Generalized Louvain Method for Community Detection in Large Networks

Pasquale De Meo*, Emilio Ferrara**, Giacomo Fiumara*, Alessandro Provetti*
*Dept. of Physics, Informatics Section. **Dept. of Mathematics. University of Messina, Italy
{pdemeo, eferrara, gfiumara, ale}@unime.it

***Abstract*—In this paper we present a novel strategy to discover the community structure of (possibly, large) networks. This approach is based on the well-know concept of network modularity optimization. To do so, our algorithm exploits a novel measure of edge centrality, based on the κ-paths. This technique allows to efficiently compute a edge ranking in large networks in near linear time. Once the centrality ranking is calculated, the algorithm computes the pairwise proximity between nodes of the network. Finally, it discovers the community structure adopting a strategy inspired by the well-known state-of-the-art *Louvain method* (henceforth, LM), efficiently maximizing the network modularity. The experiments we carried out show that our algorithm outperforms other techniques and slightly improves results of the original LM, providing reliable results. Another advantage is that its adoption is naturally extended even to unweighted networks, differently with respect to the LM.**

***Keywords*-complex networks; community structure**

I. Introduction

The investigation of the community structure inside networks has acquired a great relevance during the last years, in particular in the context of Social Network Analysis (SNA). This, also because of the unpredicted success of Online Social Networks (OSNs). In fact, social phenomena such as Facebook and Twitter amongst others, glue together millions of users under a unique network whose features are a goldmine for Social Scientists. Several works are focused on the Social Network analysis of these OSNs; others describe the strategies of analysis themselves.

In this paper we focus on the possible strategies of community detection. As to date, two paradigms exist to discover the community structure of a network. The former is based on the analysis of the global features of the network, for example its topology. These approaches are characterized by high computational complexity and high quality results. The latter paradigm relies on exploiting local information, for example those acquirable by nodes and their neighborhoods. The computational cost of these techniques is lower than those exploiting global features, but the reliability decreases.

In this work, we propose a novel strategy to discover the inner community structure of a network. The main characteristics of our approach are the followings: i) it exploits global information of the network, establishing which are the edges of the network that contribute to the creation of the community structure; ii) to do so, it adopts a novel measure of edge centrality, in order to rank all the edges of the network with respect to their proclivity to propagate information through the network itself; iii) its computational cost is low, making it feasible even for large network analysis; iv) it is able to produce reliable results, even if compared with those calculated by using more complex techniques, when this is possible; in fact, because of the computational constraints, the adoption of some existing techniques is not viable when considering large networks, and their application is only limited to small case-studies.

This paper is organized as follows: in the next Section we provide some background information about the *community detection* problem. Section III introduces the main objectives of this work and describes an intuitive sketch about the novel strategy of community detection we propose. In Section IV the key concept of κ-path edge centrality is recalled, being it a novel and efficient strategy of ranking edges with respect to their centrality in the network. All the pieces are glued together in Section V. We describe our strategy to detect the community structure, inspired by the well-known state-of-the-art LM [1], which is computationally suitable even when large networks are analyzed. Experiments that have been carried out are discussed in Section VI. Finally, Section VII concludes, depicting some future directions of research.

II. Background

Several techniques to investigate the community structure of networks have been proposed in literature during last years. There exist numerous comprehensive surveys to this problem, such as [2], [3].

In its general formulation, the problem of finding communities in a network is intended as a data clustering problem. In fact, it could be solved assigning each node of the network to a cluster, in a meaningful way. Two approaches have been widely investigated, i) *spectral clustering* based techniques, and, ii) *network modularity optimization* strategies. The former relies on the optimization of the process of cutting the graph representing the given network. The latter is based on the maximization of a benefit function, called *network modularity*. We briefly recall them, separately.

The problem of minimizing the number of cuts in a given graph has been proved to be NP-hard. To do so, different approximate techniques have been proposed. An example is by using the spectral clustering [4], exploiting the eigenvectors of the Laplacian matrix of the network.

We recall that the Laplacian matrix L of a given graph has components $L_{ij} = k_i\delta(i,j) - A_{ij}$, where k_i is the degree of a node i, $\delta(i,j)$ is the Kronecker delta (that is, $\delta(i,j) = 1$ if and only if $i = j$) and A_{ij} is the adjacency matrix representing the graph connections. Another approach relies on the strategy of the ratio cut partitioning [5], [6]. This is a function that, if minimized, allows the identification of large clusters with a minimum number of outgoing interconnections. The principal issue of spectral clustering based techniques is that one has to know in advance the number and the size of communities comprised in the given network. This makes this strategy unfeasible if the purpose is to discover the unknown community structure of a network.

The strategy exploited in this paper adopts the second paradigm, the one relying on the concept of *network modularity*. It can be explained as follows: let consider a network, represented by means of a graph $G = (V, E)$, partitioned into m communities; assuming l_s the number of edges between nodes belonging to the s-th community and d_s is the sum of the degrees of the nodes in the s-th community, the network modularity Q is given by

$$Q = \sum_{s=1}^{m} \left[\frac{l_s}{|E|} - \left(\frac{d_s}{2|E|} \right)^2 \right] \quad (1)$$

Intuitively, high values of Q implies high values of l_s for each discovered community; thus, detected communities are dense within their structure and weakly coupled among each other. Equation 1 reveals a possible maximization strategy: in order to increase the value of the first term (namely, the *coverage*), the highest possible number of edges should fall in each given community, whereas the minimization of the second term is obtained by dividing the network in several communities with small total degrees.

The problem of maximizing the network modularity has been proved to be NP complete [7]. To this purpose, several heuristic strategies to maximize the network modularity Q have been proposed as to date. Probably, the most popular one is called *Girvan-Newman strategy* [8], [9]. This approach works in two steps, i) ranking edges by using the betweenness centrality as measure of importance; ii) deleting edges in order of importance, evaluating the increase of the value of Q. In fact, it is possible to maximize the network modularity deleting edges with high value of betweenness centrality, based on the intuition that they connect nodes belonging to different communities. The process iterates until a significant increase of Q is obtained. At each iteration, each connected component of S identifies a community. Unfortunately, the computational cost of this strategy is very high (i.e., $O(n^3)$, being n the number of nodes in S). This makes it unsuitable for the analysis of large networks. The largest part of its cost is given by the calculation of the betweenness centrality, that is itself very costly (even if the most efficient algorithm [10] is adopted).

Several variants of this strategy have been proposed during the years, such as the fast clustering algorithm provided by Clauset, Newman and Moore [11], that runs in $O(n \log n)$ on sparse graphs; the extremal optimization method proposed by Duch and Arenas [12], based on a fast agglomerative approach, with $O(n^2 \log n)$ time complexity; the Newman-Leicht [13] mixture model based on statistical inferences; other maximization techniques by Newman [14] based on eigenvectors and matrices.

The state-of-the-art technique is called *Louvain method* (LM) [1]. This strategy is based on local information and is well-suited for analyzing large weighted networks. It is based on the two simple steps: i) each node is assigned to a community chosen in order to maximize the network modularity Q; the gain derived from moving a node i into a community C can simply be calculated as [1]

$$\Delta Q = \frac{\sum_C + k_i^C}{2m} - \left(\frac{\sum_{\hat{C}} + k_i}{2m} \right)^2 - \left[\frac{\sum_C}{2m} - \left(\frac{\sum_{\hat{C}}}{2m} \right)^2 - \left(\frac{k_i}{2m} \right) \right] \quad (2)$$

where $\sum_C$ is the sum of the weights of the edges inside C, $\sum_{\hat{C}}$ is the sum of the weights of the edges incident to nodes in C, k_i is the sum of the weights of the edges incident to node i, k_i^C is the sum of the weights of the edges from i to nodes in C, m is the sum of the weights of all the edges in the network; ii) the second step simply makes a new network consisting of nodes that are those communities previously found. Then the process iterates until a significant improvement of the *network modularity* is obtained.

In this paper we present an efficient community detection algorithm which represents a generalization of the LM. In fact, it can be applied even on unweighted networks and, most importantly, it exploits both global and local information. To make this possible, our strategy computes the pairwise distance between nodes of the network. To do so, edges are weighted by using a global feature which represents their aptitude to propagate information through the network. The edge weighting is based on the κ-path edge centrality, a novel measure whose calculation requires a near linear computational cost [15]. Thus, the partition of the network is obtained improving the LM. Details of our strategy are explained in the following.

III. Design Goals

In this Section we briefly and informally discuss the ideas behind our strategy. First of all, we explain the principal motivations that make our approach suitable, in particular but not only, for the analysis of the community structure of Social Networks. To this purpose, we introduce a real-life example from which we infer some features of our approach.

Let consider a social network, in which users are connected among them by friendship relations. In this context, we can assume that one of the principal activities could be exchanging information. Thus, let assume that a "message"

(that, could be, for example, a *wall post* on Facebook or a *tweet* on Twitter) represents the simplest "piece" of information and that users of this network could exchange messages, by means of their connections. This means that a user could both directly send and receive information only to/from the people in her neighborhood. In fact, this assumption will be fundamental (see further), in order to define the concepts of *community* and *community structure*. Intuitively, say that a community is defined as a group of individuals in which the interconnections are denser than outside the group (in fact, this maximizes the benefit function Q).

The aim of our community detection algorithm is to identify the partitioning of the network in communities, such that the network modularity is optimal. To do so, our strategy is to rank links of the network on the basis of their aptitude of favoring the diffusion of information. In detail, the higher the ability of a node to propagate a message, the higher its centrality in the network. This is important because, as already proved by [8], [9], we could ensure that the higher the centrality of a edge, the higher the probability that it connects different communities.

Our algorithm adopts different optimizations in order to efficiently compute the link ranking. Once we define an optimized strategy for ranking links, we can compute the pairwise distances between nodes and finally the partitioning of the network, according to the LM. The evaluation of the goodness of the partitioning in communities is attained by adopting the measure of the network modularity Q.

In the next sections we shall discuss how our algorithm is able to incorporate these requirements. First of all, in Section IV, we formally provide a definition of centrality of edges in social networks based on the propagation of messages by using simple random walks of length at most κ (called, hereafter, *κ-path edge centrality*). Then, we provide a description of an efficient implementation of this algorithm, running in $O(\kappa|E|)$, where $|E|$ is the number of edges in the network. After this, in Section V we discuss the technical details of our community detection algorithm.

IV. κ-Path Edge Centrality

The concept of κ-path edge centrality has been recently defined [15] as follows:

Definition 1: (κ-path edge centrality) For each edge e of a graph $G = (V, E)$, the κ-path edge centrality $L^\kappa(e)$ of e is defined as the sum, over all possible source nodes s, of the percentage of times that a message originated from s traverses e, assuming that the message traversals are only along random simple paths of at most κ edges.

The *κ-path edge centrality* is formalized, for an arbitrary edge *e*, as follows

$$L^\kappa(e) = \sum_{s \in V} \frac{\pi_s^\kappa(e)}{\pi_s^\kappa} \tag{3}$$

where s are all the possible source nodes, $\pi_s^\kappa(e)$ is the number of κ-paths originating from s and traversing the edge e and π_s^κ is the number of κ-paths originating from s.

A. Fast κ-path Edge Centrality Algorithm

In this section we recall the functioning of the strategy adopted to efficiently compute the κ-path edge centrality. The proposed algorithm [15] is called *Weighted Edge Random Walk κ-Path Centrality* (or, shortly, WERW-Kpath). It consists of two main steps: i) node and edge weights assignment, and ii) simulation of message propagations using random simple paths of length at most κ. In the following, the two steps are discussed separately.

1) Step 1: Node and edge weights assignment

In the first stage, the algorithm assigns a weight to both nodes and edges of the graph $G = (V, E)$ representing the given network. Node weights are exploited to choose the source nodes from which the simulation of the message propagations starts; edge weights represent initial values of centrality and they are updated during the execution of the algorithm. At the end of the execution of ρ simulations, where the optimal value $\rho = |E| - 1$ has been proved in [15], edge weights are exploited for the edge ranking.

To compute node weights, we recall the notion of *local effective density* $\delta(v)$ of a node v, as follows:

Definition 2: Local effective density Given a graph $G = (V, E)$ and a node v, its local effective density $\delta(v)$ is $\delta(v) = \dfrac{|I(v)| + |O(v)|}{2|E|}$ where $I(v)$ and $O(v)$ represent, respectively, the number of ingoing and outgoing edges incident on the node v.

This value intuitively represents how much a node contributes to the overall connectivity of the graph. The higher $\delta(v)$, the better v is connected in the graph.

As for edge weights, we recall the following definition:

Definition 3: Initial edge weight Given a graph $G = (V, E)$ and an edge e, its initial edge weight $\omega(e)^0$ is $\omega(e)^0 = \dfrac{1}{|E|}$ where $|E|$ is the cardinality of E.

Intuitively, we initially manage a "budget" consisting of $|E|$ points; these points are equally divided among all the nodes; the amount of points received by each edge represents its initial rank.

2) Step 2: Simulation of message propagations

In the second step we simulate ρ simple random walks of length at most κ on the network. In detail, at each iteration, WERW-Kpath (Algorithm 1) performs these operations:

1) A node v of the graph G is selected with a probability proportional to its local effective density $\delta(v)$

$$P(v) = \frac{\delta(v)}{\phi} \tag{4}$$

where $\phi = \sum_{v \in V} \delta(v)$ is a normalization factor.

2) All the edges in G are marked as not traversed.
3) The procedure *MessagePropagation* is invoked.

Algorithm 1 WERW-Kpath(Graph $G = (V, E)$, int κ)

1: Assign each node v: $\delta(v) = \frac{|I(v)|+|O(v)|}{|E|}$
2: Assign each edge e: $\omega(e) = \frac{1}{|E|}$
3: $\rho \leftarrow |E| - 1$
4: **for** $i = 1$ to ρ **do**
5: $N \leftarrow 0$ a counter to check the length of the κ-path
6: $v \leftarrow$ a node chosen according to Eq. 4
7: MessagePropagation(v, N, κ)

Let us describe the procedure *MessagePropagation* (Algorithm 2). This procedure carries out a loop until *both* the following conditions hold true:

- The length of the path currently generated is no greater than κ. This is managed through a length counter N.
- Assuming that the walk has reached the node v_n, there must exist at least an outgoing edge from v_n which has not been already traversed. In detail, we attached a flag $T(e)$ to each edge e; $T(e) = 1$ if the edge e has already been traversed, 0 otherwise. If we call $O(v_n)$ the set of outgoing edges from v_n, it must hold that $|O(v_n)| \neq \sum_{e \in O(v_n)} T(e)$.

The former condition allows us to consider only paths up to length κ. The latter condition, instead, avoids that the message get trapped into a cycle.

If the conditions above are satisfied, the *MessagePropagation* procedure selects an edge e_n with a probability proportional to the edge weight $\omega(e_n)$, given by

$$P(e_n) = \frac{\omega(e_n)}{\gamma} \tag{5}$$

where $\gamma = \sum_{e_n \in \hat{O}(v_n)} \omega(e_n)$ is a normalization factor, being $\hat{O}(v_n) = \{e_n \in O(v_n) \mid T(e_n) = 0\}$.

Let e_n be the selected edge and let v_{n+1} be the node reached from v_n by means of e_n. The *MessagePropagation* procedure awards a bonus (equal to $\beta = \frac{1}{|E|}$) to e_n, sets $T(e_n) = 1$ and increases the counter N by 1. The message propagation activity continues from v_{n+1}.

At the end of all the processes of simulation of message propagation, each edge $e \in E$ is assigned a centrality value $L^\kappa(e)$ (in the interval $[\frac{1}{|E|}, 1]$) equal to its final weight $\omega(e)$.

The time complexity of this algorithm is $O(\kappa|E|)$. Our community detection strategy, described in the following, adopts this algorithm to weight edges of the network.

V. Community Structure Detection

In the following, we present a novel algorithm to calculate the community structure of a network. It is baptized *Fast κ-path Community Detection* (or, shortly, FKCD). The strategy

Algorithm 2 MessagePropagation(Node v, int N, int κ)

1: **while** $N < \kappa$ and $\left(|O(v)| \neq \sum_{e \in O(v)} T(e)\right)$ **do**
2: $e_{vw} \leftarrow e \in \{O(v) \mid T(e) = 0\}$, according to Eq. 5
3: $\omega(e_{vw}) \leftarrow \omega(e_{vw}) + \frac{1}{|E|}$
4: $T(e_{vw}) \leftarrow 1$; $v \leftarrow w$; $N \leftarrow N + 1$

relies on three steps: i) ranking edges by using the WERW-Kpath algorithm; ii) calculating the proximity (the inverse of the distance) between each pair of connected nodes; ii) partitioning the network into communities so to optimize the network modularity [8], according to the LM [1]. The algorithm is discussed as follows.

A. Fast κ-path Community Detection

First of all, our Fast κ-path Community Detection (henceforth, FKCD) needs a ranking criterion to compute the aptitude of all the edges to propagate information through the network. To do so, FKCD invokes the WERW-Kpath algorithm, previously described. Once all the edges have been labeled with their κ-path edge centrality, a ranking in decreasing order of centrality could be obtained. This is not fundamental, but could be useful in some applications. Similarly, before to proceed, a first *network modularity* esteem (hereafter, Q) could be calculated. This could help in order to put into evidence how Q increases during next steps. With respect to Q, we recall that its value ranges in the interval $[0, 1]$ and, the higher Q, the better the community structure of the network appears evident. The computational cost of this first step is $O(\kappa|E|)$, with κ length of the κ-paths and $|E|$ cardinality of E.

The second step consists in calculating the proximity among each pair of connected nodes. This is done by using a L_2 distance (i.e., the *Euclidean distance*) calculated as

$$r_{ij} = \sqrt{\sum_{k=1}^{n} \frac{(L^\kappa(e_{ik}) - L^\kappa(e_{kj}))^2}{d(k)}} \tag{6}$$

where $L^\kappa(e_{ik})$ (resp., $L^\kappa(e_{kj})$) is the κ-path edge centrality of the edge e_{ik} (resp., e_{kj}) and $d(k)$ is the degree of the node. We put into evidence that, even though the L_2 measure would return a distance, in our case, the higher $L^\kappa(e_{ik})$ (resp., $L^\kappa(e_{kj})$), the more the nodes are *near*, instead of distant. This important aspect leads us to consider the results of Equation 6 as the pairwise *proximities* of nodes. This step is theoretically computationally expensive, because it should require $O(|V|^2)$ iterations, but in practice, by adopting optimization techniques, its near linear cost is $O(\overline{d}(v)|V|)$, where $\overline{d}(v)$ is the mean degree of all the nodes of the network (and it is usually small in Social Networks).

The last step is the network partitioning. The main idea is inspired by the LM [1] for detecting the community structure of weighted networks in near linear time. The partitioning is an iterative process. At each iteration, two simple steps

occur: i) each node is assigned to a community chosen in order to maximize the network modularity Q; the possible increase of Q derived from eventually moving a node i into a community C is calculated according with Equation 2; ii) the second step produces a *meta-network* whose nodes are those communities previously found. The partitioning ends when no further improvements of Q can be obtained.

This reflects in splitting communities connected by edges with high proximity, which is a global feature, thus maximizing the network modularity. Its cost is $O(\gamma|V|)$, where $|V|$ is the cardinality of V and γ is the number of iterations required by the algorithm to converge (in our experience, usually, $\gamma < 5$). The FKCD is schematized in Algorithm 3.

We recall that *CalculateDistance* computes the pairwise node distance by using Equation 6, *Partition* extracts the communities according to the LM descripted above and *NetworkModularity* calculates the value of *network modularity* by using Equation 1.

The computational cost of our strategy is near linear. In fact, $O(\kappa|E|+\bar{d}(e)|V|+\gamma|V|) = O(\Gamma|E|)$, by adopting efficient graph memorization in order to minimize the execution time for the computation of Equations 1 and 6.

Algorithm 3 FKCD(Graph $G = (V, E)$, int κ)

1: WERW-Kpath(G, κ)
2: CalculateDistance(G)
3: **while** Q increases at least of ϵ (arbitrarily small) **do**
4: P = Partition(G)
5: $Q \leftarrow$ NetworkModularity(P)

VI. Experimental Results

Our experimentation has been conducted both on synthetic and real-world online social networks, whose datasets are available online. All the experiments have been carried out by using a standard Personal Computer equipped with a Intel i5 Processor with 4 GB of RAM.

A. Synthetic Networks

The method proposed to evaluate the quality of the community structure detected by using the FKCD exploits the technique presented by Lancichinetti et al. [16]. We generated the same synthetic networks reported in [16], adopting the following configuration: i) $N = 1000$ nodes; ii) the four pairs of networks identified by $(\gamma, \beta) = (2, 1), (2, 2), (3, 1), (3, 2)$, where γ represents the exponent of the *power law distribution of node degrees*, β the exponent of the *power law distribution of the community sizes*; iii) for each pair of exponents, three values of average degree $\langle k \rangle = 15, 20, 25$; iv) for each of the combinations above, we generated six networks by varying the *mixing parameter* $\mu = 0.1, 0.2, \ldots, 0.6$.

Figure 1 highlights the quality of the obtained results. The measure adopted is the *normalized mutual information* [17]. Values obtained put into evidence that our strategy performs fair good results, avoiding the well-known effect due to the resolution limit of the modularity optimization [18]. Moreover, a classification of results as in Table I (discussed later) is omitted because values of Q obtained by using FKCD and the LM in the case of these quite small synthetic networks are very similar.

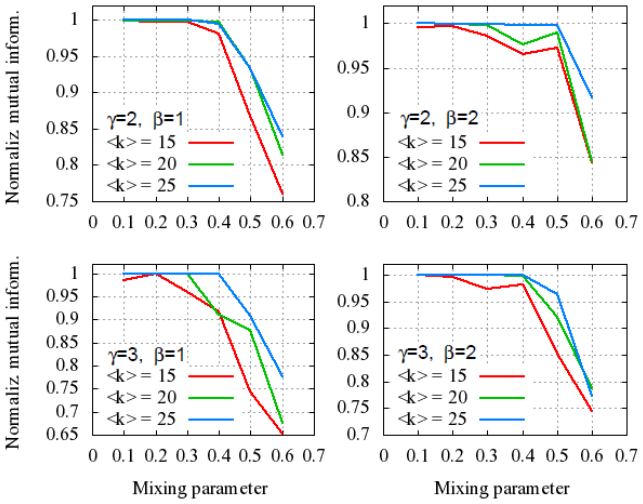

Figure 1. Test of modularity optimization using the benchmark [16], for N = 1000 nodes. The threshold value $\mu = 0.5$ represents the border beyond which communities are no longer defined in the strong sense, i.e., each node has more neighbors in its own community than in the others [19].

B. Real-world Networks

Results obtained by analyzing several real-world networks [20], [21] are summarized in Table I. This experimentation has been carried out to qualitatively analyze the performance of our strategy. Obtained results, measured by means of the *network modularity* calculated by our algorithm (FKCD), are compared against those obtained by using the original LM.

Our analysis puts into evidence the following observations: i) classic not optimized algorithms (for example Girvan-Newman [8]) are unfeasible for large network analysis; ii) results obtained by using LM are slightly higher than those obtained by using FKCD; on the other hand, LM adopts local information in order to optimize the network modularity, while our strategy exploits both local and global information; this results in (possibly) more convenient identified community structures for some applications; iii) the performance of FKCD slightly increase by using longer κ-paths; iv) both the compared efficient strategies are feasible even if analyzing large networks using standard resources of calculus (i.e., a classic personal computer); this aspect is important if we consider that there exist several Social Network Analysis tools (e.g., NodeXL[1]) that require opti-

[1]http://nodexl.codeplex.com/

mized fast algorithms to compute the community structure of networks.

Table I
DATASETS ADOPTED IN OUR EXPERIMENTATION

Network	No. nodes	No. edges	No. comm.	Fkcd$_{\kappa=5}$	Fkcd$_{\kappa=20}$	Lm
CA-GrQc	5,242	28,980	883	0.734	0.786	0.816
CA-HepTh	9,877	51,971	1,501	0.585	0.648	0.768
CA-HepPh	12,008	237,010	1,243	0.565	0.598	0.659
CA-AstroPh	18,772	396,160	1,552	0.486	0.568	0.628
CA-CondMat	23,133	186,932	2,819	0.546	0.599	0.731
Facebook	63,731	1,545,684	6,484	0.414	0.444	0.634

VII. CONCLUSIONS

The problem of discovering the community structure in large networks has been widely investigated during last years. Several efficient approaches based on local information have been proposed, and are feasible even when analyzing large networks because of their low computational cost. The main drawback of the existing techniques is that they do not consider global information about the topology of the network. In this work we presented a novel strategy that has two advantages. The former is that it exploits both local and global information. The latter is that, by using some optimization, it efficiently provides good results.

This way, our approach is able to discover the community structure in, possibly large, networks. Our experimental evaluation, carried out over both synthetic and real-world networks, proves the efficiency and the robustness of the proposed strategy.

Some future directions of research include the creation of a friendship recommender systems which suggests new possible connections to the users of a Social Network, based on the communities they belong to. Finally, we plan to design an algorithm to estimate the strength of ties between two social network users: for instance, in the case of networks like Facebook, this is equivalent to estimate the friendship degree between a pair of users.

REFERENCES

[1] V. Blondel, J. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, p. P10008, 2008.

[2] M. Porter, J. Onnela, and P. Mucha, "Communities in networks," *Notices of the American Mathematical Society*, vol. 56, no. 9, pp. 1082–1097, 2009.

[3] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3-5, pp. 75–174, 2010.

[4] A. Ng, M. Jordan, and Y. Weiss, "On Spectral Clustering: Analysis and an algorithm," in *Advances in Neural Information Processing Systems 14*, 2001.

[5] Y. Wei and C. Cheng, "Towards efficient hierarchical designs by ratio cut partitioning," in *Proc. IEEE International Conference on Computer-Aided Design*, 1989, pp. 298–301.

[6] L. Hagen and A. Kahng, "New spectral methods for ratio cut partitioning and clustering," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 11, no. 9, pp. 1074–1085, 2002.

[7] U. Brandes, D. Delling, M. Gaertler, R. Görke, M. Hoefer, Z. Nikoloski, and D. Wagner, "On finding graph clusterings with maximum modularity," in *Graph-Theoretic Concepts in Computer Science*, 2007, pp. 121–132.

[8] M. Girvan and M. Newman, "Community structure in social and biological networks," *PNAS*, vol. 99, no. 12, p. 7821, 2002.

[9] M. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E*, vol. 69, no. 2, p. 26113, 2004.

[10] U. Brandes, "A faster algorithm for betweenness centrality," *Journal of Mathematical Sociology*, vol. 25, no. 2, pp. 163–177, 2001.

[11] A. Clauset, M. Newman, and C. Moore, "Finding community structure in very large networks," *Physical Review E*, vol. 70, no. 6, p. 66111, 2004.

[12] J. Duch and A. Arenas, "Community detection in complex networks using extremal optimization," *Physical Review E*, vol. 72, no. 2, p. 27104, 2005.

[13] M. Newman and E. Leicht, "Mixture models and exploratory analysis in networks," *PNAS*, vol. 104, no. 23, p. 9564, 2007.

[14] M. Newman, "Finding community structure in networks using the eigenvectors of matrices," *Physical Review E*, vol. 74, no. 3, p. 36104, 2006.

[15] E. Ferrara, P. De Meo, and G. Fiumara, "A novel measure of edge centrality in social networks," in *under review*, 2011.

[16] A. Lancichinetti and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Physical Review E*, vol. 78, no. 4, p. 046110, 2008.

[17] L. Danon, A. Diaz-Guilera, J. Duch, and A. Arenas, "Comparing community structure identification," *Journal of Statistical Mechanics: Theory and Experiment*, p. P09008, 2005.

[18] S. Fortunato and M. Barthélemy, "Resolution limit in community detection," *PNAS*, vol. 104, no. 1, p. 36, 2007.

[19] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi, "Defining and identifying communities in networks," *PNAS*, vol. 101, no. 9, p. 2658, 2004.

[20] J. Leskovec and C. Faloutsos, "Sampling from large graphs," in *Proc. 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2006, pp. 631–636.

[21] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi, "On the evolution of user interaction in facebook," in *Proc. 2nd ACM SIGCOMM Workshop on Social Networks*, 2009.