AntLP: ant-based label propagation algorithm for community detection in social networks

Razieh Hosseini\textsuperscript{1}, Alireza Rezvanian\textsuperscript{2,3}

\textsuperscript{1}Department of Computer Engineering, Alzahra University, Tehran, Iran  
\textsuperscript{2}Department of Computer Engineering, University of Science and Culture, Tehran, Iran  
\textsuperscript{3}School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran  
\textsuperscript{✉}E-mail: rezvanian@usc.ac.ir

Abstract: In social network analysis, community detection is one of the significant tasks to study the structure and characteristics of the networks. In recent years, several intelligent and meta-heuristic algorithms have been presented for community detection in complex social networks, among them label propagation algorithm (LPA) is one of the fastest algorithms for discovering community structures. However, due to the randomness of the LPA, its performance is not suitable for the general purpose of network analysis. In this study, the authors propose an improved version of the label propagation (called AntLP) algorithm using similarity indices and ant colony optimisation (ACO). The AntLP consists of two steps: in the first step, the algorithm assigns weights for edges of the input network using several similarity indices, and in the second step, the AntLP using ACO tries to propagate labels and optimise modularity measure by grouping similar vertices in each community based on the local similarities among the vertices of the network. In order to study the performance of the AntLP, several experiments are conducted on some well-known social network datasets. Experimental simulations demonstrated that the AntLP is better than some community detection algorithms for social networks in terms of modularity, normalised mutual information and running time.

1 Introduction

Many of real networks such as information, technological and social networks, are represented as network systems which have several common characteristics, such as a small-world phenomenon [1], scale-free network property [2] and in particular having modular structures feature (also known a cluster or community structures) [3]. In recent years, some studies have been made to show that community structures are a significant characteristic in the most complex networks such as social and information networks due to the strong tendencies among human beings to form common interest groups or social communities with those users having similar activities or common interests [4, 5]. In a network, a community structure is the set of vertices with more internal links inside the community than external links between them. Finding the community structures of complex social networks plays an important role in social network analysis because it can help researchers to understand the organisation, structure, characteristics, dynamics, and evolution of the network. Discovering and analysing communities have been applied in many domains and network applications [6–9] such as epidemiology networks, biological networks, metabolic networks, ecological networks, and especially online social networks [7, 10].

In general, a social network can be modelled as a graph \( G = (V, E) \), where \( V \) is a set of vertices representing the social users, and \( E \) is a set of edges representing a kind of interaction between the users. A community in a network is defined as a group of vertices that have more edges among themselves than vertices outside the group.

In this paper, we propose a hybrid approach based on a label propagation algorithm and ant colony optimisation (ACO) algorithm, in which the edges of the graph are weighted by similarity indices. Then, the proposed algorithm using ACO tries to propagate labels and detect communities from networks by locally optimising modularity measure using individual cooperative ants. The proposed algorithm, called AntLP (ant-based label propagation) for community detection in social networks.

The rest of this paper is organised as follows. We outline the related work in Section 2. The preliminary material requirements, before describing the proposed algorithm, are presented in Section 3. Section 4 describes the standard ant-based algorithm and the proposed community detection algorithm as ant-based label propagation (AntLP) algorithm. Experimental simulations are presented in Section 5, and Section 6 concludes the paper.

2 Related work

Due to the significant task of community detection applications in social network analysis, several community detection approaches have been presented in the literature, and these have been classified in different ways [11–25]. According to a well-known classification mentioned in [11], these methods can be categorised into six groups: spectral, and clustering methods, divisive algorithms, modularity-based methods, model-based methods, local community detection methods, and feature-based assisted methods. Among these methods, modularity-based methods by optimising a quality function for a proper graph partitioning or maximising the modularity function have been popularised for community discovery in social network analysis. In [12], the researchers proposed a polynomial-time approximation algorithm for the modularity maximisation problem in the context of scale-free networks. Aztmouller et al. [15] proposed several optimistic estimates of standard community quality functions in order to efficient pruning of the search space in an exhaustive branch-and-bound algorithm. They evaluated their approach to different social media applications. Rechian et al. [16] proposed a multi-objective non-linear discrete optimisation using biogeography-based optimisation (BBO) algorithm with vertex attributes in order to reach to a trade-off between similarity of vertices’ attributes, and density of connections for discovering communities in social networks. Also, they introduced new similarity indices for vertex attributes in a community of a network.

Some of the studies are focused on finding the optimal community detection by optimisation [21–24], such as the density drop of
sub-graphs method [14]. Intuitively, this algorithm is based on the drops in density between each pair of parent and child vertices on the dendrogram, where the higher the drop in the density, the larger the corresponding child sub-graph is. Moreover, these authors proposed a novel algorithm which can output an optimal set of local communities automatically, based on the max-flow min-cut concept, which is suitable when the dendrogram is a tree. A novel measure proposed by Yang et al. [13], integrates both the concept of closed walks and the concept of clustering coefficients, instead of edge betweenness, in the well-known divisive hierarchical clustering algorithm. The edges with the lowest values are removed iteratively until the network has degenerated into isolated vertices.

Label propagation algorithms (LPAs) similar to a family of local community detection methods, propagate same labels to neighbouring vertices. Since the task of label propagation proceeds locally, a label propagation algorithm is a fast algorithm. Hosseini and Azmi [17] proposed an improved label propagation algorithm called memory-based label propagation algorithm (MLPA) for finding community structure in social networks. In their algorithm, a simple memory element is designed for each vertex of the graph and, this element store the most frequent common adoption of labels iteratively. In [18], a community detection method based on fuzzy logic is proposed by Zhang et al. that iteratively propagates membership-degrees of all vertices. In this algorithm, in an iteration, a candidate seed vertex of a potential community is first selected according to the topological characteristics. After that, the membership-degrees are propagated among adjacent vertices so that a number of communities can be obtained with respect to all selected seeds. In their algorithm discarded the chosen seeds that decrease the modularity of the community decomposition in order to improve modularity in an iteration.

Elyasi et al. [19] presented a two-phase community detection algorithm using the Louvain method, and applying a belonging matrix. In this algorithm belonging matrix determines how much a vertex belongs to a community, and finally, some of them found communities are merged based on the modularity measure. In [20], Staudt et al. presented a flexible, and extensible community detection framework with shared-memory parallelism by addressing the deficit in computing capability. Based on this framework, they designed, and implemented two efficient parallel community detection heuristics. The first is a parallel label propagation scheme, and the second is the parallel version of the Louvain method. Their methods are suitable for massive datasets.

Khoshaman et al. [21] investigated a distributed learning automata-based algorithm for detecting communities in deterministic graphs. According to this algorithm, a set of learning automata interact with each other in order to identify high-density local communities by updating the action probability vector of a network of cooperative learning automata. The improved version of this algorithm for community detection has also been developed by Ghamgosar et al. [27] using extended distributed learning automata. They also presented using a cellular learning automata-based algorithm for community detection and called their algorithm as CLACD. Since the size of a social network is commonly massive, their algorithm is first formed local communities on a partial spanning tree of the network in order to reduce the network size. As the CLACD proceeds, each LA in CLA interacts with both local, and global environments to modify the found communities that gradually yielded the near-optimal community structure of the network through the evolution of the CLA. The authors conducted several experiments to show the performance of the CLACD algorithm and compared it with some well-known community detection algorithms [28]. A comprehensive study on the learning automata approaches for community detection in complex social networks is reported in [7].

Although, most of the existing community detection algorithms focus on binary networks, and most of the networks are naturally weighted, for example delay-tolerant networks (DTNs) or online social networks (OSNs). In [29], an improved version of the betweenness-based algorithm was presented in weighted networks and was applied to both weighted, and unweighted networks. According to this algorithm, the edge betweenness, dissimilarity index, and edge-clustering coefficient are converted into the weight of the edges, leading to the conclusion that the inter-community edges have higher discriminating values (or weighted betweenness) than edges in the community edges, and hence that the inter-community edges can be more easily identified than before, and the strengths of relations will be clear. In [30], the authors illustrated the problems of community detection in weighted networks and used weighted communities for data forwarding in DTN and worm containment in OSN. They also introduced two metrics; intra-centrality, and inter-centrality, to characterise vertices in communities.

Besides, several studies have been undertaken on ant-based algorithms for community detection in social networks. Lucio et al. in [31], proposed a multi-layer ant-based algorithm, which detects communities in networks by locally optimising the modularity using individual ants. Due to the intrinsic multi-level nature of this algorithm, the method potentially can display multi-scale hierarchical structures. Moreover, the method can mitigate the resolution limit of modularity through modularity optimisation. Jin et al. in [32], proposed a new ant colony optimisation strategy based on Markov random walk theory. This method is based on the progressive strengthening of internal links (within-community links), and the weakening of external links (between-community links), which results in the underlying community structure of the network becoming visible. Sadi et al. [33] presented a method for finding cliques (or quasi-cliques) in a network, based on ACO techniques, assigning these cliques (or quasi-cliques) as vertices in a reduced graph, and subsequently using conventional community detection algorithms on the reduced graph. A multi-objective community detection algorithm named MOCD-ACO is presented in [34] by Ji et al. using the heuristic operator of ACO, and the multi-objective evolutionary algorithm based on decomposition.

Many algorithms have been presented to detect overlapping communities. In [35], the authors present an ant colony-based overlapping community detection algorithm, which includes ants’ location initialisation, ants’ movements, and post-processing phases. An ants’ location initialisation strategy is designed to identify the initial locations of the ants, and each vertex keeps the initialised label list. During the ants’ movement phase, ants move according to the transition probability matrix and use a redefined similarity index to compute new heuristic information. A novel algorithm to detect overlapping community structures in bipartite networks is proposed in [36]. The authors introduce key bi-communities, and free vertices, and some key bi-communities, and free vertices are extracted from the original bipartite networks. Then, the free vertices are allocated to a certain key bi-community using given rules. Also, this algorithm finds overlapping vertices between communities. The authors also proposed an algorithm to identify the overlapping community structure in [37], where the algorithm firstly detects fully connected sub-graphs, which are maximal sub-graphs from original networks. Then, two maximal sub-graphs having key pair-vertices can be merged into a new larger sub-graph using some defined degree functions. However, all the above ant-based algorithms use a heuristic method without explicit optimisation objectives and are computationally expensive.

3 Material and methods

A social network is typically represented as an undirected graph $G = (V, E)$, where $V = \{v_1, v_2, \ldots, v_n\}$ denotes the set of vertices and $E = \{e_1, e_2, \ldots, e_m\}$ denotes the set of edges with $m = |E|$ vertices, and $E_{ab}$ denotes the community detection algorithm [28]. A comprehensive study on the learning automata approaches for community detection in complex social networks is reported in [7]. However, most of the existing community detection algorithms focus on binary networks, and most of the networks are naturally weighted, for example delay-tolerant networks (DTNs) or online social networks (OSNs). In [29], an improved version of the betweenness-based algorithm was presented in weighted networks and was applied to both weighted, and unweighted networks. According to this algorithm, the edge betweenness, dissimilarity index, and edge-clustering coefficient are converted into the weight of the edges, leading to the conclusion that the inter-community edges have higher discriminating values (or weighted betweenness) than edges in the community edges, and hence that the inter-community edges can be more easily identified than before, and the strengths of relations will be clear. In [30], the authors illustrated the problems of community detection in weighted networks and used weighted communities for data forwarding in DTN and worm containment in OSN. They also introduced two metrics; intra-centrality, and inter-centrality, to characterise vertices in communities.
3.1 Basic definitions

In this subsection, we introduce some fundamental definitions required in the paper before describing the proposed algorithm.

Definition 1: (Vertex degree): In a given undirected graph $G = (V, E)$, the vertex degree is the number of edges incident to the vertex, i.e. the sum of the number of all edges of vertex $v_i$ is denoted by $\text{Deg}(v_i)$.

Definition 2: (Unvisited vertex): Given a partitioning $C = \{c_1, \ldots, c_s\}$ for an undirected graph $G = (V, E)$, a vertex $v_i \in c_i$ is called unvisited if $c_i$ contains only vertex $v_i$. In other words, an unvisited vertex does not yet participate in a community, and therefore it has no label.

Definition 3: (Vertex importance): Given an undirected graph $G = (V, E)$, we define the vertex importance of $v_i$ as follows:

$$\text{Importance}(v_i) = \frac{\text{Deg}(v_i)}{\max_{v_j}(\text{Deg}(v_j))}$$  \hspace{1cm} (1)

From Definition 1, it is clear that the importance of vertices determines the importance of the vertex $v_i$ in the community $c_i$, it can be stated that the higher the degree of the vertex, the higher is its value of importance.

Definition 4: (Community importance): Given an undirected partitioning $C = \{c_1, \ldots, c_s\}$ of $G = (V, E)$ we define the importance of a community $c_i$ in $C$ as the average importance of its vertices

$$\text{Importance}(c_i) = \frac{\sum_{v_i \in c_i} \text{Importance}(v_i)}{|c_i|}$$  \hspace{1cm} (2)

where $|c_i|$ denotes the cardinality of a community $c_i$ that is the number of vertices in the community $c_i$ [38].

Definition 5: (Purity of a community): Given a partitioning $C = \{c_1, \ldots, c_s\}$ of $G = (V, E)$ and a community $c_i \in C$, the purity of $c_i$ can be defined as the average of the importance of its pure vertices, as follows:

$$\text{Purity}(c_i) = \frac{\sum_{v_i \in N(v_i)} \text{Importance}(v_i)}{\sum_{v_i \in E} \text{Deg}(v_i)}$$  \hspace{1cm} (3)

where $N(v_i)$ is the set of neighbouring vertices of the vertex $v_i$. In the numerator of the above equation, only vertices of community $c_i$ are considered for which all neighbouring vertices are in the community $c_i$ [38]. It is worth noting when measuring purity, each unvisited vertex is an impure vertex, and for each community $c_i \in C$, we can state $0 \leq \text{Purity} \leq 1$.

3.2 Standard LPA

LPA was proposed by Raghavan et al. [39] as a localised community detection algorithm. In LPA, the algorithm initially assigns a unique label for each vertex, and in every iteration of the algorithm, each vertex adopts a label with the most repetition in the neighbouring vertices. As the label adoption proceeds, a consensus label propagates through the network inside the densely connected group of vertices. At the end of the algorithm, vertices having the same labels are grouped as communities. It is worth to note that in the LPA, the number of communities and their sizes are not known a priori. The standard label propagation algorithm is described as the following steps:

1. Initialise the labels at all vertices in the network. For a given vertex $v_i$, its label is $IL$. Label $IL$ should be a unique label.

2. Arrange the vertices in the network in a random sequence $S$. For a given vertex $v_i$, its order is $iS$.

3. Each vertex changes its label to the maximum number of the same label among its neighbours in the order of sequence $S$.

4. Iterate steps (2) and (3) above until no labels can be changed.

3.3 Similarity indices

Similarity indices in the network essentially measure the similarity of the structural context in the graph, where, two vertices are considered to be similar if they have many common features [40]. Since the attributes of vertices are generally hidden, we use another type of similarity named structural similarity that is based on the network structure and does not need global topological information, thereby needing less computational time. These indices are typically based on the number of common neighbours and mainly discussed in link prediction studies in online social networks [40]. Here, we introduce seven local similarity indices, including common neighbours (CN), Sorensen, Hub promoted index (HPI), Hub depressed index (HDI), and Leicht–Holme–Newman (LHN) index as the following description:

(i) CN index [40]. CN is one of the simplest measures for computing similarity between vertex $v_i$ and vertex $v_j$ by counting directed overlapping neighbours.

$$S_{ij}^{CN} = \left| \Gamma(v_i) \cap \Gamma(v_j) \right|$$  \hspace{1cm} (4)

where let $\Gamma(v_i)$ is the set of neighboring vertices of the vertex $v_i$. Intuitively, two similar vertices have many common neighbours.

(ii) Sorensen index [40]. This index also called the cosine similarity, which normalises the result of CN index by considering the degrees of vertices

$$S_{ij}^{\text{Sorensen}} = \frac{\left| \Gamma(v_i) \cap \Gamma(v_j) \right|}{\sqrt{k_{v_i} \times k_{v_j}}}$$  \hspace{1cm} (5)

where $k_{v_i}$ and $k_{v_j}$ are the degrees of vertex $v_i$ and vertex $v_j$, respectively.

(iii) Jaccard index [40]. This index is known as a popular similarity concept for many domains and it defines as

$$S_{ij}^{\text{Jaccard}} = \frac{\left| \Gamma(v_i) \cap \Gamma(v_j) \right|}{\left| \Gamma(v_i) \cup \Gamma(v_j) \right|}$$  \hspace{1cm} (6)

This index is similar to the common neighbour index; the difference is that in the Jaccard index the vertices are more similar if they have a large number of common neighbours and a small number of non-common neighbours.

(iv) Hub promoted index (HPI) [40]. Based on this index, the edges adjacent to high-degree vertices are likely to be assigned high scores, since the denominator is determined by the lower degree only. HPI is defined as follows:

$$S_{ij}^{\text{HPI}} = \frac{\left| \Gamma(v_i) \cap \Gamma(v_j) \right|}{\min(k_{v_i}^-, k_{v_j}^-)}$$  \hspace{1cm} (7)

(v) Hub depressed index (HDI) [40]. Similar to the HPI, another modification can also be considered for high-degree vertices.
as follows:

\[ s_{xy}^{PH} = \frac{\Gamma(v_x) \cap \Gamma(v_y)}{\max\{k_{v_x}, k_{v_y}\}} \]  

(vii) LHN index [40]. The LHN index, similar to all introduced similarity index, considers the number of common neighbours for the numerator and normalises the result by dividing on multiplying degrees of vertices as follows:

\[ s_{xy}^{LHN} = \frac{\Gamma(v_x) \cap \Gamma(v_y)}{k_{v_x} \times k_{v_y}} \]  

3.4 Standard ACO algorithm

ACO is inspired by the foraging behaviour of ant colonies and targets discrete optimisation problems. According to similar studies, each ant is considered to be a vertex of a complex network. At the beginning of the ACO, ants are randomly placed in different locations of the network [41]. During the evolution process, each ant uses a fitness function in order to perceive the local environment and to decide whether to move to a new location or stay in the original location. In an iteration, if there is a better position for the ant, then it moves to the new location; otherwise, it maintains a sleeping state. Finally, ant colony movements result in a clustering of corresponding vertices. The transition probability of each ant for moving from vertex \( v_i \) to vertex \( v_j \) by the \( k \)th ant is computed as follows:

\[ p_{ij}^{k} = \frac{\tau(i,j)^{\alpha} \cdot \eta(i,j)^{\beta}}{\sum_{j \in N(i)} \tau(i,j)^{\alpha} \cdot \eta(i,j)^{\beta}} \]  

where \( \tau(i,j) \) and \( \eta(i,j) \) are the amount of pheromone and the desirability on edge between vertex \( v_i \) and vertex \( v_j \), respectively. \( N(i) \) is the neighbouring vertices adjacent to vertex \( v_j \), also \( \alpha \geq 0 \) and \( \beta \geq 1 \) are the parameter controls. Note that potentially there are many different ways of choosing the transition probabilities. The above equation has the most common form because it was used in the first ACO algorithms.

3.4.1 Pheromone updating: This algorithm updates the pheromone values using previously generated solutions. Allowing the pheromone, update to be a function of the generated solution quality can help in directing future ants more strongly toward better solutions. In fact, by allowing ants to deposit a higher amount of pheromone on the shortest paths, the ants’ path-searching is more quickly biased toward the best solutions. Generally, the number of vertices moving around the environment determines the amount of pheromone, which becomes greater on the shorter paths which have more vertices. As mentioned above, the ants update pheromone trails just after a solution is generated. The algorithm updates the pheromone intensity for all vertices as follows:

\[ \tau(i,j) = \tau(i,j) + \Delta \tau(i,j)^k \]  

where \( \Delta \tau(i,j)^k \) is the amount of pheromone deposited by the \( k \)th ant.

3.4.2 Pheromone evaporation: In this step, pheromone evaporation is done as the following equation:

\[ \tau(i,j) = (1 - \rho) \tau(i,j), \quad \forall_{v_i,v_j} \in A \]  

where \( \tau(i,j) \) is the pheromone trail deposited by ants between vertices \( v_i \) and \( v_j \), and \( \rho \) is a parameter in the range \( 0 < \rho \leq 1 \). If \( (\rho = 0) \), then no pheromone evaporation takes place.

4 Proposed algorithm: AntLP

The proposed algorithm for community detection in social networks is called the ant-based label propagation (AntLP) algorithm, which is consists of four steps: in the first step, the AntLP algorithm initialises the labels and parameters, assigns a weight for each edge of the input network using a given similarity indices and computes the transition probability; in the second step, according to the ant traversal, label of each vertex is changed, and the candidate communities are formed based on the vertices with same labels; in the third step, the quality of candidate communities is computed based on the modularity function; and finally in the last step, the stopping conditions are checked. The AntLP, which is shown in Fig. 1 as Algorithm 1 is described as the following subsections.

4.1 Initialisation

In the initialisation, each vertex of the graph is first initialised as an unvisited vertex based on Definition 2 and labelled randomly, the initial community \( C \) is considered as an empty set, the importance degree of all vertices in the network are computed based on (1) and for ACO, the initial transition probability \( P(i, j) \) from vertex \( v_i \) to vertex \( v_j \) is computed according to the following equation:

\[ P(i, j) = \frac{\tau(i,j)^{\alpha} \cdot \text{sim}(i,j)^{\beta}}{\sum_{\forall_{j \in N(i)} \tau(i,j)^{\alpha} \cdot \text{sim}(i,j)^{\beta}}} \]  

where \( \text{sim}(i,j) \) is the value of a given similarity index between vertex \( v_i \) and vertex \( v_j \) (i.e. using Jaccard index as defined in (6)), \( N(i) \) is the set of neighbouring vertices for vertex \( v_i \) and \( \alpha = 2 \) and \( \beta = 1 \) are parameter controls. In the proposed algorithm, as the desirability of the edge heuristically is similar to pair vertices and is taken as heuristic information for ants to communicate with each other. It is worth to note that in the proposed algorithm, one can use any similarity index for computing similarity between two arbitrary vertices.

4.2 Composing communities

After initialisation, a starting vertex is chosen, which has the highest value of importance, and the first ant is placed at this vertex. The corresponding ant traverses the edges of network-based the transition probability defined as (14) and adds some vertices in the candidate community in such a way that each ant decides to change the label of the current vertex to the candidate community.
by taking similar label for these vertices, if the changing label of the current vertex causes an increase in the objective function $F(c_j)$ as follows:

$$F(c_j) = \text{Importance}(v_j) \times \text{Purity}(v_j, c_j)$$

where Importance($v_j$) is the importance value of vertex $v_j$ and Purity($v_j$, $c_j$) is the purity value of vertex $v_j$ in community $c_j$, computed by (3). Thus, the label of the current vertex will be changing as a new member of the candidate community and tagged as a visited vertex if the current vertex improves the value of the given objective function $F(c_j)$; otherwise, it will be ignored and will be remained as an unvisited vertex. The process of traversing ants on the edges of the network, changing and propagating label of each vertex is continued until the set of unvisited vertices is empty. As this step proceeds, each vertex receives a label and several vertices with the same labels form a community. These communities are iteratively made based on the ant traversal through the path of similar vertices, changing and propagating labels using importance between the unvisited vertices as defined in (2).

### 4.3 Evaluating communities

The step of visiting unvisited vertices by ants through the path of similar vertices for composing the set of communities based on their importance for each community is repeated until all vertices are labelled. Subsequently, the pheromones of paths visited by ants are updated in such a way that the set of communities found by the ants is evaluated using the modularity measure as follows:

$$Q(C) = \frac{1}{2m} \sum_{ij} A_{ij} \left( \delta(c_i, c_j) - \frac{k_i k_j}{2m} \right)$$

where $A$ indicates the adjacency matrix of the input network in which $A_{ij}$ is one when a vertex $v_i$ is connected to vertex $v_j$ and zero otherwise. $m$ indicates the total number of edges of the input network, $k_i$ denotes the degree of vertex $v_i$ and $\delta(c_i, c_j)$ is the delta function yields one if vertex $v_i$ and vertex $v_j$ are in the same community and zero otherwise. Therefore, if the new modularity value is better than the average of all previously obtained modularity values, the pheromones are increased; otherwise, they are decreased.

### 4.4 Stopping conditions

The process of computing transition probability, visiting vertices by ants, propagating labels, composing a set of candidate communities, evaluating the obtained set of communities and updating pheromones, is repeated until no more improvement is possible or the iteration number of the algorithm exceeds a predefined number of iterations.

### 5 Simulation results

In order to analyse the performance of the AntLP algorithm, first, some well-known social network datasets are introduced in Table 1. Then, the results of the experiments are given.

#### Table 1 Results for different similarity indices with respect to modularity measure

| Networks    | CN       | Salton   | Jaccard  | Sorenson | HPI      | LHN      | HDI      |
|-------------|----------|----------|----------|----------|----------|----------|----------|
| karate      | 0.410 ± 0.013 | 0.411 ± 0.010 | 0.412 ± 0.011 | 0.411 ± 0.012 | 0.414 ± 0.011 | 0.416 ± 0.005 | 0.415 ± 0.011 |
| dolphins    | 0.510 ± 0.011 | 0.511 ± 0.007 | 0.512 ± 0.008 | 0.512 ± 0.008 | 0.515 ± 0.008 | 0.519 ± 0.001 | 0.518 ± 0.007 |
| books       | 0.512 ± 0.001 | 0.521 ± 0.001 | 0.520 ± 0.008 | 0.514 ± 0.010 | 0.523 ± 0.010 | 0.524 ± 0.007 | 0.523 ± 0.009 |
| football    | 0.566 ± 0.009 | 0.570 ± 0.004 | 0.584 ± 0.001 | 0.567 ± 0.004 | 0.595 ± 0.002 | 0.598 ± 0.003 | 0.596 ± 0.006 |
| net-science | 0.940 ± 0.014 | 0.944 ± 0.013 | 0.945 ± 0.017 | 0.944 ± 0.016 | 0.946 ± 0.017 | 0.948 ± 0.012 | 0.948 ± 0.014 |
| power-grid  | 0.831 ± 0.016 | 0.849 ± 0.015 | 0.843 ± 0.016 | 0.832 ± 0.013 | 0.853 ± 0.012 | 0.857 ± 0.010 | 0.856 ± 0.009 |

#### 5.1 Experiments

In the following subsections, we described the report of designed experiments. The reported results are taken the average over 30 independent runs for 5000 iterations and the parameter control $\alpha$ for (14) is set to 0.1.

#### 5.1.1 Comparing the effect of similarity indices: In this experiment, the performance of the AntLP is investigated using different similarity indices including CN as defined in (4), Salton as defined in (5), Jaccard as defined in (6), Sorenson as defined in (7), HPI as defined in (8), HDI as defined in (9) and LHN as defined in (10) with respect to modularity and running time. The algorithm is performed over 5000 times, and the average and standard deviation of the results are reported. The results of modularity and running time for different similarity indices are given in Tables 2 and 3, respectively.

From the results of Table 2, one can see that AntLP using HHN is outperforms the other similarity indices. Also, based on the results of Table 3, among the similarity indices, CN yields the lower running time compared to other similarity indices. According to the results, in the rest of the experiments, we use LHN as similarity in AntLP due to having the superior modularity and the lower running time in comparison with other similarity indices.

#### 5.1.2 Comparing AntLP with other algorithms: In this experiment, the performance of the AntLP algorithm for finding the communities in term of the modularity $Q$ is compared with other community detection algorithms including LPA [39], MLPA [17], CNM [42], GANET [43], CLACD [28], MLAMA [45] and CLACD [28]. The results presented in Table 4, including both the maximum (Max) and average (Avg.) values of modularity. From Table 4, one can see that AntLP for Football, Net-science and Power-grid networks, have large values for modularity in comparison with CNM and GANET. For Books network, AntLP algorithm yields similar results to GANET, MLAMA, and CLA.

#### 5.1.3 Test of significance: In order to evaluate the significance of the differences in the results of the AntLP algorithm in comparison with other community detection algorithms statistically, we used two-tailed $t$-test with 28 degrees of freedom at a 0.05 level of significance. The t-score to check if the two algorithm means are different can be calculated by the following equation:
Table 3 Running time of the proposed algorithm with different similarities (seconds)

| Networks       | Similarity indices |
|---------------|-------------------|
|               | CN     | Salton | Jaccard | Sorenson | HPI   | HDI   | LHN   |
| karate        | 3.21   | 4.05   | 4.72    | 4.397   | 4.02  | 4.65  | 4.96  |
| dolphins      | 11.24  | 12.81  | 14.27   | 13.32   | 12.28 | 13.72 | 13.52 |
| books         | 30.67  | 42.06  | 41.60   | 48.45   | 45.92 | 45.21 | 45.84 |
| football      | 65.98  | 91.81  | 50.55   | 93.66   | 94.45 | 96.75 | 95.27 |
| net-science   | 694.05 | 784.17 | 716.14  | 745.09  | 765.31| 766.86| 767.22|
| power-grid    | 1823.35| 2252.51| 2168.84 | 2235.26 | 2294.88| 2299.51| 2301.17|

Table 4 Comparison of the community detection algorithms on real networks in terms of maximum (Max) and average (Avg.) modularity

Table 5 Comparison of the community detection algorithms on real networks in terms of maximum (Max) and average (Avg.) modularity

where $m_i$, $var_i$, $n_i$ are the means, variances, and the number of runs for algorithm $i$, respectively. The results of $t$-scores are presented in Table 5, where $P$ denotes the performance of AntLP algorithm versus other community detection algorithms (LPA [39], MLPA [17], CNM [42], GANET [43], CLA [44], MLAMA [45] and CLACD [28]). The results of LPA, MLPA, CLA, and CLACD are extracted from our simulations of these algorithms and the results of CNM, GANET, and MLAMA directly are adopted from reports of their papers.

In Table 5, the performance of AntLP versus other community detection algorithms is shown as ‘−’; ‘+’ and ‘~’ when AntLP is significantly better than, better than, or similar to the compared algorithm, respectively. Based on the test results, one may conclude that AntLP is significantly better than standard LPA for all networks. For Karate, AntLP significantly outperforms MLPA, CNM, and CNM. For five out of six networks, AntLP is similar to GANET, MLAMA, CLA, and CLACD.

5.1.4 Study on artificial networks: This experiment is carried out to study the performance of the AntLP algorithm on a well-known modular network, with respect to the normalised mutual information (NMI) measure. For this purpose, we examine the AntLP algorithm on the well-known computer-generated modular network introduced in [46]. The LFR benchmark network is an artificial network for community detection that has some of the statistical properties found in real networks, such as degree distributions and community size. Many parameters are used to determine the properties of generated networks in this benchmark network, such as $N$ (number of vertices), $k$ (average degree), $k_{\text{min}}$ (minimum degree), $k_{\text{max}}$ (maximum degree), $c_{\text{min}}$ and $c_{\text{max}}$ (minimum and maximum community sizes), $t_1$ (exponent of power-law distribution of vertices degrees), $t_2$ (exponent of power-law distribution of community sizes) and $\mu$ (mixing parameter).

In order to evaluate the performance of the proposed algorithm for finding community structures, the NMI measure is computed, to quantitatively compare a known partition with the partition found by the AntLP algorithm:

\[
\text{NMI}(A, B) = -\frac{1}{2} \sum_{i,j} N_i N_j \log((N_i N_j)/(N \cdot j/N)) + \sum_i N_i \log(N_i/N) + \sum_j (N \cdot j/N) \log((N \cdot j/N))
\]

where $c_j$ denotes the real number of communities and $c_B$ denotes the number of found communities. The matrix $N$ represents the confusion matrix, where $N_{ij}$ is simply the number of vertices in the real community $i$ that appear in the found community $j$. $N_i$ and $N_j$ are the sums over row $i$ and column $j$ of the confusion matrix, respectively. $N$ is clearly the number of vertices. If the found partition is the same as the real communities, then the NMI takes a value of one, otherwise, NMI is zero due to the fact that the found partition is completely independent of the real partition. In this experiment, the ability of the algorithm for different values of the
mixing parameter $\mu$ in LFR benchmark networks is studied. For this purpose, the artificial modular network based on LFR benchmark is generated using the following parameters: $N=1000$, $k=10$, $\mu=(0.1, 0.2, \ldots, 0.8)$, $t_1=2$, $t_2=1$, $c_{\text{min}}=10$, and $c_{\text{max}}=50$.

The results of the AntLP algorithm by comparing several other algorithms (CNM, GANet, MLAMA, CLA-Net, and CLACD) for different values of the mixing parameter $\mu$ are presented in Fig. 2 with respect to NMI. As shown in Fig. 2, for a mixing parameter value greater than 0.3 and lower than 0.6, the obtained NMI results are distinguishable, so AntLP is on the top of GA-Net, DLACD, CNM, and MLAMA in term of NMI.

6 Conclusions

In this paper, we discussed that due to the randomness of the standard LPA, its performance is not appropriate for community detection. Thus, we proposed an improved version of LPA called AntLP by assigning weights edges using several similarity indices and also using an ACO algorithm for optimising modularity of the communities found by the algorithm. In the AntLP algorithm, ants guide the search process into communities with high modularity and use similarity indices for detecting the best communities, and this step significantly improves the solution quality compared to other community detection algorithm. The experimental results on some well-known social network datasets demonstrated that the AntLP is better than LPA and some community detection algorithms for social networks in terms of modularity and NMI.

7 Acknowledgment

This research was in part, supported by a grant from IPM (no. CS1398-4-222).

8 References

[1] Watts, D.J., Strogatz, S.H.: ‘Collective dynamics of ‘small-world’ networks’, Nature, 1998, 393, (6684), pp. 440–442
[2] Clauset, A., Shalizi, C.R., Newman, M.E.J.: ’Power-law distributions in empirical data’, SIAM Rev., 2007, 51, (4), p. 43
[3] Girvan, M., Newman, M.E.J.: ‘Community structure in social and biological networks’, Proc. Natl. Acad. Sci. 2001, 99, (12), pp. 7821–7826
[4] Rezvani, A., Moradabadi, B., Ghavipour, M., et al.: ’Learning automata approach for social networks’, vol. 820 (Springer International Publishing, Cham, Switzerland, 2019)
[5] Kou, F., Du, J., He, Y., et al.: ’Social network search based on semantic analysis and learning’, CAAI Trans. Intell. Technol., 2016, 1, (4), pp. 293–302
[6] Amiri, F., Yazdani, N., Fallah, H., et al.: ’A novel community detection algorithm for privacy preservation in social networks’, Intell. Inf., 2013, 18, pp. 441–450
[7] Rezvani, A., Moradabadi, B., Ghavipour, M., et al.: ’Social community detection’, ‘Learning Automata Approach for Social Networks’, (Cham, Switzerland, 2019), pp. 151–168
[8] Gong, M., Li, G., Wang, Z., et al.: ’An efficient shortest path approach for social networks based on community structure’, CAAI Trans. Intell. Technol., 2016, 1, (13), pp. 114–123
[9] Du, W., He, X.: ’A common strategy to improve community detection performance based on the nodes’ property’, CAAI Trans. Intell. Technol., 2017, 2, (2), pp. 68–73
[10] Rossetti, G., Casabrot, R.: ’Community discovery in dynamic networks’, ACM Comput. Surv., 2018, 51, (2), pp. 1–37
[11] Fortunato, S.: ’Community detection in graphs’, Phys. Rep., 2010, 486, (3–5), pp. 75–174
[12] Dinh, N.T., Thai, M.T.: ’Community detection in scale-free networks: approximation algorithms for maximizing modularity’, IEEE J. Sel. Areas Commun., 2013, 31, (6), pp. 997–1006
[13] Yang, Y., Sun, P.G., Hu, X., et al.: ’Closed walks for community detection’, Phys. A Stat. Mech. Appl., 2014, 397, pp. 129–143
[14] Qi, X., Tang, W., Wu, Y., et al.: ’Optimal local community detection in social networks based on density drop of subgraphs’, Pattern Recognit. Lett., 2014, 36, (1), pp. 46–53
[15] Atzmueller, M., Doerfler, S., Mitzlaff, F.: ’Description-oriented community detection using exhaustive subgroup discovery’, Inf. Sci. (Ny), 2016, 329, pp. 965–984
[16] Rechian, A., Feizi-Derakhshi, M.R., Aghdasi, H.S.: ’Community detection in social networks with node attributes based on multi-objective biogeography based optimization’, Eng. Appl. Artif. Intell., 2017, 62, pp. 51–67
[17] Hervás-Martínez, A., Arxi, R.: ’Memory-based label propagation algorithm for community detection in social networks’. 2015 The Int. Symp. on Artificial Intelligence and Signal Processing (AISP), Tehran, Iran, 2015, pp. 256–260
[18] Zhang, H., Chen, X., Li, J., et al.: ’Fuzzy community detection via modularity guided membership-degree propagation’, Pattern Recognit. Lett., 2016, 76, pp. 66–72
[19] Elyasi, M., Meybodi, M., Rezvani, A., et al.: ’A fast algorithm for overlapping community detection’, 2016 Eighth Int. Conf. on Information and Knowledge Technology (IKT), Hamedan, Iran, 2016, vol. 39, no. 4, pp. 221–226
[20] Staudt, C.L., Meyerhenke, H.: ’Engineering parallel algorithms for community detection in massive networks’, IEEE Trans. Parallel Distrub. Syst., 2016, 27, (1), pp. 171–184
[21] Zhang, Y., Liu, Y., Li, J., et al.: ’WODCA a whale optimization based community detection algorithm’, Phys. A Stat. Mech. Appl., 2020, 539, pp. 1229372020
[22] Zeng, X., Wang, W., Chen, C., et al.: ’A consensus community-based particle swarm optimization for dynamic community detection’, IEEE Trans. Cybern., 2020, doi: 10.1109/TCYB.2019.2938895
[23] Bihr, A., Johnson, P., Randles, M.: ’An optimisation tool for robust community detection algorithms using content and topology information’, J. Supercomput., 2020, doi:10.1007/s11227-019-03018-x
[24] Xu, Y.: ’A spectral method to detect community structure based on the communicability modularity’, Phys. A Stat. Mech. Appl., 2020, 537, pp. 122751
[25] Luo, L., Liu, K., Guo, B., et al.: ’User interaction-oriented community detection based on cascading analysis’, Inf. Sci., 2020, 510B, pp. 70–88
[26] Khomami, M.M.D., Rezvani, A., Meybodi, M.R.: ’Distributed learning automata-based algorithm for community detection in complex networks’, Int. J. Mod. Phys. B, 2016, 30, (8), pp. 1650042
[27] Gharghouri, M., Khomami, M.M.D., Bagherpour, N., et al.: ’An extended distributed learning automata based algorithm for solving the community detection problem in social networks’. 2017 Iranian Conf. on Electrical Engineering (ICEE), Tehran, Iran, 2017, pp. 1520–1526
[28] Khomami, M.M.D., Rezvani, A., Meybodi, M.R.: ’A new cellular learning automata-based algorithm for community detection in complex social networks’, J. Comput. Sci., 2018, 24, pp. 413–426
[29] Chen, B., Jiang, J., Hu, K., et al.: ’Enhancing betweenness algorithm for detecting communities in complex networks’, Mod. Phys. Lett. B, 2014, 28, (9), pp. 1450074
[30] Lu, Z., Sun, X., Wen, Y., et al.: ’Algorithms and applications for community detection in weighted networks’, IEEE Trans. Parallel Distrib. Syst., 2015, 26, (11), pp. 2916–2926
[31] de Andrade, L.P., Espindola, R.P., Ebeneck, N.F.F.: ’Community detection by an efficient ant colony approach’. Int. Symp. on Methodologies for Intelligent Systems, Cham, Switzerland, 2014 (LNCS, 8502), pp. 1–9
[32] Jin, D., Liu, D., Yang, B., et al.: ’Ant colony optimization with a new random walk model for community detection in complex networks’, Adv. Complex Syst., 2011, 14, (5), pp. 975–985
[33] Sadi, S., Oguducu, S., Uyar, A.S.: ’An efficient community detection method using parallel clique-finding ants’, IEEE Congress on Evolutionary Computation, Barcelona, Spain, 2010, pp. 1–7
[34] Li, P., Zhang, S., Zhou, Z.: ’A decomposition-based ant colony optimization algorithm for the multi-objective community detection’, J. Ambient Intell. Humaniz. Comput., 2019, doi: 10.1007/s12652-019-01241-1
[35] Zhou, X., Liu, Y., Zhang, J., et al.: ’An ant colony based algorithm for overlapping community detection in complex networks’, Phys. A Stat. Mech. Appl., 2015, 427, pp. 289–301
[36] Cui, Y., Wang, X.: ’Uncovering overlapping community structures by the key bi-community and intimate degree in bipartite networks’, Phys. A Stat. Mech. Appl., 2014, 407, pp. 7–14

This is an open access article published by the IET, Chinese Association for Artificial Intelligence and Chongqing University of Technology under the Creative Commons Attribution -NonCommercial License (http://creativecommons.org/licenses/by-nc/3.0/)
[37] Cui, Y., Wang, X., Eustace, J.: ‘Detecting community structure via the maximal sub-graphs and belonging degrees in complex networks’, Phys. A Stat. Mech. Appl., 2014, 416, pp. 198–207

[38] Ben Romdhane, L., Chaabani, Y., Zardi, H.: ‘A robust ant colony optimization-based algorithm for community mining in large scale oriented social graphs’, Expert Syst. Appl., 2013, 40, (14), pp. 5709–5718

[39] Raghavan, U.N., Albert, R., Kumara, S.: ‘Near linear time algorithm to detect community structures in large-scale networks’, Phys. Rev. E, 2007, 76, (3), p. 036106

[40] Rezvanian, A., Moradabadi, B., Ghavipour, M., et al.: ‘Social link prediction’, Learn. Autom. Approach Social Netw., 2019, pp. 169–239

[41] Dorigo, M., Stützle, T.: ‘Ant colony optimization: overview and recent advances’ (Springer, Boston, MA, USA, 2010), pp. 227–263

[42] Clauset, A., Newman, M.E.J., Moore, C.: ‘Finding community structure in very large networks’, Phys. Rev. E, 2004, 70, (6), p. 066111

[43] Pizzuti, C.: ‘GA-Net: a genetic algorithm for community detection in social networks’. Int. Conf. on Parallel Problem Solving from Nature, 2008 (LNCS, 5199), pp. 1081–1090

[44] Zhao, Y., Jiang, W., Li, S., et al.: ‘A cellular learning automata based algorithm for detecting community structure in complex networks’, Neurocomputing, 2015, 151, (P3), pp. 1216–1226

[45] Rezapoor Mirsaleh, M., Reza Meybodi, M.: ‘A Michigan memetic algorithm for solving the community detection problem in complex network’, Neurocomputing, 2016, 214, pp. 535–545

[46] Lancichinetti, A., Fortunato, S.: ‘Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities’, Phys. Rev. E, 2009, 80, (1), p. 016118