LCD-RW: Local Community Detection by Random Walk in Social Networks

Arezoo Rajabi[1] and Hamid R. Rabiee[1, a)]

Department of Computer Engineering, Sharif University of Technology

(Dated: 18 November 2014)

In this paper, we introduce a novel local community detection algorithm based on the random walk probability distribution. After selecting an initial node, the proposed algorithm adds a *group of nodes* to the community of initial node based on normalized random walk probability distribution. Then it examines the explored community to decide on termination of the algorithm. Finally, it utilizes a global community detection algorithm to remove the irrelevant nodes from the explored community. Empirical results on artificial graphs and datasets from real networks showed that the proposed algorithm returns well-structured communities and is capable of escaping from local extremums.

Keywords: Community Detection, Random Walk, Local Algorithms

[a)]Corresponding author - rabiee@sharif.edu - http://sharif.edu/~rabiee.

Lead Paragraph:

In recent years, there has been a tremendous amount of effort towards introducing community detection algorithms, many of which are based on the assumption of having a full-knowledge of the network topology. However, most complex networks are extremely large and accessing their global information is often infeasible. Hence, local community detection algorithms have been introduced to alleviate this problem. In this approach, the community of a given node is found by using its local information. In each iteration, local algorithms explore a small part of a graph to gather information of the neighboring nodes. Then, the best node among the explored nodes is selected and added to the community of the given initial node. Because of partial knowledge about the topology of network, previously proposed algorithms usually trap in a local extremum, and hence cannot explore all nodes in the community of initial nodes. In this paper, we propose a novel algorithm that can escape from local extremums by adding a *group of nodes*, instead of choosing a single node in each iteration. In order to evaluate the proposed method, extensive simulations are conducted on artificial graphs and real network datasets. Moreover, results are compared with the communities found by OSLOM[1] which is a well-known community detection algorithm.

I. INTRODUCTION

The fast growth of complex networks and their wide range of applications have made the analysis of these networks a prevalent research area. The growth of interest in complex networks analysis and investigating their structural features have led to studies towards community detection in such networks. In contrast to random networks, in many complex networks, appearance of links is not random. Based on this fact, complex networks have a modular structure where each module is called a community. A community, is a group of nodes which are connected to each other densely, while their connection to other nodes is sparse[2]. There are two criteria of concern to define a community structure in a graph:

1. **Global Criterion:** This criterion evaluates all communities of a graph by considering

the dependency amongst them. A common global criterion is *modularity* that specifies the difference between a network with its determined community structure and a random graph[3,4].

2. **Local Criterion:** In this criterion, each community is considered as an independent entity and is evaluated by ignoring the rest of the graph[2]. *Local Modularity*(R)[5], and *Conductance*(ϕ)[6] are commonly being used as local criteria.

Based on the information used for community detection, there are two main approaches for finding communities. In the first approach, all communities of a graph are detected with the aim of maximizing (minimizing) a global criterion, which needs comprehensive information about the graph[7–9]. These methods are called *global community detection algorithms*. The complexity of these methods is high and applying them to large networks is impractical. In addition, fast growth of social networks makes access to the whole topology information of these networks infeasible. Moreover, in some applications we are only interested in communities of a special set of nodes and acquiring information about all communities is not desirable. To alleviate some of these problems, many *local community detection algorithms* have been proposed. In this type of approaches, a community for a given node(s) is found by expanding this node, and the goal is to maximize (minimize) a local criterion[5,10–15]. These algorithms have two main steps and an optional step in each iteration:

1. **Node selection step:** In this step, local algorithms select the most appropriate node by utilizing a criterion and add it to the community of the initial node.

2. **Decision step:** In this step, a criterion is used to determine when to stop adding nodes to a community.

3. **Filtering step:** Some local algorithms check the detected community to eliminate the inappropriate nodes. This step is optional and many local algorithms do not employ this step.

The previous local community detection algorithms usually trap in local extremums. As a consequence, local algorithms tend to terminate sooner than expected, and fail to extract all the nodes that belong to the community of the initial node. To address this problem, we introduce a novel algorithm that adds nodes to the community in a different manner.

The rest of this paper is organized as follows. In section II, we introduce the previous works and describe their drawbacks. In section III, we explain the random walk probability distribution. In section IV, we present an efficient algorithm to find a community for a given node. There are two approaches to evaluate local algorithms. We describe these approaches in section V. The experimental results and comparisons with the previous works are provided in section VI. Finally, the concluding remarks are presented in section VII.

II. RELATED WORK

Each network can be represented by a graph $G = (V, E)$ where V is the set of vertices, and E is the set of edges. Nodes in a network are vertices of the graph (for example a web page in WWW), and links between nodes are the edges of the graph. We assume that graphs are undirected and unweighted. As discussed earlier, local community detection methods have been introduced to find communities for a given set of nodes. They start from an initial node, and select the most appropriate nodes (specified by a criterion) from its neighbors as the members of the corresponding community, called C. After specifying the most appropriate node, these algorithms decide whether to add that node to the community or not. If the aforementioned node is added, then a node from the neighbors of the nodes in C is selected in a similar way, and this process is continued until a termination criterion is met.

Each main step of local algorithms uses a criterion. In the following, we review the criteria used in the two primary steps and their shortcomings.

A. Node selection step

Local community detection algorithms start by iteratively adding nodes to an initial community C which is an empty set. In each iteration, the algorithm looks for the most appropriate node among the neighbors of C, called N, which is defined as:

$$N = \{x \in V | (x, y) \in E, y \in C, x \notin C\} \tag{1}$$

There are two main approaches for selecting the best node and adding it to the community C:

1. **Per community evaluation:** In this approach, a local community criterion is used to choose the most appropriate node, and the node that improves this criterion more than the others, is selected. For instance, by applying a local criterion such as Local Modularity (R), a node with the highest amount of improvement in R is selected.

2. **Per node evaluation:** In this approach, a criterion that is based on the number of indegree and outdegree edges of each node, is used. Edges of each node are categorized into two classes: indegree and outdegree edges. An indegree edge ends into a node in C while an outdegree edge ends into a node that is not in C. These criteria assign weights to the indegree and outdegree edges of each node in different manners. At last, a node with the highest(lowest) total weight is selected. Some of the well known criteria in this category are investigated in the following.

   - **Outwardness**[11]**:** This measure assigns a weight to indegree and outdegree edges of a candidate node that is proportional to the degree of that node. Each indegree edge of node v has the weight $\frac{-1}{k_v}$, and each outdegree edge has the weight $\frac{1}{k_v}$, where k_v is the degree of node v. Therefore, the competency of node v is computed as:

   $$\Omega_v = \frac{(k_v^{out} - k_v^{in})}{k_v} \tag{2}$$

   In which, k_v^{out} and k_v^{in} are the number of outdegree and indegree edges of node v, respectively.

   $$k_v = k_v^{out} + k_v^{in} \tag{3}$$

   Finally, a node with the smallest outwardness is selected and added to the community. It is important to note that this criterion does not distinguish between outdegree edges pointing to the nodes in N, and the unknown nodes (U).

   - **MaxActivation**[16,17]**:** This measure allocates an activation value to the initial node. In each step, this value propagates through the nodes in $(C \cup N)$. Each node receives an activation from its neighbors which have the same distance or one step closer to the initial node.

The main problem of node selection methods is their tendency towards low degree nodes. Because, most of links of low degree nodes end inside the found community, they usually

achieve better scores. Because of this problem, most of the local community detection algorithms trap in local extremums. In other words, local criteria are unaware of the global structure of networks, and are based on maximizing the number of indegree edges and minimizing the number of outdegree edges of a community. As a result, a high degree node which mostly has more outdegree than indegree edges, is not added until most of its neighbors are added to the found community. For instance, in Fig. 1, node i is the initial node, and there are four candidate vertices ($\{X, Y, V, W\}$) that can be added to C. For example,

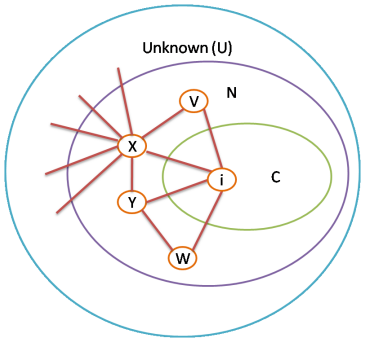


FIG. 1: Vertex i is the initial node. $\{X, V, Y, W\}$, are candidate nodes for expanding C.

Local Modularity (R) considers the difference between indegree and outdegree edges which causes low degree nodes to be added to the community first. Outwardness also considers the difference between indegree and outdegree edges, normalized by the degree of the node. In this case, low degree nodes usually have a lower outwardness. Unlike the aforementioned

TABLE I: The competencies that each measure gives to candidate nodes.

	X	Y	V	W	Selected node
$\triangle R$	0.1	0.2	0.25	0.25	V or W
Outwardness	0.875	0.333	0	0	V or W
MaxActivation($\delta = 0.5$)	1	1	0.75	0.75	X or Y

measures, MaxActivation distinguishes outdegree edges based on their ending point. This criterion scores edges that end into other candidate nodes (N), and ignores the links ending into U. Hence, despite the superiority of node Y over node X, MaxActivation treats them indifferently.

We divide local algorithms into two categories based on the node selection step. In the first

category, candidate nodes are sorted by their competency, and added until a termination criterion reaches a local extremum. The competency of each candidate node is computed once, and adding a node to C does not affect the competency of other nodes. For example, in the algorithm introduced by Andersen et al.[10,18], competencies of candidate nodes are specified by normalized random walk probabilities and computed once. Moreover, the MaxActivation criterion can be computed once at the beginning, because the competency of nodes only depend on the initial node. These algorithms which we call *static methods*, decrease the calculation complexity at the cost of decreased Precision. In the second category, unlike static methods, the competency of candidate nodes depend on the nodes in the found community, and should be updated after adding each node[5,11]. We call these algorithms *dynamic methods*. Although, the complexity of these methods is higher than static methods, dynamic methods benefit from higher Precision. Moreover, because of the small size of a community compared to the whole graph, this complexity is negligible, causing dynamic algorithms to be more desirable than static ones.

B. Decision step

In each iteration, after selecting the most appropriate node, a termination criterion is utilized to determine whether to add this node or not. A local criterion is usually used as a termination criterion. The goal of local algorithms is to minimize(maximize) this criterion. Unfortunately, in many cases, utilizing a local criterion causes a small set of nodes to be bounded by a cut which is smaller than the cut of the objective community, leading local algorithms to trap in a local minimum(maximum)[19]. Therefore, the found community is smaller than the corresponding real community. For instance, in Fig.2, there is a cut of size 3 in C_1 that is smaller than the cut size between C_1 and C_2 which is equal to 4. However, nodes of $\{1, 2, 3\}$ belong to C_1.

C. Filtering step

As mentioned before, the main problem of local community detection algorithms is entrapping in local extremum points. To alleviate this problem, some of the methods tem-

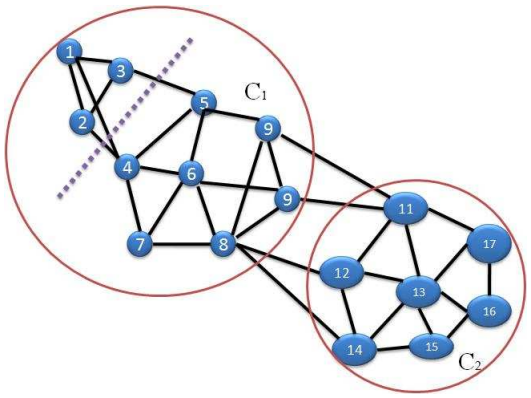


FIG. 2: There is a cut in C_1 that is smaller than the cut between two communities of C_1 *and* C_2.

porarily add the high degree nodes to the found community, and run their algorithm until the termination criterion is met. Then, they recheck the termination criterion and remove the high degree nodes that might have degraded the performance of the algorithm.

III. PRELIMINARIES

In this paper, we use the *lazy random walk* to propagate initial probabilities assigned to members of the found community C. Proportional to their degree, an initial probability is assigned to nodes in C:

$$P_0(i) = \begin{cases} \frac{deg(i)}{\sum_{i \in C} deg(i)} & \text{if } i \in C \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where $deg(i)$ denotes the degree of node i. We apply a random walk with a limited number of steps. The probability of being in each node after t steps, is defined by p_t which is computed as:

$$P_t = W^t P_0 \tag{5}$$

where W is the transition matrix for the lazy random walk and is defined as:

$$W = \frac{1}{2}(I + AD^{-1}) \tag{6}$$

where A is an adjacency matrix, and D is a diagonal matrix in which $D_{ii} = deg(i)$. A random walk is biased towards high degree nodes which may have high outdegree edges compared to indegree edges. In order to remove this tendency, the probability of being at

each node in $(C \cup N)$ after t steps is normalized by its degree.

$$r_t(v) = \frac{p_t(v)}{deg(v)} \tag{7}$$

IV. PROPOSED METHOD

In this section, we introduce a novel algorithm, called LCD-RW, that can escape from local extremums. This algorithm consists of three main steps. In the following, we explain each step in detail.

A. Node selection step

As mentioned before, to select the most appropriate node, the criterion used in the node selection step must weight the edge types properly, and discern between outdegree edges that end into N and U. The proposed algorithm applies the normalized probability distribution related to the lazy random walk. In that sense, a truncated random walk is utilized to distribute the probability(P_0), allocated to the nodes in the found community. Because of the modular structure in most complex networks, lazy random walk hardly exits from a community[20]. In other words, sparse connection between communities prevents a lazy random walk from rapidly leaving a community. Therefore, the probability of remaining in the community of the initial node, after a limited number of steps, is more than being in a node outside that community. Assume that the average ratio of outdegree to indegree edges for the nodes of a community is $\mu(=\frac{k_v^{out}}{k_v^{in}})$, and the probability of remaining in a node is p. Therefore, a lazy random walk goes to another node by a probability equal to $(1-p)$. Thus, the probability of leaving a node and going to another node in a different community in one step would be:

$$p_{out} = (1-p)\frac{k_v^{out}}{k_v^{out} + k_v^{in}} = (1-p)\frac{\mu}{\mu+1} \tag{8}$$

Therefore, the probability of leaving a community at step t for the first time is:

$$P_{out}(t) = (1-p_{out})^{(t-1)} p_{out} \tag{9}$$

By increasing the number of steps, the probability of being in each node converges to a fix value that is proportional to the degree of the node. Therefore, we apply a limited number of steps to calculate the competency of each candidate node. The proposed method for

selecting an appropriate node is depicted in Algorithm 1.

ALGORITHM 1: Selecting the best node for adding to the community

SelectNextNode $(G(V,E), C, t)$

{

$ExploredNodes \longleftarrow BFS(G, C, t)$

$CandidateNodes \longleftarrow Neighboars\ of\ C$

$\%P\ is\ a\ probability\ vector$

Foreach $v \in C \cup ExploredNodes$ **do**

if $v \in C$ **then**

$P_0(v) = \frac{deg(v)}{\sum_{n \in C} deg(n)}$

else

$P_0(v) = 0$

end if

end for

$compute\ P_t$

$nexNode \longleftarrow\ node(x)\ with\ highest\ value$

$of\ \frac{P_t(x)}{deg(x)}\ in\ CandidateNodes$

$Return\ nextNode$

}

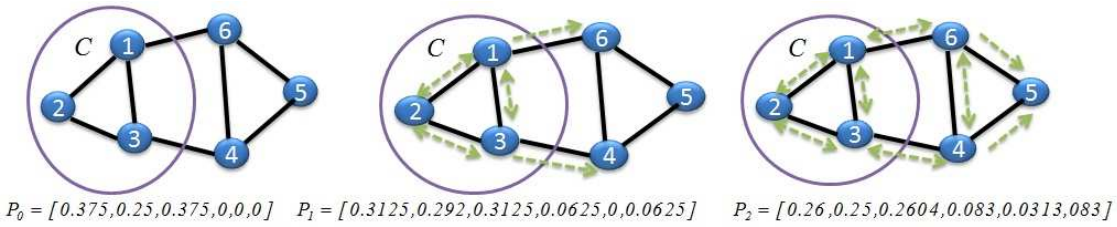


FIG. 3: The initial probabilities(P_0) are assigned to the members of the found community, C, proportional to their degree. P_1 and P_2 show the probability of each node for $step = 1\ and\ 2$. Dashed arrows indicate the possible path for a random walk for each step.

In this pseudo code, the input parameter, t, determines the number of steps for a random

TABLE II: The conditions which must be fulfilled to add a candidate node(s) to C in Fig. 4.

$C' =$	$C \cup \{1\}$	$C \cup \{2\}$	$C \cup \{1,2\}$
$\phi(C') =$	$\frac{y+1}{(y+1)+2(x+2)}$	$\frac{y+1}{(y+1)+2(x+1)}$	$\frac{y}{y+2(x+4)}$
$\phi(C') < \phi(C)\ \ if$	$x < 2y$	$x < y$	$\surd$

walk. The minimum number of steps in which outdegree edges of candidate nodes(N) that end into nodes in N participate in probability distribution is equal to 2, since in the first step the candidate nodes receive a non zero probability, and in the next step they can distribute this probability to their neighbors. As shown in Fig. 3, the edge (6,4) participate in probability distribution for *steps* ≥ 2. Moreover, a high number of steps causes the probability of being in each node to converge to a fix value, and more nodes are needed to be explored. Therefore, a small value bigger than one, is suitable for this parameter.

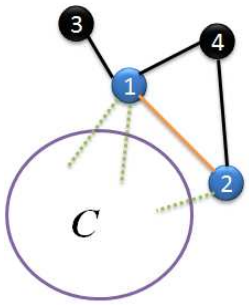


FIG. 4: C is the found community for an initial node. Nodes 1 and 2 are candidate for addition to C. Dashed lines show indegree edges of candidate nodes.

B. Descision step

As mentioned before, one of the main problems in local algorithms is entrapping in local extremums. The main reason for this problem is that local algorithms tend to add low degree nodes, and edges of low degree nodes usually end into the found community. Hence, adding these nodes causes the total number of indegree edges of the found community to increase, while the total number of outdegree edges decreases. Therefore, their competency are more

than high degree nodes. On the other hand, high degree candidate nodes which generally have more outdegree edges than indegree ones are not added, while most of their outdegree edges end into other candidate nodes, and by adding a group of high degree nodes, the total number of outdegree links of the found community is considerably decreased. For example, in Fig. 4 nodes 1 and 2 are two candidate nodes to be added to C. Suppose conductance is used as the termination criterion. The most appropriate node (according to any node selection criteria) will be added to C if it decreases the conductance of C, otherwise C will be returned. Assume that C contains x indegree edges, and $y(y = 3)$ oudegree edges, then its conductance is $\phi = \frac{y}{2x+y}$. The changes in conductance of C by adding candidate node(s) is shown in TABLE II. As described in this table, neither of these two candidate nodes can be added to C unless they satisfy some conditions (for instance, the number of indegree edges of C should be less than its external edges). Moreover, by adding both nodes, $\{1, 2\}$, the conductance of C reduces.

To overcome this problem in the proposed method (LCD-RW), we add a group of vertices which causes the probability of entrapping in local extremums to decrease, and the tendency of selecting low degree nodes become insignificant. The proposed algorithm creates a temporary community, and nodes are added to this community until the algorithm can not find a node that improves the termination criterion (conductance). Then, we decide whether to join the temporary set to the initial found set (found community) or not. This temporary set would be added to the found community if it decreases the conductance of the found community. If joining occurs, then LCD-RW continues, and a new temporary set is created, otherwise it stops.

C. Filtering step

By adding a group of nodes instead of a single node, we decrease the probability of falling into local extremums. However, adding a group of nodes to C usually causes the neighbor communities to merge. Assume C and C_{temp} are two neighbor communities. Also consider x, y as the number of indegree and outdegree edges of C, and x', y' as the number of indegree and outdegree edges of C_{temp}, respectively. We assume the number of common links between C and C_{temp} as two neighboring communities is $k(k \geq 1)$. By merging these two groups, the conductance of the new group is:

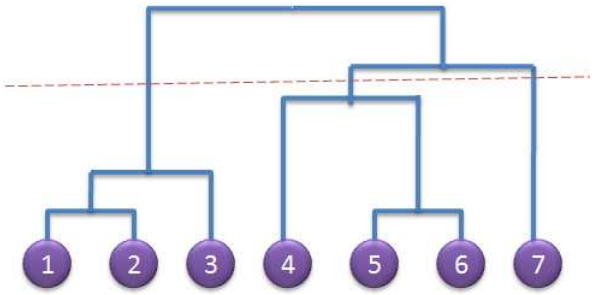


FIG. 5: Dendrogram of the hierarchical structure of a graph. At the bottom of the dendrogram, each node is a separate module and by moving upwards, modules are aggregated and larger ones are created.

$$\phi(C_1 \cup C_2) = \frac{y + y' - 2k}{y + y' - 2k + 2(x + x' + k)} \tag{10}$$

The new group will have a lower conductance if the condition described in equation 11 is fulfilled:

$$\begin{aligned}
&\frac{y + y' - 2k}{y + y' - 2k + 2(x + x' + k)} < \frac{y}{y + 2x} \Rightarrow \\
&(2x + y)(y + y' - 2k) < y(y + y') + 2y(x + x') \Rightarrow \\
&2x(y + y' - 2k) + y(y + y') - 2ky < y(y + y') + 2y(x + x') \Rightarrow \\
&x(y + y' - 2k) - ky < y(x + x') \Rightarrow \\
&-k(2x + y) < y(x + x') - x(y + y') \Rightarrow \\
&k > \frac{x(y + y') - y(x + x')}{2x + y} \Rightarrow \\
&k > \frac{xy' - yx'}{2x + y}
\end{aligned} \tag{11}$$

Note that $x, y, x', and y' > 0$, hence we can conclude that:

$$\frac{xy' - yx'}{2x + y} < \frac{y'}{2} \tag{12}$$

From equations 11 and 12, we can show that if the majority of outdegree edges of C_{temp} end in nodes of C, $(k > \frac{y'}{2})$, C, and C_{temp} are merged. Merging some neighbor communities results in a bigger community which has a lower number of outdegree edges compared to the total outdegree of the composing communities. The reason is that merging adjacent communities converts a number of outdegree edges into indegree edges. Therefore, in most networks there is a hierarchical structure between communities that can be represented by a dendrogram as shown in Fig.5. Based on this representation, divisive and aggregative global algorithms have

been emerged. The divisive top-down algorithms try to separate communities by removing edges that connect them to each other. On the other hand, the bottom-up aggregative global algorithms, merge communities to obtain the final community. In these algorithms, initially each node is considered as a separate community and these communities are merged to improve a global criterion. In both divisive and aggregative algorithms, after constructing the dendrogram of a graph, communities are detected by horizontally cutting through the dendrogram. Cutting the dendrogram at a higher level results in low conductance groups such that each group contains one or more communities, and has its own dendrogram, while cutting the dendrogram at a low level results in more communities with high conductance. As described before, the tendency of LCD-RW to merge neighbor communities leads this algorithm to return a low conductance group which is a union of neighboring communities. These communities construct a dendrogram that is obtained by a cut on the dendrogram of the graph. Thus, the found communities by LCD-RW can be considered as an independent sub-graph that has limited links to the rest of graph. Hence, we can apply a divisive or aggregative global algorithm in the filtering step to extract the real community of the initial node. In other words, these two category of global algorithms return more remarkable communities in the sub-graph of the found community by LCD-RW.

Since the complexity of global algorithms depends on the size of the graph, we apply a threshold on the size of the community found by LCD-RW. A large threshold forces the found community to contain more sub-communities that requires more time to find the real community due to using a global algorithm, and a small threshold causes LCD-RW to fall in a local minimum. Observation on many social networks shows that the maximum size for human communities in social networks, known as the Dunbar number, is less than 150 members[21]. Moreover, online-communities have less than 60 members[22]. Hence, for most of networks, 150 is a reasonable value for the threshold.

The Algorithm 2 describes the proposed LCD-RW algorithm. The global algorithm used in the filtering step is CNM[7]. This algorithm is a greedy algorithm which constructs a hierarchical tree of communities by joining communities to optimize the modularity in each step. Moreover, the complexity of CNM is low. LCD-RW is a dynamic method, because after a node is added to the temporary found community, the competency of candidate nodes are recomputed.

ALGORITHM 2: LCD-RW Algorithm

LCD-RW $(G(V,E), v, threshold)$

{

$C \longleftarrow v_0$

while $(true)$ **do**

$S \longleftarrow v_0$

$\%S\ is\ a\ temporary\ community$

while $(a\ local\ minimum for\ \phi\ is\ not\ found)$ **do**

$nextNode = SelectNextNode(G, S, step)$

$S \longleftarrow (S \cup nextNode)$

end while

if $\phi(C \cup S) < \phi(C)\ and\ |C \cup S| < threshold$ **then**

$C \longleftarrow C \cup S$

$v_0 = SelectNextNode(G, S, step)$

$S \longleftarrow \emptyset$

else

$\%LCD - RW\ stops\ adding\ nodes\ to\ C$

$break;$

end if

end while

$C_{final} \longleftarrow CNM(G(C))$

}

V. EVALUATION MEASURES

In general, two prevalent approaches are utilized to evaluate and compare the performance of local community detection algorithms. The first approach is used when the size of network is small enough to find all the communities by a global community detection algorithm. In such a scenario, we can compare the found community for a given node with the real one which is found by a well-known global community detection algorithm. In this case, we may use Precision and Recall for evaluation purposes. Precision represents the purity of the

found community, and Recall evaluates the fraction of nodes in the real community of the initial node that are detected by a global algorithm. Moreover, we can use the F-measure which is defined as:

$$
\begin{aligned}
Precision &= \frac{|C_f \bigcap C_r|}{|C_f|} \\
Recall &= \frac{|C_f \bigcap C_r|}{|C_r|} \\
F-measure &= \frac{2 \times Recall \times Precision}{Precision + Recall}
\end{aligned}
\tag{13}
$$

Where, C_r is the real community of the given node detected by a global algorithm, and C_f is the found community by the local algorithm. Many complex networks such as social

TABLE III: The characteristics of algorithms used for comparison with the proposed method

Name	Node Selection Criterion	Termination Criterion	Type of Algorithm
MaxActivation	MaxActivation	ϕ	static
Bagrow[11]	Outwardness	R	dynamic
Clauset[5]	R	R	dynamic
LCD-RW(proposed algorithm)	Random Walk	ϕ	dynamic

networks are large scale, and we can not obtain their communities by global community detection algorithms. Hence, an alternative evaluation approach is introduced that is based on the features (such as conductance) of the found communities. In this approach, we cannot judge the goodness of an algorithm by applying these measures, and only features of the found communities can be compared with each other.

As discussed before, defining a proper termination criterion is a critical issue in local algorithms, and optimizing a local criterion does not guarantee the optimization of a global criterion. In summary, using a local criterion for termination causes local algorithms to return a set of nodes that might be considerably different from the community obtained by the global algorithms[23]. In this paper, we use both approaches to evaluate the results.

TABLE IV: The characteristics of real-word networks. ♯ Node, ♯ Edge, ♯ Community and Avg(ϕ) are number of nodes, number of edges, number of communities in each network, and average conductance of communities, respectively.

Name	♯ Node	♯ Edge	♯ Community	Avg(ϕ)
US Airline	332	2126	8	0.356
Football	115	613	11	0.316
Jazz	198	2742	13	0.558
Net Science	1589	2742	196	0.0354
Protein	95	213	6	0.08
Roget	1022	3648	34	0.472

VI. RESULTS

In this section, we evaluate the proposed algorithm on real network datasets and artificial graphs. According to the first evaluation approach, we should extract all communities in a network and compare the found community for the given node with the real one. We applied the OSLOM algorithm[1] to detect communities. The proposed algorithm was compared with algorithms presented in table III. We describe local algorithms by the criterion used in each step and the algorithm type (static or dynamic).

We applied our algorithm on different real networks. The biological network used in our simulations was Protein[24]. Football[25], Jazz[26] and Net Science[27] networks are social networks that were used to evaluate our method. In addition, we used US Airline[28] as a technological network and Roget as an information network[28]. For non-social networks, we used a threshold that was bigger than the size of networks. The characteristics of these datasets are shown in TABLE IV. In general, a lower average value of ϕ for a graph shows that it has a strong community structure.

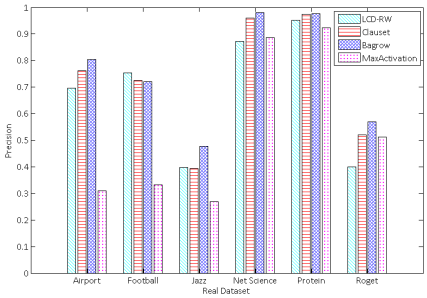
In order to construct benchmark graphs we used the LFR[29,30] method. In this method, graphs with various community structures can be generated. This method takes two input parameters; first the number of nodes in the graph, and second a parameter named μ. This parameter indicates the ratio of the number of outdegree edges to the number of indegree edges of nodes. In other words, it specifies the strength of the community structure in a

graph. Graphs with smaller values of μ have stronger community structures. In our simulations, we created graphs with one thousand nodes and different μs ($\{0.1, 0.15, 0.2, \cdots, 0.5\}$). For datasets with more than 300 nodes, we randomly sampled with a rate of 0.1 for selecting the initial nodes and executed the algorithms on those nodes.
As described, since a higher value for step does not considerably increase the Precision of the algorithm, and boost the number of candidate nodes significantly, we set its value to 3 in our simulations.

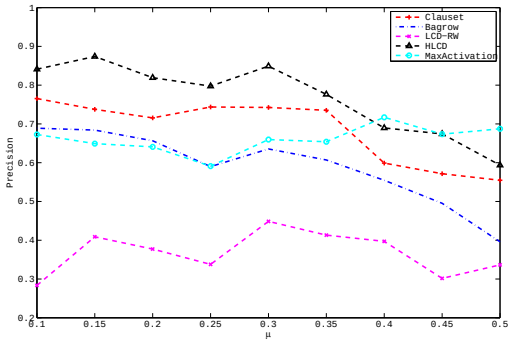
A. Precision, Recall and F-Measure

Experimental results for real-world networks are represented in Fig. 6a, 7a and 8a. As shown in these figures, in general LCD-RW has more Recall and F-measure compared to the other algorithms. MaxActivation, Clauset and Bagrow fall in a local optimum and can not return all nodes of the goal community which causes their Recall to be low. Moreover, MaxActivation utilizes a criterion that tends to add high degree nodes to a community which have more indegree edges compared to other candidate nodes. However, these nodes have high outdegree edges. As a result, this algorithm can not determine nodes which are in the goal community and its Recall and Precision are both low.
The average conductance of communities is a measure that represents the *strongness* of a community structure in a network. Communities with a high conductance, possess more outdegree edges and detecting the nodes belonging to the goal community becomes difficult. Simulation results showed that none of the algorithms work well on US Airline, Jazz and Roget datasets because of the high average conductance of the communities in these networks.

Because of entrapping in local extremums, the previous algorithms have higher Precisions at the cost of lower Recalls. The high Precision of these algorithms is caused by choosing a small group of nodes that are close to the initial one. For example, in the Protein dataset, Clauset, MaxActivation and Bagrow methods achieve a Precision close to one, but their Recall is lower than 0.5. The average size of communities in this network is ($95/6 \simeq 16$) and the average node degree is $\frac{213\times 2}{95} \simeq 6$, while the average size of the found communities by these algorithms is 8. Therefore, they return a limited number of nodes as the community members of the initial node. For instance, in the Protein dataset of Fig. 9, if an initial



(a)

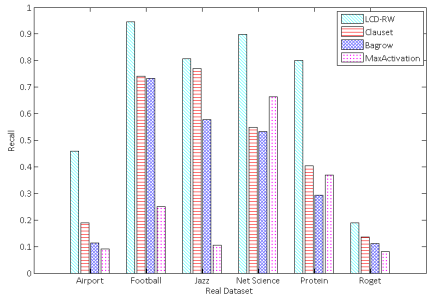


(b)

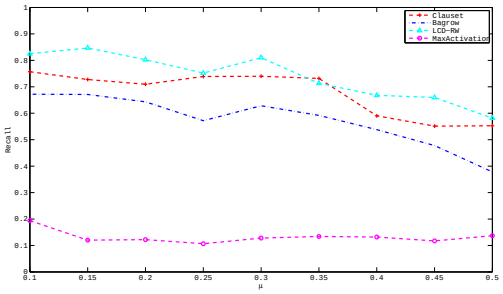
FIG. 6: Precision of algorithms for real world and LFR datasets.

node is selected from any community except $C1$, most local algorithms will detect the community of the initial node with high Precision and Recall due to strong connections between the neighboring nodes. However, if the initial node is selected form $C1$, the low density of edges in this community causes the previous local community detection algorithms to trap in a local extremum. For example, if we give vertex 9 from $C1$ as an initial node to three algorithms which use local modularity, Outwardness and MaxActivation as their node selection criterion. Also, we give this node to LCD-RW which adds a group of nodes as a batch to the found set of C, whereas other algorithms add a single node in each iteration. We also set conductance as a termination criterion for all algorithms.

Changes in conductance as a termination criterion in the decision step is shown in Fig.10. Also Fig. 11 and Fig. 12 illustrate Precision and Recall of different algorithms respectively. As shown in Fig. 10, the conductance diagram contains some local minimums. LCD-RW falls in a local minimum after other algorithms. Thus, as shown in Fig. 12, the Recall resulted from entrapping in a local minimum for the community found by LCD-RW is more



(a)



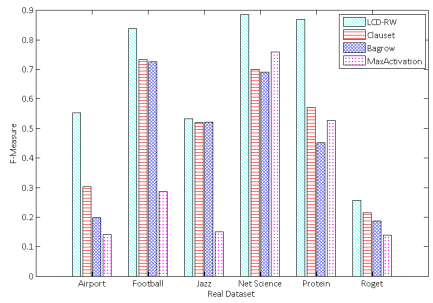
(b)

FIG. 7: Recall of algorithms for real world and LFR datasets.

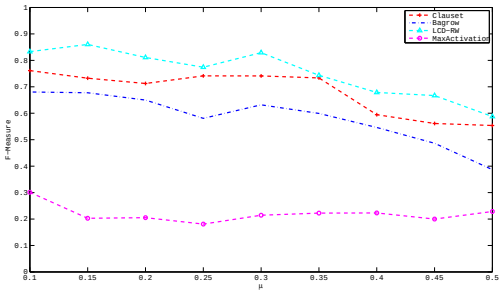
than the Recall obtained by other algorithms while all algorithms have the highest possible Precision as shown in Fig. 11.

B. Conductance

To evaluate the community structure of the found communities, we utilized conductance. As Fig. 13 shows, the average conductance of the communities found by the previous algorithms are worse than communities returned by LCD-RW. The reason is that previous algorithms trap in a local extremum and cannot add all members of the goal community. Therefore, their small set of community nodes are often bounded by a cut which is bigger than the cut of the goal community. While, our algorithm can scape from local extremums and explore the major members of the goal community. Hence, the cut set of the communities



(a)



(b)

FIG. 8: F-measure of algorithms for real world and LFR datasets.

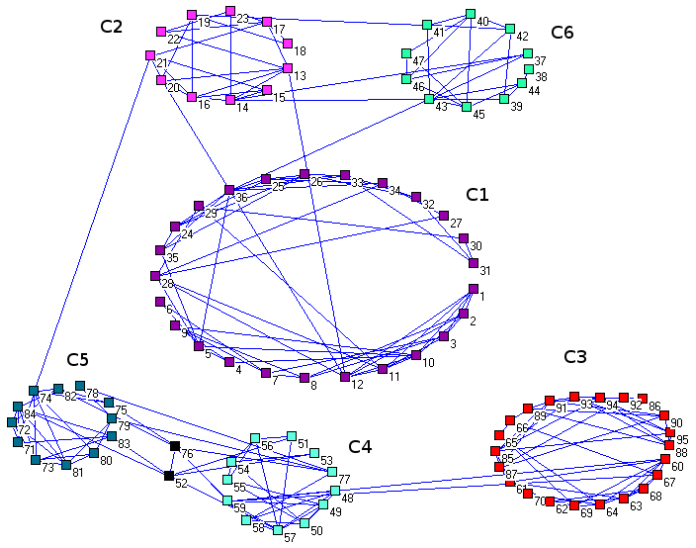


FIG. 9: Protein dataset[31]. This network has 6 communities. Nodes of 76 and 52 belong to both $C5$ and $C4$.

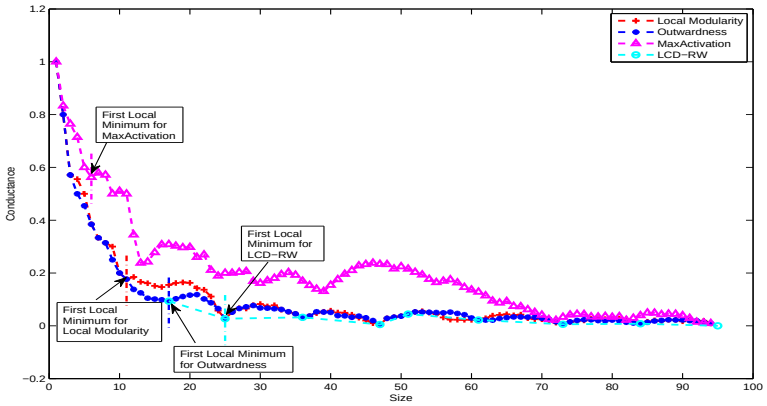


FIG. 10: Conductances changes corresponding to addition of nodes by each node selection criteria. The points that each criterion falls in its first local minimum are marked.

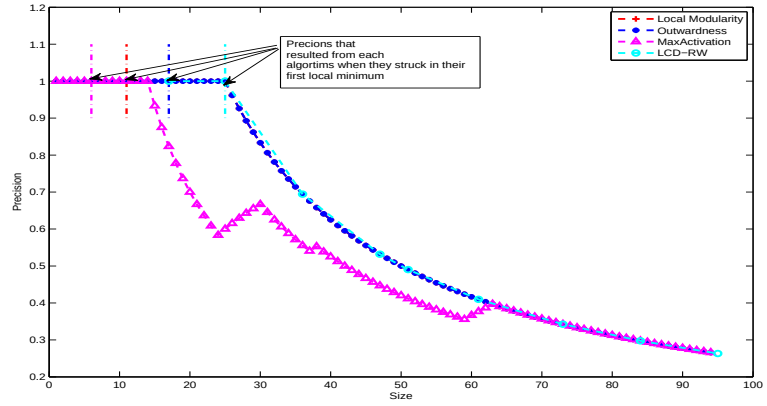


FIG. 11: Precision changes corresponding to addition of nodes by each node selection criteria. The obtained Precisions when each criterion falls in its first local minimum of its conductance are marked.

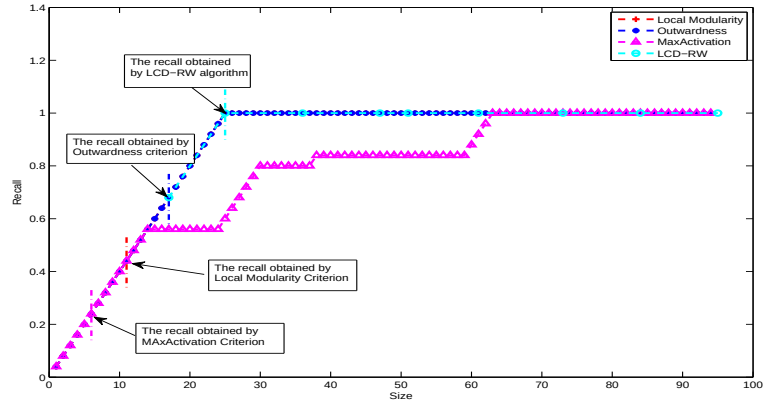
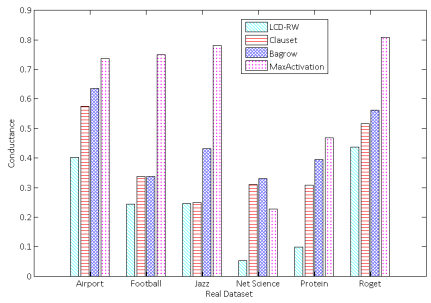
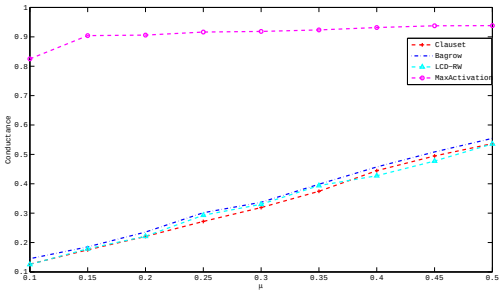


FIG. 12: Recall changes corresponding to addition of nodes by each node selection criteria.The obtained Recall when each criterion falls in its first local minimum of conductanceare marked.



(a)



(b)

FIG. 13: Conductance of algorithms for real world and LFR datasets.

found by the proposed algorithm is smaller, and hence its found communities have a more acceptable structure. The MaxActivation criterion tends towards high degree nodes, and as a result, the number of outdegree edges of communities found by it, is high and it traps in a local minimum(maximum) sooner than other algorithms. In the LFR graphs, similar results are observed. Moreover, increasing μ causes the community structure to become weak and the performance of the algorithms to degrade. Moreover, as Fig. 13b shows, the tendency of MaxActivation towards high degree nodes leads the conductance of its found community to increase more than other algorithms as μ increases.

C. Number of steps

To evaluate the the impact of the number of steps on Precision, Recall and F-measure, we applied our algorithms on the Protein dataset for different number of steps. As shown in Fig.14, increasing this parameter does not considerably impact the Precision. Furthermore, Recall is not significantly influenced by the number of steps. Therefore, the F-measure

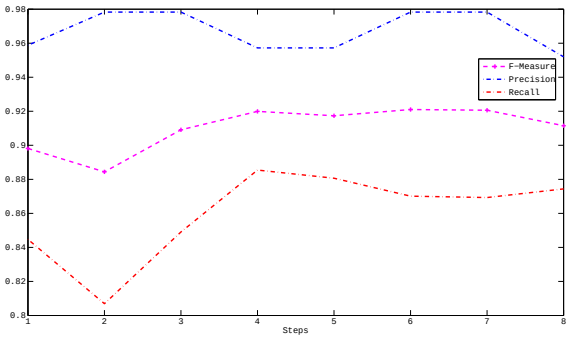


FIG. 14: The impact of the number of steps on Precision and Recall for the Protein dataset.

changes are also negligible. Since, the high values for number of steps forces our algorithm to explore a large part of the graph, we used a small value for the number of steps in our simulations so that the Precision is not considerably decreased.

VII. CONCLUSION

One of the most important characteristics of complex networks is their community structure. In general, links do not appear randomly in these networks, and they have a modular structure where each module is called a community. Many algorithms have been proposed to find communities in such networks. However, their complexity is high and in most large scale networks, the whole network structure is inaccessible. Hence, local community detection algorithms were introduced. One of the most important problems of local algorithms is entrapment in local extremums.

In this paper, we proposed a new local community detection algorithm (LCD-RW) that can escape from local extremums by adding a group of nodes instead of a single node, in the node selection step. The results showed that our algorithm returns a low conductance group of nodes that contains most of the nodes in the community of the initial node. In order to identify the goal community and improve the accuracy of our algorithm, in the filtering step, we applied a global community detection algorithm on the found community which has a few links to rest of networks and can be considered as a small independent graph. We applied the proposed method to real and artificial networks and compared it with well-known and recent local community detection algorithms. Simulations indicate that the proposed algorithm has higher F-measures and efficiently finds the well-structured communities.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Mostafa Salehi of DML for his many useful discussions.

REFERENCES

[1]A. Lancichinetti, F. Radicchi, J. J. Ramasco, and S. Fortunato, "Finding statistically significant communities in networks," PloS one, **6**, e18961 (2011).

[2]S. Fortunato, "Community detection in graphs," Physics Reports E, **486**, 75–174 (2010).

[3]D.Bader and J. McCloskey, "Modularity and graph algorithms," SIAM AN10 Minisymposium on Analyzing Massive Real-World Graphs, 12–16 (2009).

[4]M. E. J. Newman, "Modularity and community structure in networks," Proceedings of the National Academy of Sciences, **103**, 8577–8582 (2006).

[5]A. Clauset, "Finding local community structure in networks," Physical review E, **72**, 026132 (2005).

[6]R. Andersen, F. Chung, and K. Lang, "Local graph partitioning using pagerank vectors," in *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on* (IEEE, 2006) pp. 475–486.

[7]A. Clauset, M. E. J. Newman, and C. Moore, "Finding community structure in very large networks," Physical review E, **70**, 066111 (2004).

[8]J. Duch and A. Arenas, "Community detection in complex networks using extremal optimization," Physical review E, **72**, 027104 (2005).

[9]F. Radicchi, C. Castellano, F. Cecconi, F. Loreto, and D.Parisi, "Defining and identifying communities in networks," Proceedings of the National Academy of Sciences of the United States of America, **101**, 2658–2663 (2004).

[10]R. Andersen and K. L. Lang, "Communities from seed sets," in *Proceedings of the 15th international conference on World Wide Web* (ACM) pp. 223–232.

[11]J. P. Bagrow, "Evaluating local community methods in networks," Journal of Statistical Mechanics: Theory and Experiment, **2008**, P05001 (2008).

[12]J. Chen, O. Zaïane, and R. Goebel, "Local community identification in social networks," in *Social Network Analysis and Mining, 2009. ASONAM'09. International Conference on*

Advances in IEEE (IEEE, 2009) pp. 237–242.

[13]F. Luo, J. Z. Wang, and E. Promislow, "Exploring local community structures in large networks," Web Intelligence and Agent Systems, **6**, 387–400 (2008).

[14]J. P. Bagrow and E. M.Bollt, "Local method for detecting communities," Physical Review E, **72**, 046108 (2005).

[15]J. Tian, D. Chen, and Y. Fu, "A new local algorithm for detecting communities in networks," in *Education Technology and Computer Science, 2009. ETCS'09. First International Workshop on*, Vol. 2 (IEEE, 2009) pp. 721–724.

[16]L. K. Branting, "Context-sensitive detection of local community structure," Social Network Analysis and Mining, **2**, 279–289 (2012).

[17]L. K. Branting, "Incremental detection of local community structure," in *Advances in Social Networks Analysis and Mining (ASONAM), 2010 International Conference on* (IEEE, 2010) pp. 80–87.

[18]R. Andersen, F. Chung, and K. Lang, "Using pagerank to locally partition a graph," Internet Mathematics, **4**, 35–64 (2007).

[19]J. Leskovec, K. J. Lang, A. Dasgupta, and M. M. Mahoney, "Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters," Internet Mathematics, **6**, 29–123 (2009).

[20]M. Rosvall and C. T. Bergstrom, "Maps of random walks on complex networks reveal community structure," National Academy of Sciences, **105**, 1118–1123 (2008).

[21]R. Dunbar and R. I. M. Dunbar, *Grooming, Gossip and The Evolution of Language* (Harvard University Press, 1998).

[22]C. Allen, "The dunbar number as a limit to group sizes," Publicado em, **10** (2004).

[23]J. Leskovec, K. J. Lang, and M. Mahoney, "Empirical comparison of algorithms for network community detection," in *Proceedings of the 19th international conference on World wide web* (ACM) pp. 631–640.

[24]R. Milo, S. Itzkovitz, N. Kashtan, R. Levitt, S. Shen-Orr, I. Ayzenshtat, M. Sheffer, and U. Alon, "Superfamilies of evolved and designed networks," Science, **303**, 1538–1542 (2004).

[25]M. Girvan and M. E. N. Newman, "Community structure in social and biological networks," Proceedings of the National Academy of Sciences, **99**, 7821–7826 (2002).

[26]P. M. Gleiser and L. Danon, "Community structure in jazz," Advances in complex systems, **6**, 565–573 (2003).

[27]M. E. J. Newman, "The structure of scientific collaboration networks," Proceedings of the National Academy of Sciences, **98**, 404–409 (2001).

[28]V. Batagelj and A. Mrvar, *Pajek-Analysis and Visualization of large networks* (Springer, http://pajek.imfm.si/doku.php?id=data:index, 2004).

[29]A.Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," Physical Review E, **48**, 046110 (2008).

[30]A. Lancichinetti and S. Fortunato, "Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities," Physical Review E, **80**, 016118 (2009).

[31]M. Salehi, H. R. Rabiee, and A. Rajabi, "Sampling from complex networks with high community structures," Chaos, **22**, 023126–023126 (2012).

[32]W. W. Zachary, "An information flow model for conflict and fission in small groups," Anthropological Research, 452–473 (1977).