A Memetic Algorithm for Community Detection in Complex Networks

Olivier Gach[1,2] and Jin-Kao Hao[2]

[1] LIUM & IUT, Université du Maine, Av. O. Messiaen, 72085 Le Mans, France
olivier.gach@univ-lemans.fr
[2] LERIA, Université d'Angers, 2 Bd Lavoisier, 49045 Angers Cedex 01, France
hao@info.univ-angers.fr

Abstract. Community detection is an important issue in the field of complex networks. Modularity is probably the most popular partition-based measure for community detection of networks represented as graphs. We present in this article a hybrid algorithm mixing a genetic approach with a dedicated crossover operator and a multi-level local optimization procedure. Experimental evaluations on a set of 11 well-known benchmark graphs show that the proposed algorithm attains easily all the current best solutions and even improves 6 of them in terms of maximum modularity.

Keywords: heuristic, community detection, complex networks, graph partitioning, modularity, combinatorial optimization

1 Introduction

Complex networks are a graph-based model which is very useful to represent connections and interactions of the underlying entities in a real networked system [19]. A vertex of the complex network represents an object of the real system while an edge symbolizes an interaction between two objects. A typical example is social network where each vertex corresponds to a particular member of the network while the edges incident to the vertex represent the relationships between this member and other members. Other prominent complex networks include biological networks, citation networks, and the World Wide Web.

Complex networks typically display non-trivial topological features and special patterns which characterize its connectivity and impact the dynamics of processes applied to the network [17]. Discovering these particular features and patterns helps understand the dynamics of the networks and represents a real challenge for research [6].

In particular, complex networks may contain specific groups of highly interconnected vertices which are loosely associated with other groups. Such a group is commonly called community, cluster or still module [19] and all the communities of a network form a clustering. In terms of graph theory, a clustering can be defined as a partition of the vertices of the underlying graph into disjoint subsets, each subset representing a community. A community is typically characterized by two basic factors: intra-cluster density and inter-cluster density. Intuitively, a community is a cohesive group of vertices that are connected more "densely" to each other than to the vertices in other communities. To quantify the quality of a given community and more generally a clustering, *modularity* is certainly the most popular measure [18]. Under this quality measure, the problem of community detection becomes a pure combinatorial optimization problem. Formally, the modularity measure can be stated as follows.

Given a weighted graph $G = (V, E, w)$ where w is a weighting function, i.e., $w : V \times V \longmapsto \mathbb{R}$ such that for all $\{u, v\} \in E, w(\{u, v\}) \neq 0$, and for all $\{u, v\} \notin E, w(\{u, v\}) = 0$. Let $C \subseteq V$ and $C' \subseteq V$ be two vertex subsets, $W(C, C')$ the weight sum of the edges linking C and C', i.e., $W(C, C') = \sum_{u \in C, v \in C'} w(\{u, v\})$ (in this formula, each edge is counted twice). The modularity of a clustering with K communities $I = \{C_1, C_2, ..., C_K\}$ $(\forall i \in \{1, 2, ..., K\}, C_i \subset V$ and $C_i \neq \varnothing$; $\cup_{i=1}^{K} C_i = V$; $\forall i, j \in \{1, 2, ..., K\}, C_i \cap C_j = \varnothing)$ is given by:

$$Q(I) = \sum_{i=1}^{K} \left[\frac{W(C_i, C_i)}{W(V, V)} - \left(\frac{d_i}{W(V, V)} \right)^2 \right] \qquad (1)$$

where d_i is the sum of the degrees of the vertices of community C_i, i.e., $d_i = \sum_{v \in C_i} deg(v)$ with $deg(v)$ being the degree of vertex v.

It is easy to show that Q takes values in the interval [-0.5,1]. A clustering with a small Q value close to the lower bound implies the absence of real communities. A large Q value close to 1 indicates a good clustering containing highly cohesive communities. In particular the modularity of a trivial clustering with a single cluster has a value of 0.

Given the modularity measure Q, the community detection problem aims to find, among the space of all possible clusterings (partitions) of a given graph, a particular clustering with the maximal modularity Q. This is thus a highly combinatorial optimization problem and known to be NP-hard [3]. Consequently, heuristic algorithms become a natural choice to handle this problem. The heuristic algorithms proposed recently in the literature for community detection with the modularity measure belong to three general solution approaches: fast greedy agglomeration like [4], local search [22,14] and hybrid algorithms like [1,13] as some examples.

In this paper, we introduce a memetic algorithm for community detection (MA-COM). MA-COM combines a dedicated crossover operator and a multilevel optimization procedure. To maintain population diversity, MA-COM additionally uses a quality-and-distance based population updating strategy. To assess the effectiveness of the proposed MA-COM algorithm, we present computational results on a set of 11 well-known complex networks and show solutions with an improved Q value for 6 cases with respect to the best-known values reported in the literature.

2 Hybrid Evolutionary algorithm

2.1 Main scheme

Memetic algorithms are known to be an effective approach in solving a number of hard combinatorial optimization problems [16]. A memetic algorithm based on genetic local search hybrid repeatedly alternates between a recombination (or crossover) operator and a local optimization operator. The recombination operator generates new solutions which are hopefully located in new promising regions in the search space while the local optimization operator searches around the newly generated solutions in order to discover solutions of good quality.

The general scheme of our MA-COM algorithm for community detection is summarized in Algorithm 1. Basically, MA-COM begins with an initial population of solutions (line 1, Section 2.2) and then repeats an iterative process for a number of times (generations) (lines 3–11). At each generation, two solutions are randomly selected to serve as parents (line 4). The recombination operator is applied to the parents to generate a new offspring solution which is further improved by the local optimization procedure (lines 5–6, see Section 2.3). Finally, we apply a quality-and-distance based rule to decide whether the improved offspring solution can be inserted into the population (line 10, Section 2.4). The solution with the highest modularity discovered during the search is always recorded (line 7-8). The whole algorithm stops if during g consecutive generations, the modularity improvement is inferior to a given threshold ϵ. In the following subsections, we give more details on the components of our algorithm.

Algorithm 1 Pseudo-code of memetic algorithm for community detection

```
Require: Graph $G = (V, E)$.
Ensure: A clustering $I^*$ of $G$ with a maximal modularity.
 1: $P = \{I^1, I^2, ..., I^p\} \leftarrow$ Initialize_Population()   /* Sect. 2.2*/
 2: $I^* = \arg\max_{I \in P}\{Q(I)\}$   /* Record the best clustering found so far */
 3: repeat
 4:     $(I^i, I^j) \leftarrow$ Choose_Parents($P$)
 5:     $I \leftarrow$ Recombine_Parents($I^i, I^j$)   /* Sect. 2.3 */
 6:     $I \leftarrow$ Improve($I$)   /* Sect. 2.2 and 2.3 */
 7:     if $Q(I) > Q(I^*)$ then
 8:         $I^* \leftarrow I$
 9:     end if
10:     $P \leftarrow$ Update_Population($I, P$)   /* Sect. 2.4 */
11: until end_criterion
```

2.2 Initial Population

To generate the initial population P, we employ the multi-level algorithm due to Blondel et al. (named BGLL) [1] which uses the vertex mover (VM) heuristic [22] as its refinement procedure. BGLL begins with a coarsening phase to create a hierarchy of simplified graphs organized into multi-levels and for each step of the coarsening phase, the vertex mover (VM) heuristic [22] is iteratively applied to the clustering of the current graph to obtained a local optimal clustering. BGLL stops when the clustering of the graph cannot be further improved by VM.

More precisely, from the initial graph G_0 (the lowest level) where each vertex forms a community, the VM heuristic is iteratively applied to all the vertices of G_0 until no vertex can be moved to improve the modularity of the clustering C of graph G_0. Based on this local optimal clustering C, graph G_0 is transformed into a new and higher-level graph G_1 where each vertex represents a community of C and two vertices of G_1 are linked by an edge if they represent two neighboring communities in C. Now the VM heuristic is applied to G_1 to create another clustering which is used to transform G_1 to a new graph G_2 of higher level. This coarsening phase stops when the last graph cannot be further improved by the VM heuristic.

At this point, a second phase (uncoarsening) unfolds the hierarchy of graphs starting from the highest level. At each uncoarsening step, the communities represented by the vertices of the current graph are recovered. The uncoarsening phase stops when the lowest level is reached to recover the initial graph G_0. The corresponding clustering of G_0 constitutes an individual of the initial population of our memetic algorithm.

Experiments show that this initialization procedure is able to provide the memetic algorithm with diversified initial solutions of good quality.

2.3 A priority-based crossover operator

Crossover is a key element for the effectiveness of the memetic approach. We develop a crossover operator which is dedicated to the clustering problem, named priority-based crossover operator. The key idea of this operator is to take communities as genetic material of solution and try to preserve some communities from the parents. The proposed crossover operator operates on parents that are sufficiently distanced in terms of community structure.

Precisely, the crossover operator uses two parent clusterings (I^1, I^2) (which are selected randomly from the population) and a priority vector p to generate one offspring clustering. Let s and r be respectively the number of communities of clusterings I^1 and I^2. The vector p, indexed from 1 to $s + r$, is defined by a random permutation of $\{1, 2, ..., s+r\}$. The indices between 1 and s of p denotes the communities of one parent and those between $s+1$ and $s+r$ the communities of the other parent. Thus each community of the parents is designated by a unique number from 1 to $s + r$. For each community $C_i, i \in \{1, 2, ..., s + r\}$, the corresponding value in p (i.e., $p[i]$) identifies the priority of C_i for crossover. By convention, a smaller p value indicates a higher priority for the community and vise versa.

The crossover procedure generates an new offspring clustering I^o as follows. We go through one by one all the communities by following the priority order given by the vector p. We begin by selecting the community C with the highest priority according to p and transfer all the vertices of the community to form a community of the offspring I^o. We then pick the community C' with the second highest priority according to p, remove the vertices already in I^o and use the remaining vertices of C' to form a new community of I^o (empty community is discarded). We repeat this process until the community with the lowest priority is handled. Finally, the communities of I^o are re-labeled from one to the number of communities contained in the offspring.

Figure 1 illustrates the crossover procedure applied to a small graph. Among the 7 communities of the two parents, the one with the highest priority 1 (labeled 5 in parent 2 with vertices {1,2,8,10,13,17}) is transfered to the offspring. The second selected community is the one labeled 2 from parent 1 (i.e., {3,7,9,13,16}). After removing the vertex 13 already present in the offspring, we use {3,7,9,16} to form another community of the offspring. The next selected community is the one labeled 1 from parent 1 ({1,2,8,10,17}), removing the shared vertices leads to an empty community which is discarded. This process continues until all the 7 communities are examined. The resulting offspring is composed of 5 communities originating from both parents.

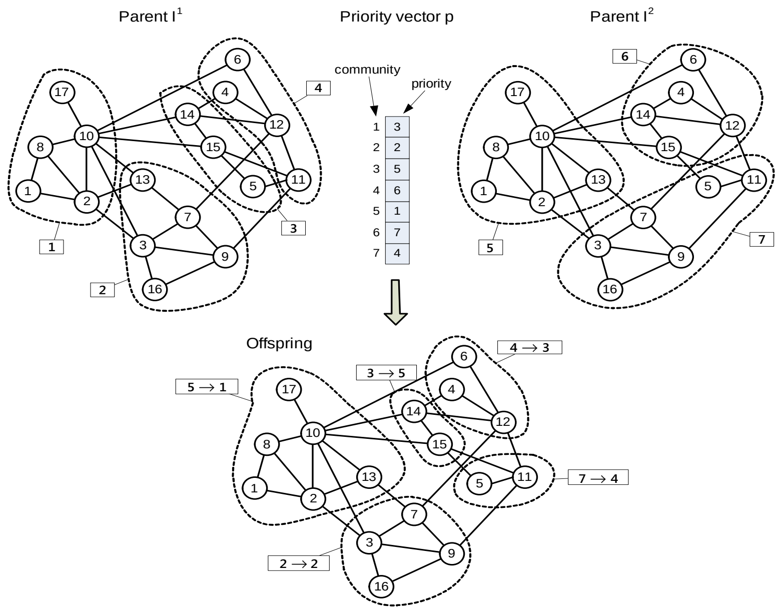

Fig. 1. Illustration of the crossover operator. Five new communities in the offspring are created from seven communities of two parents.

This crossover operator leads generally to an offspring clustering with more communities than in the parents, decreasing thus the modularity objective. To improve the quality of the offspring, we apply the BGLL algorithm described in Section 2.2 by taking the offspring as its initial solution. The improved offspring will then be considered for inclusion in the population according to a quality-and-distance strategy explained in the next section.

The time complexity of the crossover operator is $O(n)$. With an appropriate data structure, it can be implemented in one pass of the vertices of the graph.

Finally, we notice the the priority associated to each community can be defined by considering other factors like its modularity and size. Due to space limit, we do not explore these possibilities in this paper. Yet, as shown in the section of experimental evaluation, our memetic algorithm equipped with the crossover operator using random priorities works well for the set of the test graphs.

2.4 Population updating strategy

Population diversity is another critical issue in a memetic algorithm to avoid premature convergence. In our case, this is particular true due to the small size of the population used (typically several tens of solutions). For this reason, we employ a population updating strategy which considers not only the quality of the offspring, but also its distance to the solutions of the population.

Distance function Let $X = \{X_1, X_2 ... X_K\}$ and $Y = \{Y_1, Y_2 ... Y_{K'}\}$ be two clusterings of graph $G = (V, E)$. For an edge $e = \{u, v\} \in E$ and a community C of X or Y, we use $e \in C$ to state the fact that the vertices u and v of e are in the same community. Then we use the Rand Index [21] to define our distance d between X and Y as follows:

$$d(X,Y) = \frac{\sum_{e \in E} d_e(X,Y)}{m} \tag{2}$$

where $d_e(X,Y)$ of edge $e = \{u,v\}$ is defined by:

$$d_e(X,Y) = \begin{cases} 0 \text{ if } \exists X_i \in X,\ \exists Y_j \in Y \text{ s.t. } e \in X_i \text{ and } e \in Y_j \text{ OR} \\ \quad \text{if } \forall X_i \in X,\ \neg(e \in X_i) \text{ and } \forall Y_i \in Y, \neg(e \in Y_j) \\ 1 \text{ otherwise.} \end{cases} \tag{3}$$

We can show that d (called Edge Rand Index - ERI) satisfies the conditions of a mathematical distance and takes values in [0,1]. Intuitively, this distance measures the edge disagreements between two clusterings.

Updating procedure Let P and I^o be the current population and the offspring to be considered for inclusion in P respectively. Let $I \in P$ be the closest clustering to I^o according to the above distance. Let δ_{min} is a fixed distance threshold. Then we apply the following replacement strategy: if $d(I^o, I) > \delta_{min}$ and $Q(I^o) \geq Q(I)$, then I^o replaces I in P; otherwise, I^o replaces the worst clustering (with the smallest modularity) in P.

By taking into account both quality and distance, this updating strategy reinforces the population diversity when the search progresses.

3 Computational results

3.1 Experimental setup

This section is dedicated to a performance assessment of our MA-COM algorithm which is coded in Pascal. We carry out extensive experiments on a set of 11 networks (with 34 to 27519 vertices) commonly used for community detection (Table 1). Directed graphs are transformed into undirected graphs and loops are removed. Our algorithm also takes into account weighted graphs (*Cond-mat2003*). We run the program 20 times on each graph and report the maximal modularity, the average modularity and the average computing time, based on a PC equipped with a Pentium Core i7 870 of 2.93 GHz and of 8 GB of RAM. The algorithm stops after 500 consecutive generations without an improvement of modularity greater than 10^{-4}. The values for the other parameters are the following: population size (30), distance threshold δ_{min} used for population management (0.01). These same values are used to report all the results of this section, though better results would be possible by fine-tuning some parameters. In Section 3.2, we show our results in terms of the modularity criterion while in Section 3.3 we analyze some structural features of the solutions found.

3.2 Results in terms of modularity

Table 1 shows the results of the proposed memetic algorithm (MA-COM) compared to the current best-known results (BKR) ever reported in the literature in terms of the modularity values. We also include the results of the BGLL algorithm which is used to generate the initial population of our memetic algorithm. From Table 1, we observe that the proposed MA-COM algorithm obtains clusterings of equal or greater modularity for all the tested graphs. In particular, for the 6 largest graphs (from *C. elegans* to the last network), MA-COM improves the current best-known results by finding solutions with a larger modularity. For the first 5 graphs which are also the smallest ones (with no more 200 vertices and 3000 edges), BGLL alone suffices to attain the current best-known modularity values during the population initialization phase.

We also observe that the average modularity of our MA-COM algorithm is very closed to the maximum and, for all the graphs, is always equal to or better than the best-known result. This shows that MA-COM is quite stable, despite of its stochastic nature. The computing time is growing more than linearly with respect to the number of edges m. Experimental statistics show that the time complexity could be approximated by $O(m^{\alpha})$ with $\alpha \approx 1.3$.

3.3 Structural changes in clusterings

In the last section, we show that MA-COM improves the solutions of the BGLL algorithm in terms of modularity. Now we turn our attention to structural transformations of solutions achieved by MA-COM from solutions given by BGLL. For this purpose, we consider, for each of the 11 graphs and each of the 20 runs

Table 1. Results on 20 runs of the proposed MA-COM algorithm on 11 commonly used real graphs (sources in brackets). The BKR column shows the best known result with its sources in brackets. The other columns give the average and maximum modularity of the best solutions in the initial population (BGLL) and the final population of MA-COM. The number of communities of the best solution is indicated between parenthesis. Improved results are highlighted in bold.

Graph	BKR	BGLL [1]		MA-COM		
		Avg Q	Max Q (K)	Avg Q	Max Q (K)	Time(s)
Karate Club [23]	0.4198 [13,20,14]	0.4198	0.4198 (4)	0.4198	*0.4198 (4)*	0.3
Dolphins [15]	0.529 [13]	0.5281	0.5286 (5)	0.5286	*0.5286 (5)*	0.5
Political Books [12]	0.527[13]	0.5273	0.5273 (5)	0.5273	*0.5273 (5)*	1.0
College Football [7]	0.605 [13]	0.6046	0.6046 (10)	0.6046	*0.6046 (10)*	1.4
Jazz [8]	0.4452 [14]	0.4452	0.4452 (4)	0.4452	*0.4452 (4)*	5.2
C. elegans [5]	0.452 [13]	0.4457	0.4497 (11)	0.4531	**0.4533 (10)**	8.3
E-mail [10]	0.582 [13]	0.5748	0.5772 (10)	0.5828	**0.5829 (10)**	23.1
Erdos [9]	0.7162 [20]	0.6993	0.7021 (32)	0.7184	**0.7188 (34)**	88.4
Arxiv [11]	0.813 [1]	0.8166	0.8181 (60)	0.8246	**0.8254 (56)**	197.2
PGP [2]	0.8841 [13,20]	0.8841	0.8850 (95)	0.8865	**0.8867 (94)**	156.7
Condmat2003	0.8146 [20]	0.8112	0.8116 (77)	0.8165	**0.8170 (73)**	1369.7

of MA-COM, the best solution I^*_{init} (i.e., the clustering with the largest modularity) from the initial population (generated by BGLL) and the best solution I^*_{final} from the final population (generated by MA-COM). We compute then the distance between I^*_{init} and I^*_{final} using two distance measures: the well-known Normalized Mutual Information (NMI) and the Edge Rand Index (ERI) which is defined in Section 2.4 for population management. While NMI measures the information shared by I^*_{init} and I^*_{final}, ERI indicates the percentage of edges which disagree in the clusterings I^*_{init} and I^*_{final}. Table 2 show the statistics of these measures averaged over the 20 runs for each graph. Additionally, we indicate the averaged number of communities (indicator K) in the initial and final population. Finally, we present the averaged sizes of the smallest and the largest communities in the initial and final best solutions.

Table 2 shows that for the small graphs except *Dolphins*, the memetic algorithm has a limited effect on the best BGLL clustering. On the contrary, structural changes for other graphs are more or less important because an edges difference of 2.7% to 13.1% are observed in the initial best and the final best solutions. Some graphs have probably a simple structure with few local optima, for instance *PGP* (with a high NMI). Some smaller graphs like *C. elegans* seem to have a more complexe modularity landscape (13.1% of edges of the initial best solutions are changed in final best solutions).

The indicator K confirms the well-known propensity of modularity based methods to reduce the number of communities. However, the reduction is moderate, indicating that the changes revealed by the ERI distance would be mainly due to moves of vertices rather than merges of community. The good surprise comes with the smallest and largest communities. The memetic algorithm has a clear trend to help discover small communities (which are known to be difficult to detect). More generally, we believe that the crossover operator of the

Table 2. Several structural measures to compare the best solution in the initial population and the best solution in the final population: NMI (Normalized Mutual Information), ERI (Edge Rand Index), K (number of communities), average sizes of smallest and largest community over 20 runs.

Graph	NMI	ERI	K		Smallest com. size		Largest com. size	
			Initial	Final	Initial	Final	Initial	Final
Karate Club	1.000	0.0%	4.0	4.0	5.0	5.0	12.0	12.0
Dolphins	0.976	1.9%	5.0	5.0	5.0	5.0	19.9	20.0
Political Books	0.982	0.4%	5.0	5.0	3.0	3.0	40.6	40.0
College Football	1.000	0.0%	10.0	10.0	9.0	9.0	16.0	16.0
Jazz	0.999	0.1%	4.0	4.0	21.9	22.0	62.1	62.0
C. elegans	0.733	13.1%	10.2	9.2	7.5	5.0	92.0	82.2
E-mail	0.780	9.1%	10.8	10.1	43.2	36.2	185.8	168.5
Erdos	0.771	12.0%	32.5	33.9	23.8	9.7	622.5	619.6
Arxiv	0.795	7.6%	59.6	55.5	4.5	4.5	920.5	812.2
PGP	0.915	2.7%	98.0	95.0	5.9	6.0	668.5	641.7
Condmat2003	0.758	8.9%	75.8	70.6	18.5	6.6	2478.7	2266.6
Total	0.883	5.1%	28.6	27.5	13.4	10.2	465.3	431.0

algorithm acts mainly on the ambiguous vertices which are attached to several communities and help discover the right community for these vertices.

4 Conclusion and perspectives

This paper deals with the problem of community detection in complex networks with the popular modularity criterion. To approximate this hard combinatorial problem, we proposed a memetic algorithm mixing a dedicated crossover operator and a multi-level local optimization procedure. The proposed crossover operator blends the communities of two clusterings (parents) according to a priority rule. Offspring solutions are improved with the multi-level local optimizer. To maintain a healthy population diversity, we introduce a Rand Index based distance and consider both the quality of an offspring solution and its distance to the solutions of the population. Experimental results on a set of 11 popular networks showed that the proposed approach can easily match the best known results in 5 cases and discover improved solutions for the 6 other largest networks. The analysis of initial solutions and final solutions showed the benefit of memetic approach in discovering communities of small size that are difficult to find. This work demonstrated that the memetic approach is a very effective method for modularity maximization. The proposed algorithm could also be used to devise more powerful methods. One possible way would be to embed the memetic approach into the multi-level approach in order to handle very large networks.

Acknowledgment

The work is partially supported by the Pays de la Loire Region (France) within the RaDaPop (2009-2013) and LigeRO (2010-2013) projects.

References

1. V. D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre. Fast unfolding of communities in large networks. *J. Stat. Mech: Theory Exp.*, 10:8–+, October 2008.
2. M. Boguñá, R. Pastor-Satorras, A. Díaz-Guilera, A. Arenas. Models of social networks based on social distance attachment. *Phys. Rev. E*, 70(5):056122, 2004.
3. U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, D. Wagner. On modularity clustering. *IEEE Trans. Knowl. Data Eng.*, 20(2): 172–188, 2008.
4. A. Clauset, M. E. J. Newman, C. Moore. Finding community structure in very large networks. *Phys. Rev. E*, 70(6):066111, 2004.
5. J. Duch and A. Arenas. Community detection in complex networks using extremal optimization. *Phys. Rev. E*, 72(2):027104, 2005.
6. S. Fortunato. Community detection in graphs. *Physics Reports* 486: 75–174, 2010.
7. M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proc. Natl. Acad. Sci. USA*, 99(12):7821–7826, 2002.
8. P. Gleiser and L. Danon. Community structure in social and biological networks. *Advances in Complex Systems*, 6:565–573, 2003.
9. J. Grossman. The Erdös number project. http://www.oakland.edu/enp/, 2007.
10. R. Guimerà, L. Danon, A. Díaz-Guilera, F. Giralt, A. Arenas. Self-similar community structure in a network of human interactions. *Phys. Rev. E*, 68(6):065103, 2003.
11. KDD. Cornell kdd cup. http://www.cs.cornell.edu/projects/kddcup/, 2003.
12. V. Krebs. A network of books about recent us politics sold by the online bookseller amazon.com. http://www.orgnet.com, 2008.
13. X. Liu and T. Murata. Advanced modularity-specialized label propagation algorithm for detecting communities in networks. *Phys. A*, 389(7):1493–150 2009.
14. Z. Lü and W. Huang. Iterated tabu search for identifying community structure in complex networks. *Phys. Rev. E*, 80(2):026130, 2009.
15. D. Lusseau, K. Schneider, O. J. Boisseau, P. Haase, E. Slooten, S. M. Dawson. The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations. *Behav. Ecol. Sociobiol.*, 54(4):396–405, 2003.
16. F. Neri, C. Cotta, P. Moscato (Eds.) Handbook of Memetic Algorithms. Studies in Computational Intelligence 379, Springer, 2011.
17. M. E. J. Newman. The structure of scientific collaboration networks. *Proc. Natl. Acad. Sci. USA*, 98(2):404–409, 2001.
18. M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69(2):026113, 2004.
19. M. E. J. Newman. Networks: An Introduction. Oxford University Press, 2010.
20. A. Noack and R. Rotta. Multi-level algorithms for modularity clustering. *LNCS*, 5526:257–268, 2009.
21. W. M. Rand. Objective criteria for the evaluation of clustering methods. *J. Amer. Statistical Assoc.*, 66(336):846–850, 1971.
22. P. Schuetz and A. Caflisch. Efficient modularity optimization by multistep greedy algorithm and vertex mover refinement. *Phys. Rev. E*, 77(4):046112, 2008.
23. W. W. Zachary. An information flow model for conflict and fission in small groups. *J. Anthropol. Res.*, 33:452–473, 1977.