clusters.

Table 4: Networks with overlapping communities

Name	$\|V\|$	$\|E\|$	# clusters	avg (σ)
Amazon	334,863	925,872	75,149	13.23 (11.24)
DBLP	317,080	1,049,866	13,477	10.87 (20.41)
YouTube	1,134,890	2,987,624	8,385	10.22 (18.81)
LiveJournal	3,997,962	34,681,189	287,512	23.67 (24.75)

Table 5: 1-query accuracy on overlapping communities

F1-score	RWR	HKD	QDC	VRW	CRW	CRW_1^1
Amazon	0.92	**0.93**	0.90	0.70	0.92	**0.93**
DBLP	0.41	0.59	0.65	0.57	**0.72**	**0.76**
YouTube	0.05	0.06	0.19	0.13	**0.27**	**0.30**
LiveJournal	0.69	0.74	0.74	0.42	**0.81**	**0.84**

Table 6: 2-query accuracy on overlapping communities

F1-score	RWR	HKD	QDC	VRW	CRW	CRW_2^2
Amazon	0.93	0.94	0.91	0.79	**0.95**	**0.96**
DBLP	0.42	0.61	0.73	0.68	**0.78**	**0.81**
YouTube	0.05	0.06	0.21	0.16	**0.38**	**0.42**
LiveJournal	0.69	0.75	0.79	0.45	**0.84**	**0.86**

The clustering results are listed in both Table 5 and Table 6. It shows that 2-seed quires are generally better than single seed node. Among all methods HKD is relatively good since it is refined for the small overlapping communities. However, more seed nodes do not improve its performance very much (only 1% on average). The probability-reinforced methods VRW and CRW gain the biggest improvement by adding an extra seed. Once again our proposed method leads the accuracy measurements. For seed nodes more than 2, Fig. 3 shows that most methods perform better with more seeds. However, CRW, VRW and QDC take better advantage of more seed nodes, as they all have mechanisms of reinforcement on node weights.

Fig. 3 gives the results where seed nodes are from a single community. However, when seed nodes are from distinct communities, the other methods cannot use this relationship, so their performances are very low (Fig. 4). On the contrary, our method takes advantage of this cannot-link information, resulting in the increased accuracy with more communities (colors).

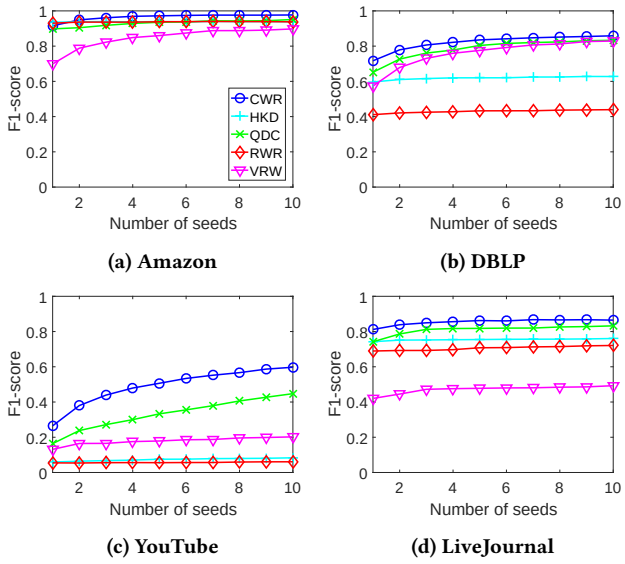
We also evaluate the efficiency of the selected methods on datasets provided in Table 4, since networks are relatively large. To be fair, the running time is measured on single color and single node on every method. Fig. 5 shows that CRW is one of the fastest methods of all, comparable to the HKD method, and is generally 100 times faster than the other approaches. Generally, CRW takes no more than 200 milliseconds in finding a target local cluster. For the scalability of colors and seed nodes, the running time of CRW does not increase with number of seed nodes, and is proportional with the number of colors.

4.3 Impact of Color Attraction and Repulsion

To analyze the sensitivity of the main parameters of our method ($\lambda_1, \lambda_2, \theta$), we generate synthetic networks by the standard LFR benchmark [10].

We first test the impact of attraction and repulsion of Algorithm 1, by varying factor λ_1 and λ_2 on the synthetic graph of Fig. 6. The results are presented in Fig. 7, in terms of accuracy vs. attraction λ_1 and repulsion λ_2. We set $\psi(t) = 0{:}9^t$ is this experiment.

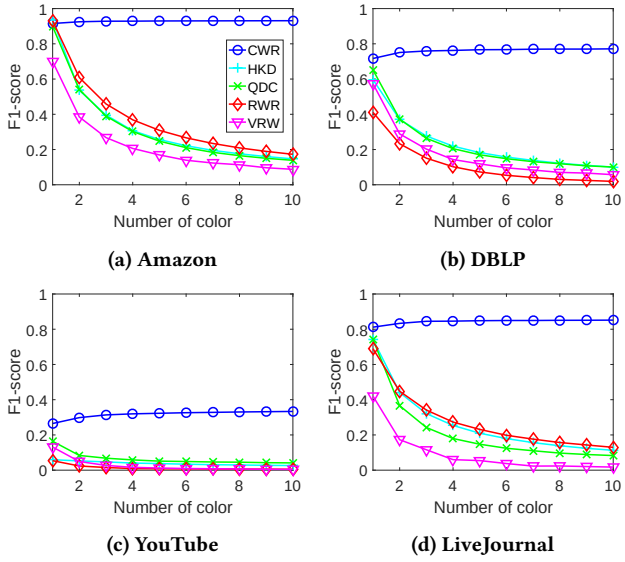
First we try our algorithm on the traditional problem of local graph clustering, where there is only one target community and accordingly only one color (Fig. 7a). We randomly choose 1 seed node (1 1) or 2 seed nodes (2 1) from a community, or 4 seed nodes from 2 different communities (2 2). Since there is no different color, we use attraction coefficient λ_1 only and set $\lambda_2 = 0$. In Fig. 7a the x-axis is the logarithmic value of λ_1 and y-axis is the F1-score. When λ_1 is close to 0, our algorithm degenerates into the conventional random walk based local clustering. By this means we can compare our methods with the basic random walk, as well as measuring the sensitivity of parameters λ_1 and λ_2.



(a) Amazon (b) DBLP

(c) YouTube (d) LiveJournal

Figure 3: Impact of adding more seed nodes



(a) Amazon (b) DBLP

(c) YouTube (d) LiveJournal

Figure 4: Impact of adding more colors

From curve of 1 1 we can see that the clustering accuracy increases obviously with the parameter λ_1. This indicates that even on traditional graph clustering problem, our algorithm still outperforms the basic random walk method. For curve 2 1, initially its accuracy is higher than 1 1 because of the information introduced by more seed nodes. However, the accuracy drops slightly after $\lambda_1 > 10^3$, because excessively strong attraction around the two individual seed nodes separates the target community. The attraction even works when seeds nodes are not from the same community, but labeled as the same color (curve 2 2).

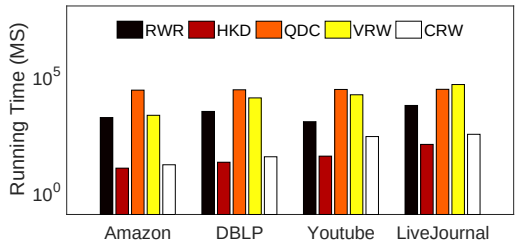


Figure 5: Running time comparison

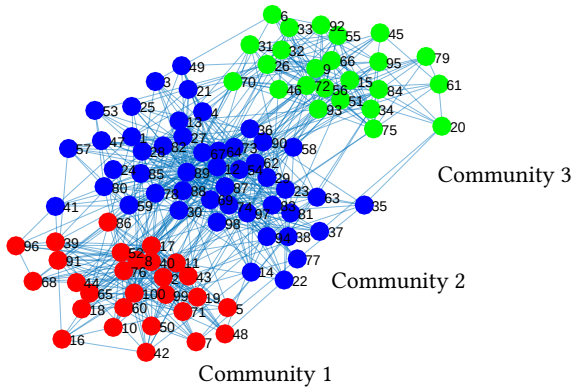


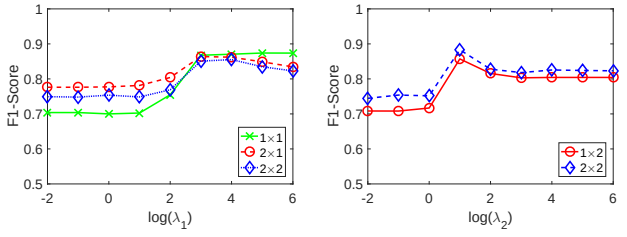
Figure 6: A 100-node graph generated by the LFR benchmark. The red, blue, green nodes are from 3 distinct ground-truth communities.

Similar results are observed by using repulsion force only and set $\lambda_1 = 0$. In experiments shown in Fig. 7b, we select 1 or 2 nodes from two adjacent communities and label them with different colors. The figures illustrate that the repulsion enhances clustering accuracy, as walkers from the two different communities are divided by distinct colors. However, overdose of repulsion ($\lambda_2 > 10$) drives walkers from the border of communities and decreases the accuracy.

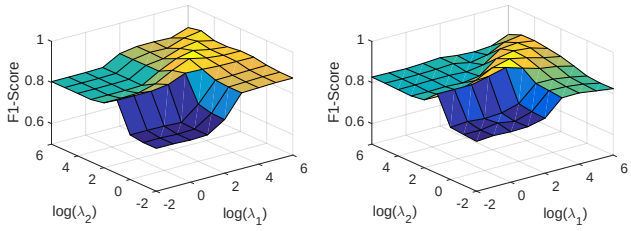
Finally, we combine both attraction and repulsion forces together (shown in Fig. 7c and 7d). The best accuracy results come with $\lambda_1 = 10^4$ and $\lambda_2 = 10^2$, and the performance are maintained when these two parameters keep on increasing. The favorable λ_1 and λ_2 are large because of the stabilizer "1" in color reinforcement (Eq. 3), which requires large λ_1 and λ_2 to change the behavior of random walkers.

4.4 Evaluation of Speedup Strategy

We generate a network with 1 million nodes and 15 million edges by LFR benchmark to test the impact of threshold θ on Algorithm 2. The results are presented in Fig. 8. When the threshold θ rises from 10^{-15} to 10^{-5}, the average running time drops dramatically (Fig. 8a). Note that when $\theta \neq 0$, Algorithm 2 reduces to a non-decaying version of Algorithm 1. The total speedup is more than 2 orders of magnitudes, when threshold increases to above 10^{-5}.

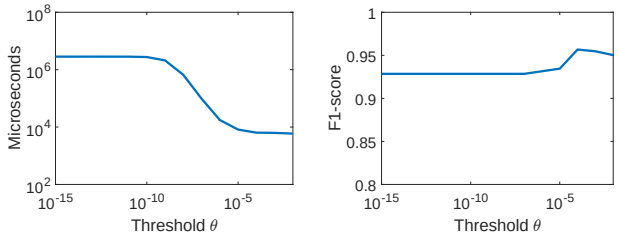


(a) Attraction only **(b) Repulsion only**

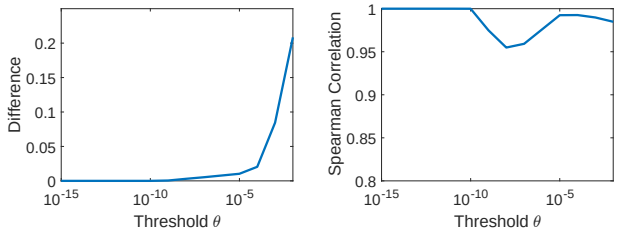


(c) 1 2 attraction and repulsion **(d) 2 2 attraction and repulsion**

Figure 7: Impact of color attraction and repulsion. λ_1 is the attraction coefficient and λ_2 is the repulsion coefficient.



(a) Running time **(b) Local clustering accuracy**



(c) Vector difference **(d) Spearman correlation**

Figure 8: Impact of threshold θ on Algorithm 2

In the meantime, the clustering accuracy remains intact and even increases from $\theta = 10^{-5}$. Looking closely, the recall of clustering remains stable while the precision rises after $\theta > 10^{-5}$. This is because the nodes of the target local clustering (true positive) are not faraway from seed nodes. Larger threshold θ makes the search more local: thus irrelevant nodes will not be enclosed, which reduces the false positive.

We also examine the difference between the output vectors from Algorithm 1 and Algorithm 2. From Fig. 8d, the summation of absolute difference value is only 1.03% at $\theta = 10^{-5}$ (Fig. 8c). Does

this difference substantially change the orders of nodes, which is crucial to clustering result? Spearman's rank correlation reveals that it is not the case (Fig. 8d). In calculating correlation, we only compare non-zero entries from both vectors. Though the correlation drops a little between 10^{-10} and 10^{-5} due to the noise introduced by the speedup strategy, it returns to above 0.98 after $\theta > 0{:}5$, where the correlation is limited to local clusters only. This indicates that the heuristic of Algorithm 2 has very little deflection on the order of nodes in the target local cluster.

In summary, the introduction of threshold θ largely reduces the running time of the proposed algorithm, and the clustering accuracy is enhanced. We adopt $\theta = 10^{-5}$ as a trade-off between running time and accuracy in our other experiments.

5 CASE STUDY

In this section we present two case studies to examine nature of the proposed colored random walk algorithm, explain why CRW works with even a single color, and provide more examples on how CRW divides dense subgraph into meaningful partitions.

5.1 Karate Club Social Network

In this section we look into the colored random walk (CRW) method by the karate club network [23], to see how CRW improves the clustering accuracy even by a single seed node.

The karate club network contains 34 members (nodes) (Fig. 9). Edges are interactions among members. A conflict between the club instructor (node 1) and the president (node 34) divides the club into two separate groups (labeled by blue and red, respectively), which serves as clustering ground truth. It is obvious that node 1 and node 34 are influential nodes with large degrees in the two groups, connecting to many other members.

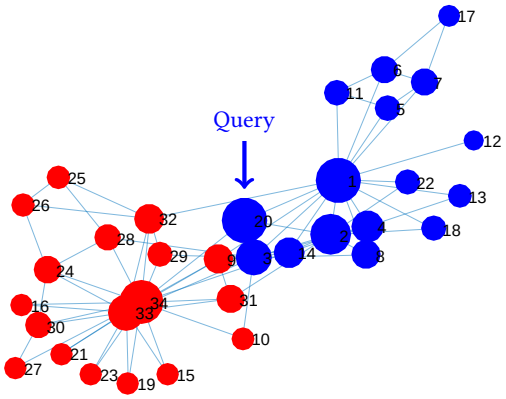
We choose node 20 as the seed node for both CRW and random walk with restart (RWR). It is a hard task as node 20 is on the border of these two clusters, though it is categorized to the blue cluster. The proximity scores are represented by size of nodes in Fig. 9.

From Fig. 9a, we can see that the proximity score distribution is relatively even on all nodes. This is not desirable since it is not easy to distinguish the node membership by the flat proximity scores. In contrast, proximity scores concentrate on the blue community, especially the central influential nodes 1 (Fig. 9b). These nodes accumulate more colors (proximity scores) and have more attraction to the colored walkers, and essentially helps to divide the two clusters.

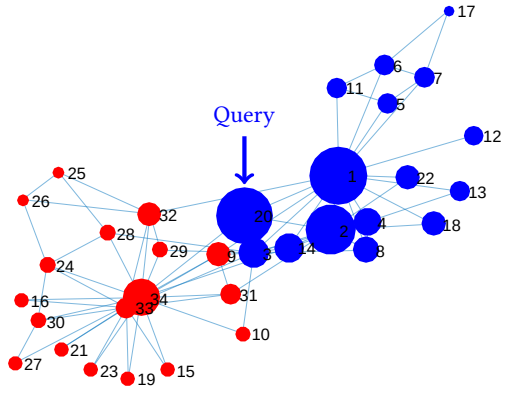
5.2 Brain Activity Network

Fig. 10a depicts another example where prior knowledge about the difference of seed nodes is critical. It is part of a brain activity network where each node is a cortical area and edges are functional associations [5]. The nodes in this network are from Broadman area 6 (BA6) and Broadman area 13 (BA13). Since these two areas are closely connected with each other, conventional methods cannot divide these areas. Once knowing the difference of seed nodes (e.g., node 7 and 14), it becomes easier to separate the two Broadman areas. The clustering result can be viewed by 3D coordinates[1] in Fig. 10b.

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



(a) Proximity scores by random walk with restart (RWR), $\alpha = 0{:}9, T = 10$



(b) Proximity scores by colored random walk (CRW), $\alpha = 0{:}9, T = 10, \lambda_1 = 70, \lambda_2 = 0, \psi(t) = 1$

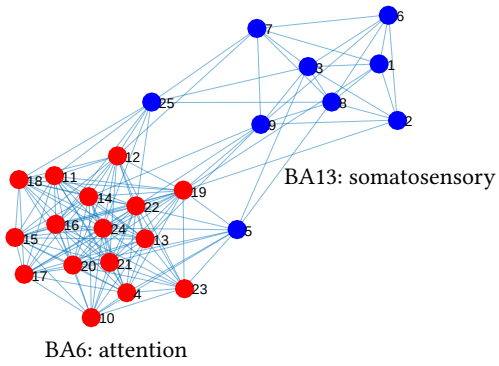
Figure 9: Comparison of proximity scores of RWR and CRW on the karate club network

6 CONCLUSION

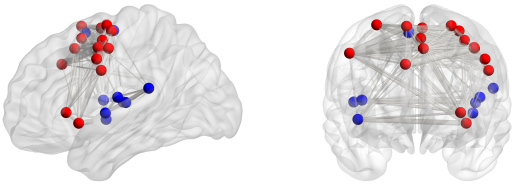
In this paper we advance a colored random walk method to solve the constrained local graph clustering problem, where the communities membership of seed nodes are known as prior knowledge. The proposed method dynamically adjusts node weights for each color during the random walk process, and the updated node weights help to trap the colored random walkers into the corresponding communities. Experiments show that the performance of our method is above the state-of-the-art approaches, even for conventional single-color and single-seed scenario. With more seed nodes and colors available, the performance of the proposed method is significantly better.

ACKNOWLEDGEMENT

This work was partially supported by the National Science Foundation grant IIS-1707548.



(a) Part of brain activity network



(b) 3D coordinates of the network

Figure 10: Each node is an area in brain. Red and blue nodes are clusters divided with prior knowledge that node 3 and node 24 are different

REFERENCES

[1] Reid Andersen, Fan Chung, and Kevin Lang. 2006. Local graph partitioning using pagerank vectors. In *FOCS*.

[2] Michel Benaïm et al. 1997. Vertex-reinforced random walks and a conjecture of Pemantle. *The Annals of Probability* 25, 1 (1997), 361–392.

[3] Yuchen Bian, Jingchao Ni, Wei Cheng, and Xiang Zhang. 2017. Many Heads are Better than One: Local Community Detection by the Multi-Walker Chain. In *Data Mining (ICDM), 2017 IEEE International Conference on*. IEEE, 21–30.

[4] Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. 2006. Semi-Supervised Learning. Adaptive Computation and Machine Learning series.

[5] Nicolas A Crossley, Andrea Mechelli, Petra E Vértes, Toby T Winton-Brown, Ameera X Patel, Cedric E Ginestet, Philip McGuire, and Edward T Bullmore. 2013. Cognitive relevance of the community structure of the human brain functional coactivation network. *Proceedings of the National Academy of Sciences* 110, 28 (2013), 11583–11588.

[6] Wanyun Cui, Yanghua Xiao, Haixun Wang, and Wei Wang. 2014. Local search of communities in large graphs. In *SIGMOD*.

[7] Roger A Horn, Roger A Horn, and Charles R Johnson. 1990. *Matrix analysis*. Cambridge university press.

[8] Kyle Kloster and David F Gleich. 2014. Heat kernel based community detection. In *SIGKDD*.

[9] Isabel M Kloumann and Jon M Kleinberg. 2014. Community membership identification from small seed sets. In *KDD*.

[10] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. 2008. Benchmark graphs for testing community detection algorithms. *Physical review E* 78, 4 (2008), 046110.

[11] Rui Liu, Wei Cheng, Hanghang Tong, Wei Wang, and Xiang Zhang. 2015. Robust Multi-Network Clustering via Joint Cross-Domain Cluster Alignment. In *ICDM*.

[12] Qiaozhu Mei, Jian Guo, and Dragomir Radev. 2010. Divrank: the interplay of prestige and diversity in information networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. Acm, 1009–1018.

[13] Jingchao Ni, Hongliang Fei, Wei Fan, and Xiang Zhang. 2017. Automated Medical Diagnosis by Ranking Clusters Across the Symptom-Disease Network. In *Data Mining (ICDM), 2017 IEEE International Conference on*. IEEE, 1009–1014.

[14] Jingchao Ni, Hongliang Fei, Wei Fan, and Xiang Zhang. 2017. Cross-Network Clustering and Cluster Ranking for Medical Diagnosis. In *ICDE*.

[15] Jingchao Ni, Mehmet Koyuturk, Hanghang Tong, Jonathan Haines, Rong Xu, and Xiang Zhang. 2016. Disease gene prioritization by integrating tissue-specific molecular networks using a robust multi-network model. *BMC bioinformatics* 17, 1 (2016), 453.

[16] Robin Pemantle et al. 2007. A survey of random processes with reinforcement. *Probability surveys* 4 (2007), 1–79.

[17] Satu Elisa Schaeffer. 2007. Graph clustering. *Computer science review* 1, 1 (2007), 27–64.

[18] Mauro Sozio and Aristides Gionis. 2010. The community-search problem and how to plan a successful cocktail party. In *KDD*.

[19] Hanghang Tong, Christos Faloutsos, Brian Gallagher, and Tina Eliassi-Rad. 2007. Fast best-effort pattern matching in large attributed graphs. In *KDD*.

[20] Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. 2006. Fast random walk with restart and its applications. (2006).

[21] Marc A Van Driel, Jorn Bruggeman, Gert Vriend, Han G Brunner, and Jack AM Leunissen. 2006. A text-mining analysis of the human phenome. *European journal of human genetics* 14, 5 (2006), 535–542.

[22] Yubao Wu, Ruoming Jin, Jing Li, and Xiang Zhang. 2015. Robust local community detection: on free rider effect and its elimination. *Proceedings of the VLDB Endowment* 8, 7 (2015), 798–809.

[23] Wayne W Zachary. 1977. An information flow model for conflict and fission in small groups. *Journal of anthropological research* 33, 4 (1977), 452–473.

[24] Denny Zhou, Olivier Bousquet, Thomas N Lal, Jason Weston, and Bernhard Schölkopf. 2004. Learning with local and global consistency. In *Advances in neural information processing systems*. 321–328.

[25] Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. 2003. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*. 912–919.