Boosting the Detection of Modular Community Structure with Genetic Algorithms and Local Search

Clara Pizzuti
Institute for High Performance Computing and Networking
National Reasearch Council of Italy
Via P. Bucci 41C, 87036 Rende (CS), Italy
Email: pizzuti@icar.cnr.it

ABSTRACT

The discovery of modular communities to uncover the complex interconnections hidden in networks is an intensively investigated problem in recent years. Many approaches optimize a quality function, modularity, that is also a validation measure of a network partition in clusters. The paper proposes an approach, based on Genetic Algorithms, that reveals community structure in networks by optimizing modularity. The method boosts the modularity of the partition obtained by the genetic algorithm by performing a local greedy search step on this partition. Experiments on synthetic and real life networks show that the method is able to successfully reveal highly modular network structure.

Categories and Subject Descriptors

H.2.8 [**Database Managment**]: Database Applications —*Data Mining*; I.2.2 [**Artificial Intelligence**]: Automatic Programming; I.5.3 [**Computing Methodologies**]: Pattern Recognition—*Clustering*

General Terms

Algorithms

Keywords

Complex Networks, Modularity, Genetic Algorithms, Local Search

1. INTRODUCTION

The tangled relationships between objects of many artificial and natural systems can be represented by networks of nodes and edges where objects are denoted by nodes, and interactions by edges connecting nodes. Complex networks can be analyzed at different levels of granularity. The node level is the smallest scale to study. At this level the node degree can give valuable information on the role played by the objects participating in the network. More interestingly, the community or sub-graph level investigates the division of a network into groups (also called clusters or modules) having dense intra-connections, and sparse inter-connections. This partitioning is typical to many networks, thus the study of *community structure* can give important information and useful insights to understand how the structure of ties affects individuals and their relationships. In fact, members of a community interact with each other, they share information, and can have a remarkable influence on the behavior of the other objects of the community.

The problem of community detection has been receiving a lot of attention in the last few years, and many different approaches, coming from different fields such as physics, statistics, data mining, have been proposed [1, 2, 4, 21, 23, 26, 30, 13, 25, 9, 15, 32].

One of the most known approaches is that of Newman and Girvan [26], where the introduction of the $modularity$ concept fulfills a twofold objective. It gives a definition of modular structure, thus providing a validity index to measure the density of the links inside a community with respect to the links between communities, and, at the same time, a quality function that drives the search for modules.

Finding the partitioning of a network that maximizes modularity has been hypothesized to be *NP-Hard* by Clauset et al. [4], and the decision version of modularity maximization has been proved to be *NP-Complete* by Brandes et al. [3]. Thus, since its introduction, many different heuristic techniques have been proposed to optimize its value. Greedy strategies are used in [2, 4, 23], spectral division in [25, 30, 34], simulated annealing in [17, 31], extremal optimization in [7], and Genetic Algorithms in [8, 9, 15, 32, 28].

Among the above approaches, the method of Blondel et al. [2] is a fast and accurate algorithm that locally maximizes modularity by searching in the neighborhood of each node.

In this paper an algorithm, named *GAMod*, to discover communities in networks by combining *Genetic Algorithms (GAs)* [16] and local search is proposed. The approach searches for an optimal partitioning of a network by running the genetic algorithm for a fixed number of steps. The fitness function adopted is the modularity. The dense and modular communities present in the network structure are obtained at the end of the algorithm by selectively exploring the search space. At this point, a greedy local search, similar to that employed by Blondel et al. [2], is applied. For each node i, the neighbors j are considered and the gain in modularity when i is moved from its community to that of j is computed. If the gain is positive, then i is assigned to the community for which the gain is maximum. It is worth to note that Blondel et al. [2] iteratively apply this strategy until a local maximum is reached. *GAMod*, instead, finds a locally optimal solution by running the genetic algorithm, and then performs a final local greedy step to further boost modularity of the solution already obtained by the GA.

Experiments on synthetic and real life networks show that the combination of the genetic approach with local search sensibly improves modularity values.

The paper is organized as follows. The next section provides

SAC'12 March 25-29, 2012, Riva del Garda, Italy.

the necessary background to formalize the problem and defines the modularity concept. In section 3 a description of the method along with the representation adopted and the variation operators used are provided. In section 4 the results of the method on synthetic and real life data sets are presented. Finally, section 5 concludes the paper.

2. PRELIMINARIES

A network $\mathcal{N}$ can be modelled as a graph $G = (V, E)$ where V is a set of objects, called nodes or vertices, and E is a set of links, called edges, that connect two elements of V. Let $n =\mid V \mid$ be the number of nodes, and $m =\mid E \mid$ the number of edges of G. $\mathcal{C} = \{C_1, \ldots, C_k\}$ denotes a partitioning, also called clustering, of V. Each C_i $i = 1, \ldots, k$ is a group of vertices (i.e. a sub-graph) of G. A community C_i in a network is a group of nodes having high intra-cluster density of edges, and lower inter-cluster density [11].

2.1 Definition of Modularity

The $modularity$ introduced by Newman and Girvan [26] is a quality index to evaluate the goodness of a partition, widely recognized from the researcher community. The idea underlying the modularity is that a random graph has not a clustering structure, thus the edge density of a cluster should be higher than the expected density of a subgraph whose nodes are connected at random. This expected edge density depends on a chosen *null model*. Modularity can be written in the following way:

$$Q = \frac{1}{2m} \sum_{ij} (A_{ij} - P_{ij}) \delta(C_i, C_j)$$

where A is the adjacency matrix of the graph G, m is the number of edges of the graph, and P_{ij} is the expected number of edges between nodes i and j in the null model. δ is the $Kronecker$ function and yields one if i and j are in the same community, zero otherwise. When it is assumed that the random graph has the same degree distribution of the original graph, $P_{ij} = \frac{k_i k_j}{2m}$, where k_i and k_j are the degrees of nodes i and j respectively. Thus the modularity expression becomes:

$$Q = \frac{1}{2m} \sum_{ij} (A_{ij} - \frac{k_i k_j}{2m}) \delta(C_i, C_j)$$

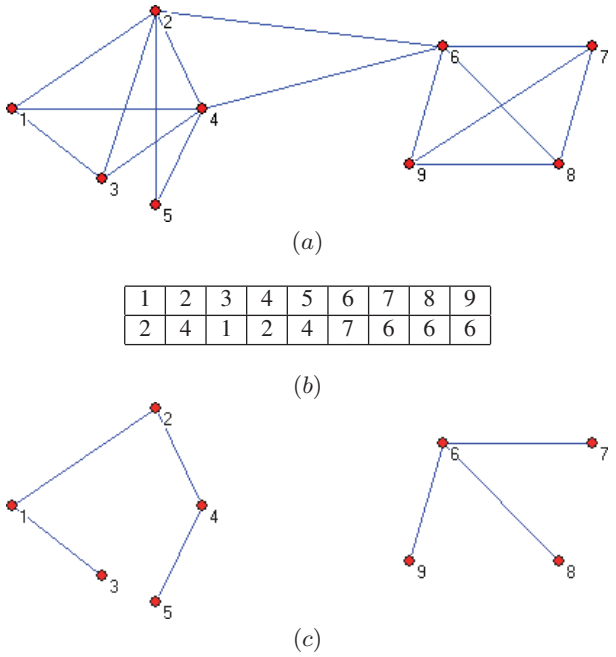
Since only the pairs of vertices belonging to the same cluster contribute to the sum, the modularity can be rewritten as

$$Q = \sum_{s=1}^{k} [\frac{l_s}{m} - (\frac{d_s}{2m})^2]$$

where k is the number of modules found inside a network, l_s is the total number of edges joining vertices inside the module s, and d_s is the sum of the degrees of the nodes of s. Thus the first term of each summand is the fraction of edges inside a community, and the second one is the expected value of the fraction of edges that would be in the network if edges fall at random without regard to the community structure. Values approaching 1 indicate strong community structure.

2.2 Approaches to Modularity Maximization

The concept of modularity has been introduced by Newman and Girvan [26] to evaluate the quality of network partitioning generated by community detection algorithms, and has become a popular objective function to optimize after Newman proposed an agglomerative hierarchical method that maximizes modularity value of a clustering [23].



1	2	3	4	5	6	7	8	9
2	4	1	2	4	7	6	6	6

Figure 1: A network modelled as a graph; (b) locus-based representation of network division in the two communities $\{1, 2, 3, 4, 5\}$ and $\{6, 7, 8, 9\}$; (c) graph-based structure of the chromosome.

One of the most efficient and efficacious methods that partitions networks based on the modularity optimization, is the algorithm of Blondel et al. [2], in the following referred as $BGLL$. The algorithm consists of two phases that are repeated iteratively until no further improvement can be obtained. At the beginning each node of the network is considered a community. Then, for each node i, all its neighbors j are considered and the gain in modularity of removing i from its community and adding it to the j community is computed. The node is placed in the community for which the gain is positive and maximum. If no community has positive gain, i remains in its original group. This first phase is repeated until no node move can improve the modularity. The second phase builds a network where the communities obtained are considered as the new nodes and a link between two communities A, B exists if there is an edge between a node belonging to A and a node belonging to B. At this point the method can be reiterated until no more changes can be done to improve modularity.

In the next section a method based on genetic algorithms enriched with a local search step, analogous to that employed by BGLL, is presented and shown to enhance the modularity of network division obtained by the algorithm.

3. ALGORITHM DESCRIPTION

In the following we give a description of the algorithm *GAMod*, the representation, and the variation operators used.

Genetic representation: Our algorithm uses the locus-based adjacency representation proposed in [27]. In this graph-based representation an individual of the population consists of n genes $g_1, \ldots, g_n$ and each gene can assume allele values j in the range

node	1	2	3	4	5	6	7	8	9
Parent1:	2	1	4	1	9	7	9	9	8
Parent2:	2	4	1	2	4	7	6	6	6
Mask:	0	1	0	1	0	0	1	1	1
Offspring:	2	4	4	2	9	7	6	6	6

Figure 2: Example of uniform crossover

$\{1, \ldots, n\}$. Genes and alleles represent nodes of the graph $G = (V, E)$ modelling a network $\mathcal{N}$, and a value j assigned to the ith gene is interpreted as a link between the nodes i and j of V. This means that in the clustering solution found i and j will be in the same cluster. A main advantage of this representation is that the number k of clusters is automatically determined by the number of connected components contained in an individual. Suppose to have the network shown in figure 1(a). It consists of nine nodes. The network can be partitioned in the two groups $C_1 = \{1, 2, 3, 4, 5\}$ and $C_2 = \{6, 7, 8, 9\}$. The chromosome shown in figure 1(b), represents the partition of the graph in these two groups. Thus, for example, in this chromosome node 1 is connected to node 2, node 2 is connected to node 4, and so on. The two connected components corresponding to this individual are depicted in figure 1(c). Each connected component thus provides a grouping of nodes, and all the components constitute a partition of the network.

Initialization: The initialization process of each individual in the population should generate a division of the network in connected groups of nodes. We experimented that connecting each node at random with one of its neighbors hampers the genetic algorithm to converge to high quality solutions. A way to overcome this drawback is to connect each node i with the neighbor j sharing the maximum number of neighbors. However, this could cause the generation of a population of almost equal individuals, thus a balance between these two opposite requirements must be obtained. To this end, after a trial and error procedure, we initialized half population with the former random strategy and the other half with the latter one.

Uniform Crossover: We used uniform crossover because it guarantees the maintenance of node connections in the child individual. Given two parents, a random binary vector is created. Uniform crossover then selects the genes where the vector is a 0 from the first parent, and the genes where the vector is a 1 from the second parent, and combines the genes to form the child. Since the child at each position i contains a value j coming from one of the two parents, the edge (i, j) exists. Figure 2 shows an example of uniform crossover. Consider node 5 that is connected to node 9 in Parent1 and to node 4 in Parent2, since the mask has a 0 in the 5th position, then the offspring will have 9 in that position.

Mutation: The mutation operator randomly changes the value j of i-th gene to one of its neighbors. This mutation guarantees the generation of a mutated child in which each node is linked only with one of its neighbors.

Given a network $\mathcal{N}$ and the graph G modelling it, *GAMod* starts with a population initialized as described above, such that each node is linked with one of its neighbors. Every individual I generates a graph structure in which each component is a connected subgraph of G. For a fixed number of generations the genetic algorithm computes the fitness function (i.e. modularity) of the current partitioning, and applies the specialized variation operators to produce the new population. The individual having the best modularity value is returned as solution. At this point a local greedy search is performed. For each node i, the neighbors j are considered and the gain in modularity when i is moved from its community to that of j is computed. If the gain is positive, then i is assigned to the community for which the gain is maximum. In the next section we show that the combination of genetic and local search provides network division having high modularity values.

4. EXPERIMENTAL RESULTS

In this section we study the effectiveness of our approach on two different benchmarks of synthetic data sets. Then we test the results obtained by *GAMod* on some real-worlds networks. We show that our genetic algorithm successfully detects the network structure and it is competitive with other approaches. The *GAMod* algorithm has been written in MATLAB 4.3 R2010a, using the Genetic Algorithms and Direct Search Toolbox 2. In order to set parameter values, a trial and error procedure has been employed and then the parameter values giving good results for the benchmark data sets have been selected. Thus we set crossover rate to 0.8, mutation rate to 0.6, elite reproduction 10% of the population size, roulette selection function. The population size is 300, the number of generations 100. For all the data sets, the statistical significance of the results produced by *GAMod* has been checked by performing a t-test at the 5% significance level. The p-values returned are, on average, below 0.05E-10, thus the significance level is very high since the probability that a community computed by *GAMod* could be obtained by chance is very low.

4.1 Partition Evaluation

In order to evaluate the quality of the clustering obtained by our method, we use the *Normalized Mutual Information* (NMI), a similarity measure proved to be reliable by Danon et al. [6], and one of the most used to test community detection algorithms on synthetic networks for which the community structure is known. The NMI is defined as follows. Given two partitions A and B of a network in communities, let C be the confusion matrix whose element C_{ij} is the number of nodes of community i of the partition A that are also in the community j of the partition B. The normalized mutual information $I(A, B)$ is defined as :

$$I(A,B) = \frac{-2\sum_{i=1}^{c_A}\sum_{j=1}^{c_B} C_{ij} log(C_{ij}N/C_{i.}C_{.j})}{\sum_{i=1}^{c_A} C_{i.} log(C_{i.}/N) + \sum_{j=1}^{c_B} C_{.j} log(C_{.j}/N)}$$

where c_A (c_B) is the number of groups in the partition A (B), $C_{i.}$ ($C_{.j}$) is the sum of the elements of C in row i (column j), and N is the number of nodes. If $A = B$, $I(A, B) = 1$. If A and B are completely different, $I(A, B) = 0$.

4.2 Synthetic data sets.

In order to check the ability of our approach to successfully detect the community structure of a network, we use two different benchmarks. The first is the benchmark proposed by Girvan and Newan in [13] (referred as GN benchmark). The network consists of 128 nodes divided into four communities of 32 nodes each. Edges are placed between vertex pairs at random but such that $z_{in} + z_{out} = 16$, where z_{in} and z_{out} are the internal and external degree of a node with respect to its community. If $z_{in} > z_{out}$ the neighbors of a node inside its group are more than the neighbors belonging to the other three groups, thus a good algorithm should discover them. The GN benchmark, as observed in [19], however, is rather simple since it is characterized by communities having all the same size and by nodes having the same expected degree. Thus, Lancichinetti et al. [20] proposed a new class of benchmarks (LFR benchmark) that extend the GN benchmark by introducing power law degree distributions and different community size. In [19] it

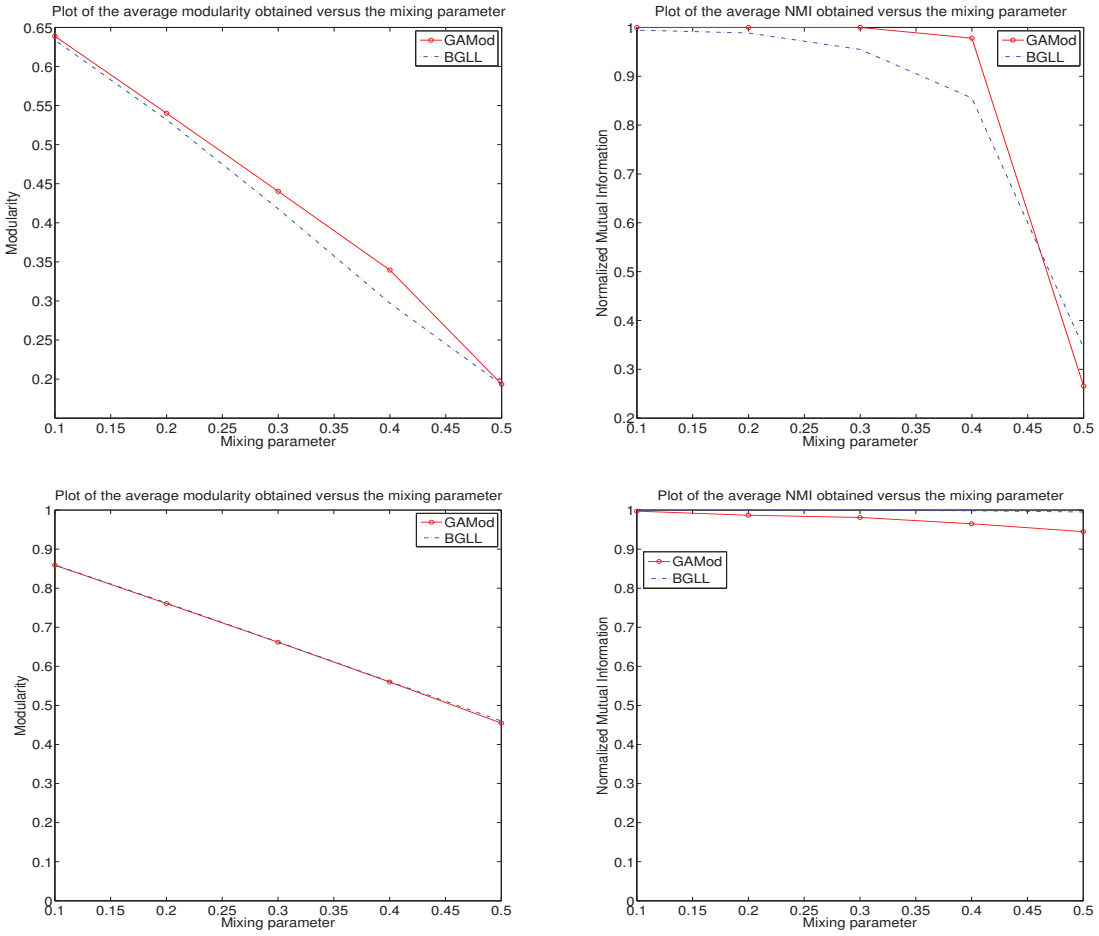


Figure 3: Modularity (left) and Normalized Mutual Information (right) obtained by *GAMod* and BGLL on the synthetic GN (top) and LFR (bottom) benchmarks when the mixing parameter μ varies from 0.1 to 0.5.

has been experimented that many community detection algorithms perform well on the GN benchmark, but give poor results on the LFR benchmark. In particular, they showed that the Blondel et al. [2] method outperforms other approaches on the LFR networks.

Both GN and LFR benchmarks are characterized by the *mixing parameter* $\mu = \frac{z_{out}}{z_{in}+z_{out}}$ that gives the ratio between the external degree of a node and the total degree of the node. When $\mu < 0.5$ the communities are well defined, thus a good algorithm should discover them. We generated 100 different networks for values of μ ranging from 0.1 to 0.5 for both GN and LFR benchmarks. Then we computed the *Normalized Mutual Information* to measure the similarity between the true partitions and the detected ones, and the modularity to evaluate the goodness of the partitioning obtained. As regards the LFR benchmark, the network generated [1] is the same of that used in [19] for unweighted and undirected graphs. It is constituted by 1000 nodes, average node degree 20, maximum node degree 50, exponent of degree distribution -2, community size distribution -1.

Figure 3 shows the values of modularity (top left) obtained by *GAMod* and BGLL methods on the GN benchmark when the mixing parameter varies from 0.1 (clear community structure) to 0.5 (not well defined community structure), and the corresponding normalized mutual information (figure top right). The figures point out that on this network for $\mu \leq 4$ *GAMod* reaches average higher values of both modularity and NMI. For $\mu = 5$, while the modularity is almost the same, the NMI value obtained by $BGLL$ is slightly higher (0.343 w.r.t. 0.265). However, among all the executions, the maximum modularity obtained by both has been 0.468 for $BGLL$ and 0.456 for *GAMod*.

Figure 3 shows the same experiment on the LFR benchmark (bottom figures). Also in this case the modularity values computed by the two methods are almost the same, while the NMI obtained by $BGLL$ is a little bit higher than that obtained by *GAMod*. These experiments point out that the genetic approach is able to uncover community structure and it is comparable with the Blondel et al. algorithm. In the next section we test the two methods on real-life networks and we show the very good performance of *GAMod* on them.

4.3 Modularity Comparison on Real-Life Networks

In this section we compare the modularity values obtained by *GAMod* and $BGLL$ on a set of ten networks of different size. Then the modularity found on some of these networks is matched against the results reported by Pujol et al. in [30]. In all the cases we show

[1]The software to generate the LFR benchmarks can be found at http://sites.google.com/site/andrealancichinetti/software

Table 1: Comparison between modularity values and number of communities obtained by *GAMod* and Blondel et al. [2].

			GAMod				BGLL	
Network	**nodes**	**edges**	**best**	**avg**	**std dev**	**NC**	**Mod**	**NC**
Karate	34	78	0.4198	0.4198	0	4	0.4020	3
Dolphins	62	159	0.5285	0.5270	0.589e-3	5	0.4952	10
Krebs	105	440	0.5256	0.5251	0.84e-3	5	0.5156	8
Adjnoun	112	425	0.2894	0.2664	0.118e-1	6	0.2364	25
Football	115	613	0.6046	0.6040	0.11e-2	10	0.6010	12
Jazz	198	2742	0.4425	0.4373	0.583e-2	3	0.4228	7
C. Elegans	453	4596	0.4160	0.4070	0.646e-2	10	0.3731	63
Scientometrics	2678	10368	0.5818	0.5713	0.5135e-2	84	0.5483	24
Directors Board	1130	6647	0.9005	0.8965	0.318e-2	72	0.9084	60
Erdös	6927	11850	0.6817	0.6760	0.29e-2	127	0.6845	49

Table 2: Comparison between best modularity values and number of communities obtained by *GAMod*, Pujol et al. [30], Dutch and Arenas algorithm [7], and the Newman's fast algorithm [23]

	GAMod		PBD		DA		Newman	
Network	**Mod**	**NC**	**Mod**	**NC**	**Mod**	**NC**	**Mod**	**NC**
Karate	**0.4198**	4	0.3937	4	0.4176	4	0.3807	3
C. Elegans	0.4160	10	0.4164	7	**0.4376**	10	0.40	10
Scientometrics	0.5818	84	0.5629	10	**0.6042**	19	0.5555	24
Directors Board	**0.9005**	72	0.8273	16	0.8113	27	0.8046	21
Erdös	**0.6817**	127	0.6817	20	0.6520	88	0.6723	57

that *GAMod* gets high modularity values. The networks used are the Zackary Karate Club [35], the Bottlenose Dolphins network compiled by Lusseau [22], the network of political books compiled by V. Krebs (unpublished http://www.orgnet.com/), adjacency network of common adjectives and nouns in the novel David Copperfield by Charles Dickens [24], the American College Football network [13], the network of Jazz musicians [14], the Metabolic network C. Elegans [18], the citation network Scientometrics [5], the affiliation network among the Spanish top directors board [12], and the scientific collaboration networks of $Erdös$ [29].
Table 1 lists the ten networks with the number of nodes and edges, along with the modularity and number of communities achieved by *GAMod* and $BGLL$. For the former algorithm both the best and average modularity, together with the standard deviation are reported. $BGLL$ obtained the same values for all the executions. The table points out that *GAMod* achieves almost the same results of $BGLL$ on the $Erdös$ and directors networks, and higher modularity, both best and average values, for all the other networks considered.

Table 2 compares *GAMod* with other three methods, the Pujol et al. algorithm [30], referred as PDB, based on random walkers, the Dutch and Arenas extremal optimization algorithm [7], referred as DA, and the Newman's fast algorithm [23]. In such a case the DA method achieves a higher modularity than that obtained by *GAMod* on the C. Elegans and Scientometrics networks, while for the other networks *GAMod* has the best performance.

It is worth to note that the number of communities found by *GAMod* (the number is relative to the best modularity) is lower than that obtained by $BGLL$, but almost the same of that found by PBD, DA, and Newman on the Karate and C. Elegans network, and higher for the other networks. Pujol et al. [30] argues that finding a small number of groups simplifies the analysis of the obtained results. However, recently, it has been proved that the optimization of modularity has a resolution limit that depends on the total size of the network and the interconnections of the modules [10]. This implies that modules below an intrinsic scale, even if tightly connected, cannot be found. Table 2 highlights that, though *GAMod* achieves higher modularity values on the larger networks (Scientometrics, Directors Board, and Erdös), the number of communities is higher than the other three approaches, thus the genetic algorithm seems to be able to partially elude the resolution limit problem.

5. CONCLUSIONS

The paper presented a genetic algorithm enriched with a local search step able to uncover highly modular community structure. The method has been shown to boost modularity of the clusters obtained and to outperform state-of-the-art approaches. One of the main criticisms in using genetic algorithms, compared to traditional optimization algorithms, is the high execution time required to generate a solution. The major limitation of evolutionary algorithms is, in fact, the repeated fitness function evaluation that, for complex problems could often be prohibitive. The problem is exacerbated when large populations of individuals are used. In our approach fitness evaluation is rather simple, thus the main problem comes from the network size. However Genetic Algorithms are naturally suited to be implemented on parallel architectures [33], and an implementation of *GAMod* on a parallel machine could be realized in order to deal with very large networks, and make the approach proposed competitive in terms of computation time with respect to other faster methods that discover communities.

The current implementation of *GAMod* uses the Genetic Algorithm and Direct Search Toolbox 2 available on Matlab. Since Matlab provides also the Parallel Computing Toolbox to perform parallel computations on clusters of computers, and a built-in parallel implementation of Genetic Algorithms, we executed the Erdös network on a E4 cluster of 12 nodes, each having a 2 CPU Quad Core Intel Xeon E5520 2,26GHz, 12GB of RAM, using a variable number of cores to test the efficiency of the method. We experimented that, as the number of cores augments from one to 32, the execution time decreases linearly. Thus, having at disposal enough computing resources, *GAMod* could obtain a good speed up, thus sensibly reducing its execution times.

6. REFERENCES

[1] A. Arenas and A. Diaz-Guilera. Synchronization and modularity in complex networks. *European Physical Journal ST*, 143:19–25, 2007.

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

[3] Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert GŽrke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. : On modularity clustering. . *IEEE Trans. Knowledge and Data Engineering*, 20(2):172–188, 2008.

[4] A. Clauset, M. E. J. Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical Review E*, 70:066111, 2004.

[5] Garfields's collection of citation networks. http://www.garfield.library.upenn.edu/histcomp/.

[6] L. Danon, A. Díaz-Guilera, J. Duch, and A. Arenas. Comparing community structure identification. *Journal of Statistical Mechanics*, P09008, 2005.

[7] Jordi Duch and Alex Arenas. Community identification using extremal optimization. *Physics Review E*, 72(027104), 2005.

[8] Zhidan Feng, Xiaowei Xu, Nurcan Yuruk, and Thomas A. J. Schweiger. A novel similarity-based modularity function for graph partitioning. In *Proc. 9th Int. Conf. on Data Warehousing and Knowledge Discovery(DaWaK'07)*, pages 385–396, 2007.

[9] A. Firat, S. Chatterjee, and M. Yilmaz. Genetic clustering of social networks using random walk. *Computational Statistics and Data Analysis*, 51(12):6285–6294, 2007.

[10] S. Fortunato and M. Barthélemy. Resolution limit in community detection. *Proc. National Academy of Science, USA*, 104(36), 2007.

[11] Santo Fortunato. Community detection in graphs. *Phisics Reports*, 486:75–174, 2010.

[12] Data from the project "Small Worlds of Corporate Networks". IESE Business School, University of Navarra.

[13] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. In *Proc. National. Academy of Science. USA 99*, pages 7821–7826, 2002.

[14] Pablo M. Gleiser and Leon Danon. Community structure in jazz. *Advances in Complex Systems*, 6(4):565–573, 2003.

[15] A. Gog, D. Dumitrescu, and B. Hirsbrunner. Community detection in complex networks using collaborative evolutionary algorithms. In *9th European Conference on Artificial Life (ECAL'07)*, pages 886–894, 2007.

[16] D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Publishing, 1989.

[17] Roger Guimerá, Marta Sales-Pardo, and Luís A. Nunes Amaral. Modularity from fluctuations in random graphs and complex networks. *Physical Review E*, 70(2):025101, 2004.

[18] H. Jeong, B. Tombor, R. Albert, Z. Oltvai, and A.-L. Barabási. The large-scale organization of metabolic networks. *Nature*, 470:651–655, 2000.

[19] Andrea Lancichinetti and Santo Fortunato. Community detection algorithms: a comparative analysis. *Physical Review E*, 80(056117), 2009.

[20] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E*, 78(046110), 2008.

[21] S. Lozano, J. Duch, and A. Arenas. Analysis of large social datasets by community detection. *European Physical Journal ST*, 143:257–259, 2007.

[22] D. Lusseau. The emergent properties of dolphin social network. *Biology Letters, Proc. R. Soc. London B (suppl.)*, 2003.

[23] M. E. J. Newman. Fast algorithm for detecting community structure in networks. *Physical Review*, E69:066133, 2004.

[24] M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review*, E 74:036104, 2006.

[25] M. E. J. Newman. Modularity and community structure in networks. *Proc. Natl. Acad. Sci. USA 103*, pages 8577–8582, 2006.

[26] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review*, E69:026113, 2004.

[27] Y.J. Park and M.S. Song. A genetic algorithm for clustering problems. In *Proc. of 3rd Annual Conference on Genetic Algorithms*, pages 2–9, 1989.

[28] C. Pizzuti. GA-NET: a genetic algorithm for community detection in social networks. In *Proc. of the 10th Intenational Conference on Parallel Problem Solving from Nature (PPSN 2008)*, pages 1081–1090, 2008.

[29] Erdös Number Project. http://www.oakland.edu/enp/thedata/.

[30] Josep M. Pujol, Javier Béjar, and Jordi Delgado. Clustering algorithm for determining community structure in large networks. *Physics Review*, E 74(016107), 2006.

[31] Jorg Reichardt and Stefan Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74:016110, 2006.

[32] M. Tasgin and A. Bingol. Communities detection in complex networks using genetic algorithms. In *Proc. of the European Conference on Complex Systems (ECSS'06)*, 2006.

[33] Marco Tomassini. *Parallel and Distributed Evolutionary Algorithms: A Review*. in Evolutionary Algorithms in Engineering and Computer Science, J. Wiley and Sons, Chichester et al. eds., 1999.

[34] Scott White and Padhraic Smyth. A spectral clustering approach to finding communities in graphs. In *Proc. of the 5th SIAM Conference on Data Mining*, pages 274–285, 2005.

[35] W.W Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33:452–473, 1977.