Future Generation Computer Systems 88 (2018) 785–791

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Iterated Greedy algorithm for performing community detection in social networks

Jesús Sánchez-Oro [*], Abraham Duarte

Dept. Computer Sciences, Universidad Rey Juan Carlos, Tulipán s/n, 28933 Móstoles, Madrid, Spain

HIGHLIGHTS

- We propose a new algorithm for detecting communities in social networks.
- The algorithm optimizes the modularity of the communities detected.
- The proposed method favorably compares with the best previous method.
- The results highlight the relevance of using modularity for detecting communities.

ARTICLE INFO

Article history:
Received 20 October 2017
Received in revised form 28 May 2018
Accepted 7 June 2018
Available online 20 June 2018

Keywords:
Social networks
Community detection
Iterated greedy
Metaheuristics

ABSTRACT

The spreading of social networks in our society has aroused the interest of the scientific community in hard optimization problems related to them. Community detection is becoming one of the most challenging problems in social network analysis. The continuous growth of these networks makes exact methods for detecting communities not suitable for being used, since they require large computing times. In this paper, we propose a metaheuristic approach based on the Iterated Greedy methodology for detecting communities in large social networks. The computational results presented in this work show the relevance of the proposal when compared with traditional community detection algorithms in terms of both quality and computing time.

1. Introduction

Social networks have become one of the main media all over the world in the last years, as the number of users is in continuous growth [1,2]. The rationale behind this exponential expansion might be related to the immediacy of the information. Nowadays, any new information is firstly published in social networks and, after that, in traditional media. Furthermore, users are getting used to obtain information from social networks instead of considering traditional media [3].

The transmission of information through social networks has created new lines of research, like viral information detection [4], analysis of the relevance of social network users [5], and community detection [6], among others (see [7,8]). In this work, we focus on the detection of communities in social networks, which is a relevant problem not only in social network analysis, but also in areas like natural disaster management [9], biology [10], semantic web [11], or cybersecurity issues [12].

The community detection problem consists in dividing a network of users into an unknown number of groups, with the objective of optimizing the value of a function that determines the quality of the division. Although this problem has been widely studied from both, exact and heuristic perspectives [13–15], the best objective function used to find the best partition of a network in groups is still under discussion [16].

The quality of a community detection over a social graph has been widely studied from both exact and heuristic perspectives. The Louvain algorithm [13] is focused on maximizing the modularity. It is a heuristic algorithm that follows a greedy criterion to insert a node in a community. Specifically, a node will be added to a community if and only if it leads to an increment in the modularity value, stopping when no improvement is found. The Infomap algorithm [17] focuses on finding the minimum information description of a random walk, using the Minimum Description Length objective function. Finally, the Label Propagation algorithm [14] tries to find the best communities by iteratively assigning to each node the community where most of its adjacent nodes belong to, trying to maximize the modularity metric. These algorithms are based on the structure of the network in order to improve the community detection. However, some works include additional

* Corresponding author.
E-mail address: jesus.sanchezoro@urjc.es (J. Sánchez-Oro).

https://doi.org/10.1016/j.future.2018.06.010
0167-739X/© 2018 Elsevier B.V. All rights reserved.

information of the network in the community detection, like the traffic between two nodes [18] or in wireless sensor networks [19].

As far as we know, the best heuristic method for finding high quality communities in graphs derived from social network is a bioinspired algorithm based on Ant Colony Optimization [6]. This algorithm is focused on detecting communities in Ego Networks, where a user (node) is selected as the center of the graph (Ego) and then all the connected users (nodes) are added, together with the relations (edges) between each pair of users. The main objective in community detection over Ego Networks is to find the groups connected to a certain user in a social network [20].

In this paper we propose a new Iterated Greedy algorithm [21] for detecting communities in Ego Networks. This algorithm starts from an initial solution, constructed by a heuristic procedure. Then, it iteratively improves it by performing two well differenced phases: destruction and reconstruction.

The remaining of the paper is structured as follows: Section 2 formally describes the problem under consideration, Section 3 presents the algorithmic description of the Iterated Greedy method proposed, Section 4 describes the computational experiments performed for analyzing the quality of the proposal, and, finally, Section 5 summarizes the conclusions derived from this research.

2. Problem definition

Before presenting the problem under consideration, it is necessary to provide a formal definition of a network of users. Specifically, given a set of users connected in a social network, we define the graph $G = (V, E)$ where V is the set of n nodes (each user is represented by a unique node) and E is the set of m edges. An edge $(u, v) \in E$, with $u, v \in V$ represents that there is a connection between users u and v. Notice that the meaning of the connection (both users are friends, work in the same company, etc.) is totally dependent of the nature of the social network.

It is important to remark that we are facing an unsupervised clustering problem, since the optimal clustering is not usually known in advance. Therefore, we need to focus on metrics that are able to evaluate the quality of a partition without knowing the optimal one. The most relevant metrics are based on maximizing the density of edges that connect nodes in the same cluster (intra-cluster edges) and, at the same time, minimizing the number of edges that connect nodes located in different clusters (inter-cluster edges).

In this context, there are three main metrics for evaluating the quality of a given partition [22]: modularity, conductance, and coverage. The three considered metrics are normalized in the range 0–1, where 1 is the optimal score for coverage and modularity, an 0 for conductance. Notice that not all networks can reach the optimal score due to their internal structure.

Before formally defining each metric, it is necessary to introduce the solution structure for the Community Detection Problem (CDP). A solution (or partition) for the CDP is represented as the set of clusters $\mathcal{K}$, where each node $v \in V$ is assigned to a different cluster K_i, with $\bigcup_{1 \le i \le |\mathcal{K}|} K_i = V$ and $K_i \cap K_j = \emptyset$, with $1 \le i, j \le |\mathcal{K}|$. Additionally, φ is a function defined as $\varphi : V \to \{1, 2, \ldots |\mathcal{K}|$ that represents the cluster to which it belongs a particular node. For example, for a given node $v \in V$, $\varphi(v) = 2$ would indicate that v is located at cluster K_2.

The most simple metric is the coverage [23], which analyzes the number of intra-cluster edges in a given solution with respect to the total number of edges in the network. More formally,

$$Cv(G, \varphi) = \frac{|(u, v) \in E : \varphi(u) = \varphi(v)|}{|E|}$$

Notice that the optimization of this metric can eventually lead to the trivial clustering where all the nodes are in the same cluster.

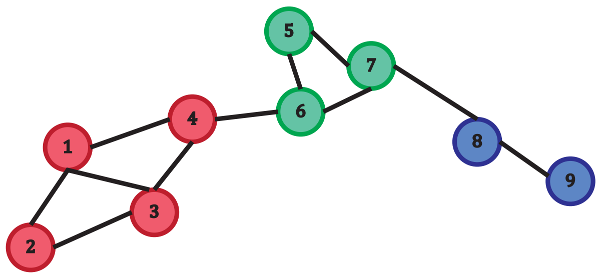


Fig. 1. Example graph with a possible community detection (each community corresponds to a different color). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

The conductance of a cluster compares how many inter-cluster edges are in a particular cluster with respect to the total number of edges with an endpoint in that cluster or those with no endpoint in the cluster. In mathematical terms,

$$Cn_k(G, \varphi) = \frac{|(u, v) \in E : \varphi(u) \neq \varphi(v)|}{\min\left(E_k, \overline{E_k}\right)}$$

where $E_k = |(u, v) \in E : \varphi(u) = k \vee \varphi(v) = k|$ is the set of edges with an endpoint in cluster k and $\overline{E_k} = |(u, v) \in E : \varphi(u) \neq k \wedge \varphi(v) \neq k|$ is the set of edges with no endpoint in cluster k. Then, the conductance of a solution φ for graph G is evaluated as the average conductance among all clusters in the solution. More formally,

$$Cn(G, \varphi) = \frac{1}{|\mathcal{K}|} \sum_{k=1}^{|\mathcal{K}|} Cn_k(G, \varphi)$$

where $|\varphi|$ is computed as the number of clusters in solution φ.

The conductance can also be computed by considering intra-cluster edges. The aforementioned definition based on inter-cluster edges is focused on the inter-cluster sparsity, while the one based on intra-cluster edges emphasizes intra-cluster density [24].

The last metric considered is the modularity of a solution, which compares the actual intra-cluster edges with the probability of finding that edge in a random graph [25,26]. This metric has been widely used by the most relevant clustering algorithms in the literature [27,28], although it presents some limitations when considering large scale networks [29]. Modularity of a solution φ for a graph G is formally defined as:

$$Md(G, \varphi) = \sum_{k=1}^{|\mathcal{K}|} \left(e_{kk} - a_k^2\right)$$

where

$$e_{kk} = |(u, v) \in E : \varphi(u) = \varphi(v)| \,/ |E|$$

represents the probability of intra-cluster edges in cluster k, while

$$a_k = |(u, v) \in E : \varphi(u) = k \vee \varphi(v) = k| \,/ |E|$$

represents the probability of an edge with at least one endpoint in cluster k.

Fig. 1 shows an example graph where three communities have been detected, each one highlighted with a different color, together with the value of each one of the considered metrics. Specifically, the red community contains nodes 1, 2, 3, and 4; the green one contains nodes 5, 6, and 7; and the last one (blue) contains nodes 8 and 9. The values for the aforementioned metrics are $Cv(G, \varphi) = 0.82$, $Cn(G, \varphi) = 0.36$, and $Md(G, \varphi) = 0.42$.

This work is focused on optimizing the modularity of the community detection, since it is considered the most robust metric to evaluate the quality of the partition for several community detection algorithms [30].

3. Iterated greedy

Metaheuristics are a set of widely recognized strategies whose main objective is to generate high quality solutions for hard optimization problems requiring, in general, short computing times [31]. Since they are approximate algorithms, metaheuristics cannot guarantee the optimality of the obtained solutions, although they have been proven to be effective and efficient in several recent works [32,33].

Iterated Greedy (IG) is a metaheuristic framework originally proposed in 2007 for solving a scheduling problem [21,34], but it has evolved to become one of most used metaheuristics in the last years [35,36].

The basis of IG resides in exploring the search space by alternating intensification and diversification iteratively in two consecutive phases: destruction and reconstruction. Algorithm 1 shows the pseudocode of this framework.

Algorithm 1 IteratedGreedy(G, φ, it, β)

```
1: $\varphi_b \leftarrow \varphi$
2: for $i \in 1 \ldots it$ do
3:     $\varphi' \leftarrow$ Destruct($\varphi, \beta$)
4:     $\varphi'' \leftarrow$ Reconstruct($\varphi'$)
5:     $\varphi \leftarrow$ AcceptanceCriterion($\varphi, \varphi''$)
6:     if $Md(\varphi) > Md(\varphi_b)$ then        ▷ Improve
7:         $\varphi_b \leftarrow \varphi$
8:     end if
9: end for
10: return $\varphi_b$
```

The method starts from an initial solution φ for a given network G. The initial solution is generated using the constructive method described in Section 3.1. The stopping criterion for IG is usually either the number of iterations or the computing time. The algorithm proposed in this work stops after performing *it* iterations (steps 2–9). Each iteration starts with a destruction phase (step 3), which is presented in Section 3.2. After that, the solution obtained is subjected to a reconstruction process (step 4), described in Section 3.2. The reconstructed solution φ'' is then accepted if it surpasses the acceptance criterion selected (step 5). In this work, we have not established any acceptance criterion, so every solution is accepted after reconstruction. Finally, if the reconstructed solution φ'' outperforms, in terms of modularity, the incumbent solution φ_b, it is updated (steps 6–8). The method ends after performing *it* iterations, returning the best solution found during the search, φ_b.

It is worth mentioning that due to the semi-random nature of the constructive procedure described in Section 3.1, it is interesting to consider not only one initial solution but several different and diverse solutions. For this reason, we propose a Multi-Start Iterated Greedy (MSIG) algorithm which executes the IG algorithm described in Algorithm 1 over each solution generated by the constructive procedure.

3.1. Constructive procedure

The method for generating the initial solution, called Greedy Constructive Procedure (GCP), follows a traditional greedy approach, where each node is added to the best cluster according to a greedy function value. Additionally, the method is randomized in order to increase the diversity of the solutions generated. Algorithm 2 presents the pseudocode of GCP.

The algorithm starts selecting at random the first node to be added to a cluster among all nodes in V (step 1). The first cluster is

Algorithm 2 GCP($G = (V, E)$)

```
1: $v \leftarrow$ SelectRandom($V$)
2: $K_0 \leftarrow \{v\}$
3: add($\varphi, K_0$)
4: $CL \leftarrow V \setminus \{v\}$
5: while $CL \neq \emptyset$ do
6:     $v \leftarrow$ SelectRandom($CL$)
7:     $Md_b \leftarrow \max_{K_i \in \varphi} Md\,(\varphi : K_i \leftarrow K_i \cup \{v\})$
8:     if $Md_b > Md(\varphi)$ then
9:         $K_i \leftarrow \arg\max_{K_i \in \varphi} Md\,(\varphi : K_i \leftarrow K_i \cup \{v\})$
10:        $K_i \leftarrow K_i \cup \{v\}$
11:    else
12:        $K' \leftarrow \{v\}$
13:        add($\varphi, K'$)
14:    end if
15:    $CL \leftarrow CL \setminus \{v\}$
16: end while
17: return $\varphi$
```

created containing the corresponding node and it is added to the solution under construction φ (steps 2 and 3). The method then creates a candidate list CL with all the nodes in V except the one previously added (step 4).

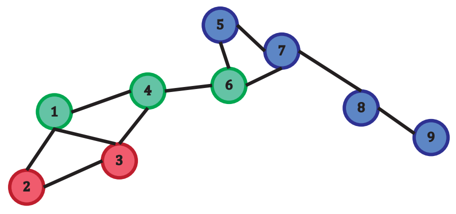
GCP iterates until the candidate list is empty, adding a new node to a cluster in each iteration as follows (steps 5–16). The next node is selected at random from the CL (step 6) and GCP evaluates the resulting modularity when inserting it in each cluster (step 7). If the best modularity obtained in the evaluation is larger than the current modularity of the solution (improvement), the node is inserted in the cluster that produces the best modularity (steps 8–10). Otherwise, a new cluster is created and the selected node is inserted in it (steps 11–13). Finally, the node is removed from the candidate list (step 15). The method returns a feasible solution with every node assigned to a cluster.

3.2. Destruction and reconstruction phase

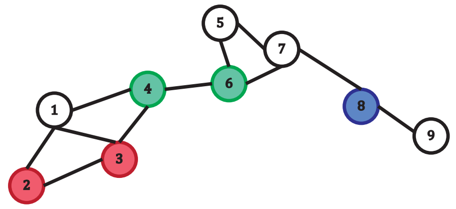
The destruction phase of the IG algorithm starts from a feasible solution generated with the constructive method described in Section 3.1. This phase is devoted to perturb the incumbent solution and requires from a parameter β that controls the perturbation size (best β values are discussed in Section 4). Specifically, the destruction phase consist of randomly removing $\beta \cdot n$ nodes from their corresponding clusters, which will be later reassigned in the reconstruction phase.

Notice that it is interesting to explore solutions with different number of clusters, in order to explore a wider portion of the solution space. This is mainly because community detection belongs to the family of unsupervised clustering problems, where the optimal number of clusters is not known *a priori*. In order to do so, the proposed destruction method considers that, if a node v is removed from a cluster K_i with $|K_i| = 1$ (i.e., v is the only node in K_i), then the cluster is also removed from the solution, reducing the number of clusters in the incumbent solution.

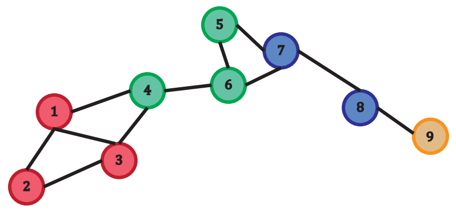
After performing the destruction method, the reconstruction phase is responsible for assigning the previously removed nodes to a new cluster. This method selects, for each node, the cluster that produces the solution with the largest modularity value. Furthermore, to maintain the diversity in the number of clusters, the method also considers the creation of a new cluster where the node under evaluation is added. The goal is to test if the best modularity value is produced when creating a new cluster instead of inserting the node in one of the available ones. The reconstruction



(a) Initial solution with three clusters.



(b) Resulting solution after destruction phase where nodes 1, 5, 7, and 9 have been removed from their respective clusters.



(c) Reconstructed solution where a new cluster have been added.

Fig. 2. Example of the destruction and reconstruction phase performed over an initial solution with three clusters. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

phase ends when all the nodes have been reassigned to a new cluster.

Fig. 2 shows an example of performing the destruction and reconstruction phase over an initial solution. Fig. 2a depicts again the example with three clusters represented in red, blue, and green colors. The results of the destruction phase is presented in Fig. 2b, where nodes 1, 5, 7, and 9 have been selected and removed from their corresponding clusters. After that, the reconstruction phase must assign a cluster to these nodes, resulting in solution depicted in Fig. 2c, where a new cluster has been added in order to increase modularity.

4. Computational experiments

This section is devoted to provide a complete analysis of the computational experiments performed to evaluate the results obtained by the proposed MSIG algorithm. All the algorithms have been developed in Java SE 8 and they have been tested in an Intel Core i7 CPU 920 (2.67 GHz) and 8 GB RAM.

The set of instances considered have been derived from two of the most used social networks: Facebook and Twitter. Specifically, we have considered 10 ego networks from Facebook and 973 from Twitter, with the number of nodes ranging from 10 to 1045 and the number of edges from 5 to 60050. All the datasets have been obtained from the Stanford Network Analysis Project,[1] specifically from the Facebook and Twitter Ego datasets [37]. Additionally, we have made publicly available[2] the instance files used in this work and the individual results obtained for each instance[3] in order to ease further comparisons.

The computational experiments are divided into two different stages: preliminary and final experimentation. The former is devoted to perform a deeper analysis on the configuration of the parameters of the MSIG algorithm, while the latter is intended to compare the best configuration of IG with the best previous method found in the state of the art. Specifically, the results are compared with the ones obtained by the Ant Colony Optimization (ACO) algorithm presented in [6].

4.1. Preliminary experimentation

Preliminary experiments are performed over a subset of 84 out of 983 representative instances in order to avoid overtraining in the MSIG algorithm. The first experiment is intended to select the best β parameter for the algorithm (i.e., the percentage of nodes removed from the solution in the destruction phase). Specifically, we compare the results of the constructive procedure (GCP) isolated with the results obtained by executing the MSIG with $\beta = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]$. We have generated 100 different solutions for both, the constructive and MSIG algorithm. Considering that MSIG requires more computing time, we execute 10 independent iterations. Fig. 3 depicts the results in terms of the deviation (in percentage) with respect to the best solution found in the experiment, for each instance, Dev(%), and the number of times that each method matches the best solution in the experiment, for each instance, #Best. Regarding the computing time, GCP is the fastest method, requiring less than a second on average per instance, while all the MSIG variants require about 15 s on average per instance.

The variant with the smallest deviation (blue bars) and the largest number of best solutions reached (orange line) is MSIG with $\beta = 0.5$, followed by $\beta = 0.4$ with a smaller number of best solutions found and $\beta = 0.6$ with slightly larger deviation. It is important to remark that these results are in line with the idea of selecting small values for β in Iterated Greedy. The rationale behind this is that selecting large β values will results in a completely different solution and, in that case, it is recommended to construct a new solution from scratch.

4.2. Ground truth

The instances used in this work are available together with the ground truth of the communities given by the ego user (i.e., the central user of the ego network). It is important to remark that these communities have been defined by the user, without taking into account any objective metric and they are based on the subjective judgment of the ego user (close friends, coworkers, etc.). However, this ground truth may not be the best partition when considering objective metrics based on the network structure, since the ego user does not have a global perspective of the network when defining the communities. In order to test this hypothesis, we compare the coverage and conductance of the communities given by the ground truth with those obtained by the best variant of the proposed algorithm, MSIG(0.5), considering the complete set of instances. Table 1 presents the results derived from comparing the user partition (Ego user) with the MSIG(0.5).

Analyzing the results over Facebook networks, MSIG obtains consistently better results in both metrics. Attending to these results, we observe that the perception that an ego user has about the communities depends on the specific social network. In particular, the modularity values are close in Facebook while they are considerably different in Twitter. This result can be partially explained by

1 http://snap.stanford.edu/snappy/index.html.
2 http://www.optsicom.es/sna/sna_instances.zip.
3 http://www.optsicom.es/sna/results.xlsx.

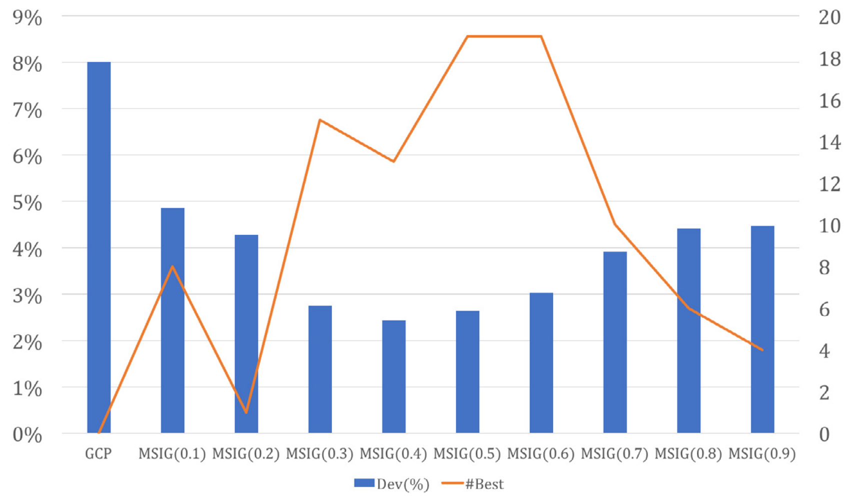


Fig. 3. Comparison among GCP and MSIG with different β values (between parenthesis).

Table 1
Comparison of the modularity obtained with the best MSIG variant versus the ground truth given by the ego user.

	Metric	Facebook	Twitter
Ego user	Coverage	0.5986	0.6028
	Conductance	0.6744	0.5416
MSIG(0.5)	Coverage	0.9431	0.8062
	Conductance	0.2802	0.5284

Table 2
Comparison of the modularity obtained by the best MSIG variant versus the best previous method (ACO) with topological (ACO-T), profile (ACO-P), and combined (ACO-M) information used.

	Facebook	
	Coverage	Conductance
ACO-M	0.5178 ± 0.1603	0.8821 ± 0.1179
ACO-P	0.5091 ± 0.1247	0.8919 ± 0.1194
ACO-T	0.4978 ± 0.1580	0.8861 ± 0.1129
MSIG(0.5)	0.9431 ± 0.0343	0.1938 ± 0.1088
	Twitter	
	Coverage	Conductance
ACO-M	0.2182 ± 0.1682	0.9440 ± 0.0653
ACO-P	0.2216 ± 0.1743	0.9440 ± 0.0648
ACO-T	0.2190 ± 0.1677	0.9445 ± 0.0651
MSIG(0.5)	0.7198 ± 0.1226	0.5284 ± 0.2673

the fact that, on the one hand, Facebook is a social network often used to share media information with personal friends. Then, the ego user usually knows the complete network structure. On the other hand, Twitter is intended to share short messages with a larger audience in which the ego user does not necessarily know all the people. Furthermore, in order to be Facebook friends is strictly necessary that both users accept a friendship request, while in Twitter a user can follow or be followed by another without permissions. Therefore, it is easier to know people which we are connected with in Facebook than in Twitter. Then, it eases the division into communities by the ego user.

4.3. *Final experiment*

The last experiment is devoted to compare the results obtained by MSIG with the best previous method found in the state of the art [6], which is an Ant Colony Optimization (ACO) algorithm. It is important to remark that this algorithm is designed for optimizing the Omega Index value, which can be used when the ground truth of the network is known *a priori*. However, the objective of the algorithm proposed in this work is to provide a community detection which optimizes the modularity value. It is worth mentioning that following this approach, we provide two objective metrics useful when the ground truth is not available.

Each algorithm have been evaluated considering the conductance and coverage metrics described in Section 2 in order to evaluate the quality of the community detection with a metric that has not been optimized, proving the robustness of the algorithm.

We have included three variants of the ACO algorithm: ACO-T, which uses the topological information of the network; ACO-P, which is focused on the profile information; and ACO-M, which combines both topological and profile information. Table 2 shows, for each method, the corresponding metric value (coverage and conductance) averaged over the instances in each benchmark (Facebook and Twitter). In order to show the robustness of the method, we also report the standard deviation of each metric.

Considering the ACO variants, there are only slight differences in the average objective function values obtained with each method. Specifically, ACO-M is the best ACO variant in the two metrics when considering the Facebook dataset, while ACO-P is the best variant for the Twitter benchmark in all the metrics. However, the proposed MSIG algorithm clearly outperforms every ACO variant when considering the two metrics. Again, these results confirm the hypothesis that it easier to find partitions with larger modularity values in Facebook that in Twitter.

With the aim of providing a fair comparison among the analyzed algorithms, the Iterated Greedy algorithm has been executed during a similar number of iterations than the previous ones. Specifically, ACO variants, as stated in the original work [6], were configured with 10 ants, 50 steps and 10 repetitions, resulting in 5000 iterations. Iterated Greedy performs 100 constructions, applying the combination of destruction and reconstruction 10 times for each constructed solution, resulting in 1000 iterations.

The computing time required by the MSIG algorithm is, on average, 15 s per instance, resulting in not only a state-of-the-art method for detecting communities in terms of performance, but also in terms of required computing time. Therefore, our proposed method can be used in real-time applications.

5. Conclusions

This paper presented a Multi-start Iterated Greedy algorithm for detecting communities in social networks. In particular, it is focused on the optimization of the modularity value, which is a metric traditionally used in unsupervised community detection, where the optimal solution is not known *a priori*. The MSIG method is conformed by a new procedure for generating initial solutions that follows a greedy criterion maximizing the modularity value. Additionally, new destructive and constructive phases for escaping from local optima are proposed to increase the diversification component of the algorithm, resulting in a wider exploration of the search space.

The results obtained were compared with the best previous methods found in the literature, emerging MSIG as the best performing procedure. Furthermore, we highlighted the problems derived from getting a ground truth directly from the user, without considering any objective metrics. Specifically, in those social networks where the ego user has not the complete information, subjective judgments usually drive to confusing partitions.

There are several avenues to continue this research. In particular, the study of the combination of two or more objective metrics in order to find even better partitions. Finally, this approach could be easily adapted to include specific information of the social network by assigning weight to the nodes and/or edges of the ego network.

Acknowledgments

This work has been partially founded by Ministerio de Economía y Competitividad, Spain with grants refs. TIN2015-65460-C2-2-P and S2013/ICE-2894, respectively.

References

[1] L. Backstrom, D. Huttenlocher, J. Kleinberg, X. Lan, Group formation in large social networks: Membership, growth, and evolution, in: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, in: KDD '06, ACM, New York, NY, USA, 2006, pp. 44–54.

[2] F.Y. Wang, D. Zeng, J.A. Hendler, Q. Zhang, Z. Feng, Y. Gao, H. Wang, G. Lai, A study of the human flesh search engine: Crowd-powered expansion of online knowledge, Computer 43 (8) (2010) 45–53.

[3] M. Bruhn, V. Schoenmueller, D. Schäfer, Are social media replacing traditional media in terms of brand equity creation? Manag. Res. Rev. 35 (9) (2012) 770–790.

[4] W. Chen, C. Wang, Y. Wang, Scalable influence maximization for prevalent viral marketing in large-scale social networks, in: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, in: KDD '10, ACM, New York, NY, USA, 2010, pp. 1029–1038.

[5] L.C. Freeman, Centrality in social networks conceptual clarification, Soc. Networks 1 (3) (1978) 215–239.

[6] A. Gonzalez Pardo, J.J. Jung, D. Camacho, ACO-based clustering for Ego Network analysis, Future Gener. Comput. Syst. 66 (Supplement C) (2017) 160–170.

[7] M. Hong, J.J. Jung, D. Camacho, GRSAT: A novel method on group recommendation by social affinity and trustworthiness, Cybern. Syst. 48 (3) (2017) 140–161.

[8] R. Lara-Cabrera, A. González-Pardo, K. Benouaret, N. Faci, D. Benslimane, D. Camacho, Measuring the radicalisation risk in social networks, IEEE Access 5 (2017) 10892–10900.

[9] T. Sakaki, M. Okazaki, Y. Matsuo, Earthquake shakes twitter users: Real-time event detection by social sensors, in: Proceedings of the 19th International Conference on World Wide Web, in: WWW '10, ACM, New York, NY, USA, 2010, pp. 851–860.

[10] E. Boros, P.L. Hammer, Pseudo-Boolean optimization, Discrete Appl. Math. 123 (1) (2002) 155–225.

[11] Y. Matsuo, J. Mori, M. Hamasaki, T. Nishimura, H. Takeda, K. Hasida, M. Ishizuka, POLYPHONET: An advanced social network extraction system from the Web, Web Semant. Sci. Serv. Agents World Wide Web 5 (4) (2007) 262–278.

[12] H. Yu, P.B. Gibbons, M. Kaminsky, F. Xiao, Sybillimit: A near-optimal social network defense against sybil attacks, in: 2008 IEEE Symposium on Security and Privacy (Sp 2008), 2008, pp. 3–17.

[13] V.D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech. Theory Exp. 2008 (10) (2008) P10008.

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

[15] G. Tibély, J. Kertész, On the equivalence of the label propagation method of community detection and a Potts model approach, Physica A 387 (19) (2008) 4982–4984.

[16] A. Lancichinetti, S. Fortunato, Community detection algorithms: a comparative analysis, Phys. Rev. E 80 (5) (2009) 056117.

[17] M. Rosvall, C.T. Bergstrom, Maps of random walks on complex networks reveal community structure, Proc. Nat. Acad. Sci. 105 (4) (2008) 1118–1123.

[18] M. Naldi, S. Salcedo-Sanz, L. Carro-Calvo, L. Laura, A. Portilla-Figueras, G.F. Italiano, A traffic-based evolutionary algorithm for network clustering, Appl. Soft Comput. 13 (11) (2013) 4303–4319.

[19] B.A. Attea, E.A. Khalil, A new evolutionary based routing protocol for clustered heterogeneous wireless sensor networks, Appl. Soft Comput. 12 (7) (2012) 1950–1957.

[20] J. Xie, S. Kelley, B.K. Szymanski, Overlapping community detection in networks: The state-of-the-art and comparative study, ACM Comput. Surv. 45 (4) (2013) 1–35.

[21] R. Ruiz, T. Stützle, A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem, Eur. J. Oper. Res. 177 (3) (2007) 2033–2049.

[22] S. Emmons, S. Kobourov, M. Gallant, K. Börner, Analysis of network clustering algorithms and cluster quality metrics at scale, PloS One 11 (7) (2016) e0159161.

[23] S. Kobourov, S. Pupyrev, P. Simonetto, Visualizing graphs as maps with contiguous regions, in: N. Elmqvist, M. Hlawitschka, J. Kennedy (Eds.), EuroVis - Short Papers, The Eurographics Association, 2014.

[24] H. Almeida, D. Guedes, W. Meira, M.J. Zaki, Is there a best quality metric for graph clusters? in: D. Gunopulos, T. Hofmann, D. Malerba, M. Vazirgiannis (Eds.), Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2011, Athens, Greece, September 5-9, 2011. Proceedings, Part I, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 44–59.

[25] M.E.J. Newman, M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E 69 (2) (2004) 026113.

[26] M.E.J. Newman, Fast algorithm for detecting community structure in networks, Phys. Rev. E 69 (6) (2004) 066133.

[27] V. Blondel, J. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech. Theory Exp. 2008 (10) (2008) P10008.

[28] L. Waltman, N. van Eck, A smart local moving algorithm for large-scale modularity-based community detection, Eur. Phys. J. B 86 (11) (2013) 471.

[29] S. Fortunato, M. Barthélemy, Resolution limit in community detection, Proc. Natl. Acad. Sci. 104 (1) (2007) 36–41.

[30] Z. Yang, R. Algesheimer, C.J. Tessone, A comparative analysis of community detection algorithms on artificial networks, Sci. Rep. 6 (2016) 30750 EP–.

[31] R. Martí, P. Pardalos, M. Resende (Eds.), Handbook of Heuristics, Springer International Publishing, 2018.

[32] J. Sánchez Oro, A. Martínez Gavara, M. Laguna, R. Martí, A. Duarte, Variable neighborhood scatter search for the incremental graph drawing problem, Comput. Optim. Appl. (2017).

[33] B. Menéndez, E.G. Pardo, J. Sánchez-Oro, A. Duarte, Parallel variable neighborhood search for the minmax order batching problem, Int. Trans. Oper. Res. 24 (3) (2017) 635–662.

[34] R. Ruiz, T. Stützle, An Iterated Greedy heuristic for the sequence dependent setup times flowshop problem with makespan and weighted tardiness objectives, European J. Oper. Res. 187 (3) (2008) 1143–1159.

[35] J. Dubois Lacoste, F. Pagnozzi, T. Stützle, An iterated greedy algorithm with optimization of partial solutions for the makespan permutation flowshop problem, Comput. Oper. Res. 81 (Supplement C) (2017) 160–166.

[36] Z. Yuan, A. Fügenschuh, H. Homfeld, P. Balaprakash, T. Stützle, M. Schoch, Iterated greedy algorithms for a real-world cyclic train scheduling problem, in: M.J. Blesa, C. Blum, C. Cotta, A.J. Fernández, J.E. Gallardo, A. Roli, M. Sampels (Eds.), Hybrid Metaheuristics: 5th International Workshop, HM 2008, MáLaga, Spain, October 8-9, 2008. Proceedings, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 102–116.

[37] J. McAuley, J. Leskovec, Learning to discover social circles in ego networks, in: Proceedings of the 25th International Conference on Neural Information Processing Systems, in: NIPS'12, Curran Associates Inc., USA, 2012, pp. 539–547.

Jesús Sánchez-Oro was born in Madrid (Spain) on December 31, 1987. He holds a degree in Computer Science from the Universidad Rey Juan Carlos (2010), his Master's degree in Computer Vision from the Universidad Rey Juan Carlos in 2011, and his Ph.D. in Computer Science in 2016 from the Universidad Rey Juan Carlos. He is visiting professor at the Computer Science Department, and he is a member of the Group for Research on Algorithms For Optimization (GRAFO). His main research interests focus on Artificial Intelligence and Operations Research, specially in heuristics and metaheuristics for solving hard optimization problems.

Abraham Duarte was born in Hervás (Cáceres, Spain) on October 24, 1975. He received his M.S. degree in Physics Sciences (Electronic Speciality) from the Universidad Complutense de Madrid (UCM) in 1998 and his Ph.D. in Computer Science in 2004 from the Universidad Rey Juan Carlos (URJC). He is Associate Professor at the Computer Science Department where is the leader of the Group for Research on Algorithms For Optimization (GRAFO). His main research interests focus on the interface among Computer Science, Artificial Intelligence and Operations Research. Most of his publications deal with the development of metaheuristics procedures for optimization problems.