On the analysis of the influence of the evaluation metric in community detection using GRASP

Sergio Pérez-Peló
Dept. of Computer Sciences
Universidad Rey Juan Carlos
C/Tulipán, S/N, 28933, Móstoles, Spain
Email: sergio.perez.pelo@urjc.es

Jesús Sánchez-Oro
Dept. of Computer Sciences
Universidad Rey Juan Carlos
C/Tulipán, S/N, 28933, Móstoles, Spain
Email: jesus.sanchezoro@urjc.es

Abraham Duarte
Dept. of Computer Sciences
Universidad Rey Juan Carlos
C/Tulipán, S/N, 28933, Móstoles, Spain
Email: abraham.duarte@urjc.es

***Abstract*—Community detection in social networks is becoming one of the key tasks in social network analysis, since it helps analyze groups of users with similar interests, detect radicalisms, or reduce the size of the data to be analyzed, among other applications. This paper presents a metaheuristic approach based on Greedy Randomized Adaptive Search Procedure methodology for detecting communities in social networks. The community detection is modeled as an optimization problem where the objective function to be optimized is the modularity of the network, a well known metric in community detection. The results obtained outperforms traditional methods of community detection as Edge Betweenness, Fast Greedy and Infomap over a set of real-life instances derived from Twitter.**

I. Introduction

The evolution of social networks in the last decades has aroused the interest of scientist from different and diverse areas, from psychology to computer sciences. Millions of people constantly share all their personal and professional information in several social networks. Furthermore, social networks have become one of the most used information sources, mainly due to their ability to provide the user with real-time content. Social networks are not only a new way of communication, but also a powerful tool that can be used to gather information related to several issues: which political party is the favourite for the next elections, what are the most commented movies in the last year, which is the best rated restaurant in a certain area, etc.

The analysis of social networks has become one of the most popular and challenging tasks in data science [1]. One of the most tackled problems in social networks is the analysis of the relevance of the users in a given social network. The relevance of a user is usually related to the number of followers or friends that the user has in a certain social network. However, this concept can be extended since a user may be relevant not only if he/she is connected with a large number of users, but with users that are also relevant. Several metrics have been proposed for analyzing the relevance of a user in a social network, emerging the PageRank [2] as one of the most used.

The problem of evaluating the relevance of a user has evolved in a more complex problem which consists of detecting specific users, often named influencers, with certain personal attributes that can be personal (credibility or enthusiasm) or related to their social networks (connectiviy or centrality). These attributes allow them to influcence a large number of users either directly or indirectly [3].

Another important problem regarding the influence of people in other users is the analysis of sentiments in social networks. It is focused on finding out what do people think about a certain topic by analyzing the information they post in social networks. We refer the reader to [4] to find a complete survey on sentiment analysis techniques.

The previously described problems are related to individual users. However, there also exists some problems related to the structure of the network, devoted to find specific attributes and properties that can help to infer additional information of the social network. Community detection emerges as one of the most studied problems related to the structure of the network.

Most of the social networks present a common feature named community structure. Networks that have this property has the capacity to be divided into groups in such a way that the connections among users in the same group are dense, while connections among users in different groups are sparse. A connection can represent different features depending on the social network and the user profile, from professional relationships to friendships or hobbies in common. Community detection tasks are devoted to find and analyze these groups in order to better understand and visualize the structure of network and the relationships among their users.

Performing community detection algorithms over current social networks requires from a huge computationally effort mainly due to the continuous growth of social networks. Furthermore, since social networks are constantly changing (new friendships, mentions to users, viral information, etc.), it is interesting to perform the community detection in the shortest possible computing time, producing real-time information. These features make traditional exact methods not suitable for the current size of social networks, requiring from approximation algorithms in order to accelerate the process without losing quality. Recent works have tackled the community detection algorithm from a non-exact perspective in order to generate high quality solutions in short computing time [5].

The growth of social networks complicates their representation and understanding. The communities of a social network usually summarizes the whole network but reducing its size and, therefore, making it easier to analyze. Furthermore, detecting communities in social networks has several practical applications. Recommender systems leverage the data of similar users in order to suggest new contents. In order to find similar users in a network we can simply perform a community detection over the network [6], improving the results of the recommender system. Communities in social networks also identifies people with similar interests, allowing us to evaluate the popularity of a political party [7], or even to detect radicalisms in social networks [8].

The remaining of the paper is structured as follows: Section II formally defines the problem considered as well as the metrics proposed for the evaluation of solutions; Section III describes the traditional algorithms proposed for detecting communities in social networks; Section IV presents the new algorithm proposed for detecting communities; Section V shows the computational experiments performed to test the quality of the proposal; and finally Section VI draws some conclusions on the research.

II. Problem statement

A social network is represented as a graph $G = (V, E)$, where the set of vertices V, with $|V| = n$, represents the users of the network and the set of edges E, with $|E| = m$, represents relations between users belonging to the network. An edge $(v_1, v_2) \in E$, with $v_1, v_2 \in V$ can represent different types of relations depending on the social network under consideration. For example, in Twitter a relation represents that a user follows / is followed by an other user, while in LinkedIn it represents a professional relationship.

This work is focused on the Community Detection Problem (CDP), which involves grouping users of a social network into clusters. A desirable community in a social network is densely connected to the nodes in the same community and sparsely connected (or even unconnected) to nodes in other communities. Therefore, the main objective is to obtain groups or clusters of users that are similar among them and, at the same time, different to the users in other clusters with respect to a certain criterion.

A solution for the CDP is represented by a set of decision variables S, with $|S| = n$, where $S_v = j$ indicates that vertex v is assigned to cluster j in solution S. Figure 1(a) shows an example graph with 19 vertices and 31 edges derived from a social network. In this example, an edge represents a friendship relationship between two users; for instance, users `A` and `B` are friends, while users `A` and `C` are not friends but they have a friend in common, which is vertex `D`.

Figure 1(b) shows a possible solution S for the community detection problem, where each cluster is represented with a different color. Regarding the solution representation previously defined, we can check the cluster for each vertex. Table I shows the community in which each vertex has been inserted (for example, vertex `A` belongs to community 1, vertex `G` to community 2, and so on).

The CDP then can be modeled as an optimization problem which consists of finding a solution $S^\star$ that maximizes a certain objective function value. In mathematical terms,

$$S^\star \leftarrow \arg\max_{S \in \mathbb{S}} f(S)$$

where $\mathbb{S}$ is the set of all possible solutions for a given social network.

There exists a large variety of quality metrics that can be used as objective function for finding high quality solutions. Several metrics considers that the optimal partition (ground truth) is known beforehand, and tries to minimize the distance of the current partition with respect to the optimal one (e.g., Omega-Index [5]). However, this work considers networks where the optimal partition is not known. In this case, most of the metrics are focused on maximizing the density of intra-cluster edges (those connecting vertices of the same cluster) while minimizing inter-cluster edges (those connecting vertices in different clusters).

We consider two metrics that has been traditionally considered for optimizing the quality of a solution for the CDP: conductance and modularity [9]. For the sake of simplicity, all metrics are normalized in the range 0–1, where 1 indicates the value for the optimal partition and 0 is the expected score for a random assignment of users to clusters. Notice that in some of the metrics the optimal score of 1 is not possible for some networks due to their internal structure.

The first metric considered is the conductance [10]. Given a cluster k, its conductance, $Cn(k, G)$, is defined as the number of edges that connect vertices of different clusters divided by the minimum between the number of edges with an endpoint in the cluster and the number of edges with no endpoint in the cluster. A large value in the conductance indicates that there are several edges connecting vertices in different clusters and, therefore, the cluster does not represent a community. More formally,

$$\begin{aligned} Cn(k, G) &= \frac{|(v,u) \in E \,:\, S_v = k \wedge S_u \neq k|}{\min\{E_k, \overline{E_k}\}} \\ E_k &= |(v,u) \in E \,:\, S_v = k \vee S_u = k| \\ \overline{E_k} &= |(v,u) \in E \,:\, S_v \neq k \wedge S_u \neq k| \end{aligned}$$

Then, the conductance of a complete solution $Cn(S, G)$ is evaluated as the average conductance for all the clusters in the graph. In order to have a direct comparison with other metrics, we subtract that value from 1, so the objective is again to maximize the opposite of the conductance $\overline{Cn}(S, G)$ to produce high quality solutions (i.e., $\overline{Cn}(S, G) = 1 - Cn(S, G)$). Then, the opposite of the conductance value for the example depicted in Figure 1(b) is $\overline{Cn}(S, G) = 0.63$.

The third metric studied is the modularity [11], which evaluates, for each edge connecting vertices in the same cluster, the probability of the existence of that edge in a random graph. The modularity is evaluated as:

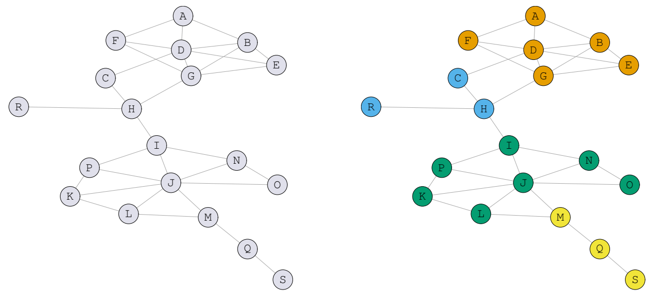


Fig. 1: 1(a) Example of a graph derived from a social network and 1(b) a possible solution for the community detection (each community is represented with a different color).

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	1	1	1	1	1	2	2	3	2	3	3	3	3	3	4	3	4	4

TABLE I: Cluster assigned to each vertex in the solution depicted in Figure 1(b).

$$Md(S,G) = \sum_{j=1}^{\max(S)} (e_{jj} - a_j^2)$$
$$e_{jj} = \frac{|\{(v,u) \in E \,:\, S_v = S_u = j\}|}{|E|}$$
$$a_j = \frac{|\{(v,u) \in E \,:\, S_v = j\}|}{|E|}$$

where $\max(S)$ is the maximum value for the S_i variables, which corresponds to the number of clusters in the solution. The majority of the traditional algorithms for community detection considers this metric as the one to be optimized in order to find high quality communities. The modularity value for the graph depicted in Figure 1(b) is $Md(S,G) = 0.50$.

III. Algorithms for Community Detection

Several algorithms has been proposed for detecting communities in social networks. Community detection algorithms can be classified in two different classes: agglomerative or divisive clustering. On the one hand, agglomerative methods starts from a solution where each vertex is located in a different cluster and tries to optimize a given objective function by joining two or more communities at each step. On the other hand, divisive methods starts from a solution with all the vertices located in a single cluster, and the objective function is optimized by dividing one or more clusters in each step. Most of the algorithms are not exact procedures, since in most of the networks it is not feasible to find the optimal solution in a reasonable time, mainly due to the number of users in the network. This Section is devoted to describe the most used algorithms in the state of the art for the CDP, in order to have a framework of comparison for the algorithm presented in this work.

A. Edge-Betwenness

The idea of the Edge-Betweenness algorithm [12] relies on identifying those vertices that appears in the majority of the paths in the graph. Specifically, authors define the edge betwenness of an edge as the number of shortest paths between pairs of vertices that contains the edge under evaluation. Therefore, groups or communities are generated by removing the edge with the largest edge betwenness value in each step. This algorithm presents a complexity of $O(m^2 n)$.

B. Fast-Greedy

The Fast-Greedy algorithm [13] is focused on optimizing the modularity of the solutions generated. This agglomerative method starts from a solution where each vertex is located in a different cluster and iteratively join the two clusters that produce the solution with maximum modularity value. The optimization and data structures presented in the original work reduces the complexity of the algorithm to $O(n \cdot m \cdot \log n)$.

C. Infomap

The Infomap algorithm [14] proposed a fast stochastic and recursive search method which is based on joining neighbor vertices into the same community. The method starts with each vertex located in a different community. Then, it randomly selects a vertex and assigns it to the community that minimizes the map equation. The map equation is presented in this work and it is an efficient estimation of the optimality of a certain partition. Then, the method creates a new network where the new vertices are the communities detected until now. The algorithm stops when no changes are produced in the communities.

D. Evaluation of the previous methods

This Section is devoted to evaluate the results obtained by the different methods over an example graph that presents community structure [15]. Figure 2 illustrates the graphical results over the community detection in the graph, where each community is represented with a different color.

As it can be seen, the results are different for each algorithm. Additionally, Table II presents the results obtained by each considered algorithm over the example graph depicted in Figure 2, considering the three metrics described in Section II and the number of communities found.

First of all, we will analyze the modularity metric, since it is the most used metric in community detection optimization. The best results in modularity corresponds to the Fast Greedy (0.5284) algorithm, closely followed by Edge Betweenness (0.5245) and InfoMap (0.5231). All the values are close since the considered algorithms are focused on optimizing the modularity.

Regarding the conductance, we can see that best results are obtained again with the Fast Greedy approach, closely followed by the Edge Betweenness algorithm. In this case, both Label Propagation and Infomap present worst results in term of conductance. These results suggest that optimizing one of the metrics does not guarantee a good result in the other considered ones.

Finally, analyzing the number of communities detected, most of the algorithms detect 6 communities, which seems to be the actual number of communities in the social network. The Label Propagation algorithms prematurely stops the search, resulting in only 4 communities, while the largest number of communities, 8, is found by Infomap algorithm.

IV. Greedy Randomized Adaptive Search Procedure

Greedy Randomized Adaptive Search Procedure (GRASP) is a metaheuristic originally presented in [16] and formally defined in [17]. We refer the reader to [18] for a recent survey on this methodology. This metaheuristic can be divided into two main phases: solution generation and local improvement.

The solution generation phase iteratively adds elements to an initially empty solution until it becomes feasible. The first element is usually selected at random, acting as a seed for the procedure. The algorithm then constructs a candidate list (*CL*) with all the elements that must be included in the solution. After that, a Restricted Candidate List (*RCL*) is created with the most promising elements of the *CL* according to a predefined greedy function. Then, in each iteration, an element is selected at random from the *RCL* and added to the solution under construction, updating the *CL* and *RCL* in each step until reaching a feasible solution.

The construction phase of the GRASP algorithm presents a random part devoted to increase the diversity of the solutions generated. In particular, in the previous description, the random part relies on the selection of the next element from the *RCL*. Therefore, most of the obtained solutions are not a local optimum and can be improved by means of a local optimizer. The second phase of the GRASP algorithm is intended to find a local optimum of the solution generated, usually applying a local search method, although it can be replaced with a more complex optimizer.

The algorithm presented in this section is able to optimize any of the metrics defined in Section I. However, since the algorithm considered for the comparison are focused on optimizing the modularity, the proposed algorithm is also focused on optimizing the modularity, which has been traditionally considered as a good optimization metric.

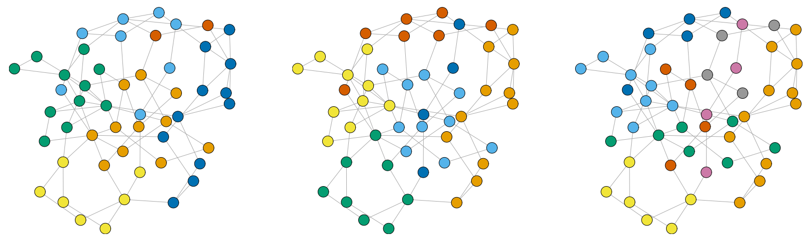
A. Constructive procedure

The constructive procedure designed for the community detection problem, named *GRASPAGG* follows an agglomerative approach, where each element is initially located in a different cluster. Then, *GRASPAGG* iteratively joins two of the most promising clusters with the objective of maximizing one of the aforementioned metrics. Algorithm 1 shows the pseudocode of the *GRASPAGG* constructive method.

Algorithm 1 $GRASPAGG(G, \alpha)$

1: $S_v \leftarrow v \ \forall v \in V$
2: $CL \leftarrow \{1, 2, \ldots n\}$
3: $continue \leftarrow$ `True`
4: **while** $continue$ **do**
5: $\quad continue \leftarrow$ `False`
6: $\quad g_{\min} \leftarrow \min_{j \in CL}(e_{jj} - a_j^2)$
7: $\quad g_{\max} \leftarrow \max_{j \in CL}(e_{jj} - a_j^2)$
8: $\quad \mu \leftarrow g_{\min} + \alpha \cdot (g_{\max} - g_{\min})$
9: $\quad RCL \leftarrow \{j \in CL \ : \ (e_{jj} - a_j^2) \geq \mu\}$
10: $\quad j_1 \leftarrow Random(RCL)$
11: $\quad Md_{best} \leftarrow Md(S, G)$
12: $\quad j_2 \leftarrow -1$
13: $\quad$ **for** $j' \in 1 \ldots CL$ **do**
14: $\qquad S' \leftarrow S$
15: $\qquad S'_v \leftarrow j_1 \ \forall S_v = j'$
16: $\qquad$ **if** $Md(S', G) > Md_{best}$ **then**
17: $\qquad\quad Md_{best} \leftarrow Md(S', G)$
18: $\qquad\quad continue \leftarrow$ `True`
19: $\qquad\quad j_2 \leftarrow j'$
20: $\qquad$ **end if**
21: $\quad$ **end for**
22: $\quad$ **if** $continue$ **then**
23: $\qquad S_v \leftarrow j_1 \ \forall S_v = j_2$
24: $\qquad CL \leftarrow CL \setminus \{j_2\}$
25: $\quad$ **end if**
26: **end while**
27: **return** S

The method starts by assigning a different cluster to each node in the graph G (step 1). Then, the *CL* is constructed with every cluster in the solution S under construction (step 2). Then, the minimum (g_{min}) and maximum ($g_{\max}$) values for the greedy function under evaluation are calculated (steps 6-7). The proposed greedy function is the modularity value of each cluster j, which is $e_{jj} - a_j^2$, as stated in Section II. Then, a threshold μ is evaluated (step 8) to construct the *RCL* with the most promising candidates in *CL* (step 9). The next steps selects the two clusters that will be merged in the current iteration, being the first cluster j_1 to be merged selected at



(a) Edge-Betweenness (b) Fast-Greedy (c) Infomap

Fig. 2: Comparison of the community detection of the described algorithms over a example graph with 50 nodes that presents community structure.

TABLE II: Evaluation of the solution generated by each algorithm over the example graph using the three considered metrics

Algorithm	Modularity	Coverage	Conductance	Number of communities
Edge Betweenness	0.5245	0.7250	0.5248	6
Fast Greedy	0.5284	0.7125	0.5306	6
Infomap	0.5231	0.6750	0.4732	8

random from the *RCL* (step 10). The second cluster j_2 is the one that maximizes the modularity of the resulting solution after merging clusters j_1 and j_2 (steps 11-21). If the method has found an improvement in the modularity after joining both clusters, a new iteration is performed, updating the incumbent solution (step 23) and the candidate list (step 24). *GRASPAGG* stops when it is not possible to join two clusters improving the modularity, returning the best solution found.

B. Local optimization

This section presents a local search procedure designed to find a local optimum for every solution constructed in the previous phase. In order to define a local search method we firstly need to define the neighborhood in which the local optimum will be found. For this problem, we consider all the solutions that can be reached from a given solution S by moving a node from one cluster to another. Specifically, after performing the move $Move(S, v, j)$, the vertex v will be located at cluster j (i.e., $S_v \leftarrow j$). Notice that if v was the last vertex in its original cluster j', then cluster j' will disappear after performing the move.

The next step for defining the local search method is the selection of the vertex to be moved to another community. For this purpose, we define a heuristic criteria based on the number of intra-cluster edges of the vertex under evaluation with respect to the total number of edges in the graph. Specifically, the local search selects the vertex v with the smallest ratio r between number of edges in the same cluster and the total number of incident edges to v. More formally,

$$r(v, S) \leftarrow \frac{|(v, u) \in E \ : \ S_v = S_u|}{|(v, w) \in E|} \quad \forall u, w \in V$$

The local search method selects, for each community, the node with the smallest value of this selection criteria among all nodes in the graph. Then, the node is moved to the community that maximizes the modularity among all the existing communities in the incumbent solution.

The proposed local search procedure follows a first improvement approach. In particular, the first improvement move found is performed, restarting the search again, opposite to performing the best available move, which is often rather time consuming. The method stops when no improvement is found after evaluating the move of a node in every community.

V. Computational Results

This Section is devoted to analyze the quality of the proposed algorithm when compared with the most popular community detection algorithms presented in Section III. Since most of the algorithms are focused on optimizing the modularity, the evaluation of the quality must be performed over a different metric. In this work we consider conductance as the evaluation metric, for testing the robustness of the methods. We additionally include the modularity value obtained by each algorithm, although it should not be considered in the evaluation of the quality of the community detection. However, we consider that it is interesting to analyze how far an algorithm is able to optimize the detection considering the modularity value. The proposed algorithm have been implemented in Java 8 and the experiments have been conducted in an Intel Core 2 Duo E7300 2.66 GHz with 4 GB RAM.

The instances used for the experiment have been extracted from the Twitter SNAP dataset[1]. Specifically, we have selected 100 instances with vertices ranging from 50 to 250, that represents the ego-network of several Twitter users (data is anonymized in the dataset).

The first experiment is devoted to tune the α parameter of the *GRASPAPP* procedure. This parameter controls the degree of randomness of the method: on the one hand, $\alpha = 0$ results in a totally random method, while $\alpha = 1$ considers

[1] https://snap.stanford.edu/data/egonets-Twitter.html

a completely greedy method. Therefore, it is interesting to test values distributed in the range 0–1 to analyze whether the best results for the CDP are obtained with a small or large percentage of randomness in the construction. In this experiment we have considered $\alpha = \{0.25, 0.50, 0.75, RND\}$, where *RND* indicates that a random value of α is selected for each construction. This experiment has been conducted over a subset of 20 representative instances in order to avoid overfitting.

Table III reports the results obtained with the different values of α. Specifically, three statistics are considered: Modularity, the average of the best modularity value obtained for each instance; Dev (%) the average deviation with respect to the best solution found in the experiment; and #Best, the number of times that an algorithm reaches that best solution.

TABLE III: Results obtained by the GRASP algorithm considering different values for α parameter

α	Modularity	Dev (%)	#Best
0.25	0.31961	1.60	9
0.50	0.32019	1.30	5
0.75	0.32063	1.08	4
RND	0.32080	1.08	6

As it can be derived from Table III, the best results are obtained with $\alpha = 0.75$. In particular, is able to obtain the best modularity values and reaches the best solution in 9 out of 20 instances. The average deviation value of 0.65% indicates that, in those instances in which it is not able to reach the best value, it remains rather close to it. Therefore, we select $\alpha = 0.75$ for the final experiment.

The final experiment is intended to compare the quality of the solutions provided by our proposal with respect to the traditional methods described in Section III.

TABLE IV: Comparison of the considered metrics over all the algorithms presented in Section III and the proposed GRASP method.

	Modularity	Conductance
EB	0.14272	0.11319
FG	0.25064	0.34494
IM	0.14216	0.33014
GRASP	0.26029	0.38206

These results show the superiority of the proposal when considering both modularity and conductance, supported by a p-value lower than 0.0001 when applying the Friedman statistical test.

VI. Conclusions

This paper has proposed a new metaheuristic method for community detection in social network based on Greedy Randomized Adaptive Search Procedure methodology. The problem is addressed by optimizing the modularity metric, which is a robust metric to evaluate the quality of a partition in a social network. The algorithm is compared with several well-known traditional algorithms for community detection using conductance as evaluation metric. The computational results show how GRASP is able to obtain better results in both metrics than the previous methods, emerging as a competitive algorithm for detecting communities in social networks.

Acknowledgment

This work has been partially founded by Ministerio de Economía y Competitividad with grant ref. TIN2015-65460-C2-2-P.

References

[1] S. N. Dorogovtsev and J. F. F. Mendes, *Evolution of Networks: From Biological Nets to the Internet and WWW.* Oxford University Press, Oxford, 2003.

[2] L. Page, S. Brin, R. Motwani, and T. Winograd, “The PageRank citation ranking: Bringing order to the web.” Stanford InfoLab, Tech. Rep., 1999.

[3] M. Gladwell, *The Tipping Point How Little Things Can Make a Big Difference*. Boston: Little Brown and Company, 2000.

[4] B. Pang and L. Lee, “Opinion mining and sentiment analysis,” *Foundations and Trends in Information Retrieval*, vol. 2, no. 1-2, pp. 1–135, 2008.

[5] A. González-Pardo, J. J. Jung, and D. Camacho, “ACO-based clustering for Ego Network analysis.” *Future Generation Comp. Syst.*, vol. 66, pp. 160–170, 2017.

[6] C. Cao, Q. Ni, and Y. Zhai, “An Improved Collaborative Filtering Recommendation Algorithm Based on Community Detection in Social Networks.” in *GECCO*, S. Silva and A. I. Esparcia-Alczar, Eds. ACM, 2015, pp. 1–8.

[7] N. Zalmout and M. Ghanem, “Multidimensional community detection in Twitter.” in *ICITST*. IEEE, 2013, pp. 83–88.

[8] D. Camacho, A. González-Pardo, A. Ortigosa, I. Gilpérez-López, and C. Urruela, “RiskTrack: A New Approach for Risk Assessment on Radicalisation Based on Social Media Data.” in *AfCAI*, ser. CEUR Workshop Proceedings, M. T. H. Ezquerro, G. J. Nalepa, and J. T. P. Mendez, Eds., vol. 1794. CEUR-WS.org, 2016.

[9] S. Emmons, S. Kobourov, M. Gallant, and K. Börner, “Analysis of Network Clustering Algorithms and Cluster Quality Metrics at Scale,” *PLoS ONE*, vol. 11, no. 7, 2016.

[10] H. Almeida, D. O. G. Neto, W. M. Jr., and M. J. Zaki, “Is There a Best Quality Metric for Graph Clusters?” in *ECML/PKDD (1)*, ser. Lecture Notes in Computer Science, D. Gunopulos, T. Hofmann, D. Malerba, and M. Vazirgiannis, Eds., vol. 6911. Springer, 2011, pp. 44–59.

[11] M. E. J. Newman and M. Girvan, “Finding and evaluating community structure in networks,” *Physical Review*, vol. E 69, no. 026113, 2004.

[12] M. Girvan and M. E. J. Newman, “Community structure in social and biological networks,” *PNAS*, vol. 99, no. 12, pp. 7821–7826, 2002.

[13] A. Clauset, M. E. J. Newman, and C. Moore, “Finding community structure in very large networks,” *Physical Review E*, vol. 70, p. 066111, 2004.

[14] M. Rosvall, D. Axelsson, and C. T. Bergstrom, “The map equation,” *The European Physical Journal Special Topics*, vol. 178, no. 1, pp. 13–23, 2009.

[15] A. Lancichinetti, S. Fortunato, and F. Radicchi, “Benchmark graphs for testing community detection algorithms,” *Phys. Rev. E*, vol. 78, no. 4, p. 046110, 2008.

[16] T. A. Feo and M. G. C. Resende, “A probabilistic heuristic for a computationally difficult set covering problem,” *Operations Research Letters*, vol. 8, no. 2, pp. 67 – 71, 1989.

[17] T. A. Feo, M. G. C. Resende, and S. H. Smith, “A Greedy Randomized Adaptive Search Procedure for Maximum Independent Set.” *Operations Research*, vol. 42, no. 5, pp. 860–878, 1994.

[18] M. G. C. Resende and C. C. Ribeiro, *GRASP: Greedy Randomized Adaptive Search Procedures*. Boston, MA: Springer US, 2014, pp. 287–312.