Improving heuristics for network modularity maximization using an exact algorithm

Sonia Cafieri[1], Pierre Hansen[2,3], and Leo Liberti[3]

[1] Dept. Mathématiques et Informatique, ENAC, 7 av. E. Belin, 31055 Toulouse, France, sonia.cafieri@enac.fr
[2] GERAD, HEC Montreal, Canada, pierre.hansen@gerad.ca
[3] LIX, École Polytechnique, 91128 Palaiseau, France, liberti@lix.polytechnique.fr

Abstract. Heuristics are widely applied to modularity maximization models for the identification of communities in complex networks. We present an approach to be applied as a post-processing on heuristic methods in order to improve their performances. Starting from a given partition, we test with an exact algorithm for bipartitioning if it is worthwhile to split some communities or to merge two of them. A combination of merge and split actions is also performed. Computational experiments show that the proposed approach is effective in improving heuristic results.
Keywords: clustering, bipartition, network, graph, community, modularity, heuristic, exact algorithm.

1 Introduction

The identification of communities in complex networks has become in recent years a very active research domain because of the common representation of complex real-world systems arising in a variety of fields as networks, where one aims to find *communities*, or *clusters*, of entities grouped on the basis of some relationship holding among them. Telecommunication networks such as the World Wide Web, biological networks representing interactions between organisms and transportation networks used to represent movements of goods and passengers, are some examples of real-life applications.

A very successfull class of methods to detect communities in networks is based on the concept of modularity. In [1] the definition of modularity for a partition of a network as the sum for all communities of the difference between the fraction of edges they contain and the expected fraction of edges if they are placed at random, keeping the same degree distribution, was introduced. High values of modularity correspond to a clear partition of a network. Methods for the identification of communities are then based on the idea of maximizing modularity. Many heuristics have been proposed, while exact algorithms for modularity maximization are rare. Heuristics are based on agglomerative hierarchical clustering [2–6], simulated annealing [7–9], mean field annealing [10], genetic search [11], extremal optimization [12], spectral clustering [13], linear programming followed

by randomized rounding [14], dynamical clustering [15], multilevel partitioning [16], contraction-dilation [17], multistep greedy search [18], quantum mechanics [19] and many more [6, 20–24]. These heuristics are able to solve large instances with up to hundred or thousand entities and therefore are often preferred to exact algorithms, even though they do not have a guarantee of optimality. Exact algorithms for maximizing modularity have been proposed in [25, 26]. They can only solve instances with about a hundred entities in reasonable time.

Given a partition found by a heuristic, one can apply an exact algorithm to the reduced networks represented by the communities found, thus obtaining a new, better partition. We propose such an approach to improve heuristic solutions, which is based on merging and splitting some communities if this is worthwhile in terms of increase of the modularity value. An exact algorithm for bipartitioning is used to split a community. We apply our approach as post-processing of some known heuristics for modularity maximization, obtaining improved solutions and, for some datasets, the optimal partition. These results, that show the impact of exact algorithms on heuristic schemes, may in turn lead to more efficient exact algorithms, where some steps are sometimes performed heuristically. This is the case, for example, of column generation algorithms.

The paper is organized as follows. In the next section, the proposed approach to improve heuristic results for modularity maximization is described, presenting in particular an exact algorithm for bipartition. Section 3 presents the results of computational experiments carried out applying the proposed approach as post-processing to two known heuristics on some datasets from the literature. Conclusions are given in Section 4.

2 Improving heuristics for modularity maximization

2.1 An exact algorithm for bipartitions

We present in this section an exact algorithm for bipartition. Although it can be applied in full generality to any graph, we specifically test it in the role of post-processing step to heuristic algorithms for the identification of community in networks, to improve their solution quality.

We model the bipartitioning problem using some variables to identify to which community each vertex and each edge belongs. In this respect, our model is similar to that of Xu et al. [26]. These authors proposed in 2007 [26] a modularity maximization model to obtain a partition (generally with more than two communities) of a network, that leads to a mixed integer convex quadratic program. They were able to solve exactly instances up to 104 vertices.

Let $G = (V, E)$ be a graph, or network, with set of vertices V of cardinality n and set of edges E of cardinality m. First, we recall the definition of modularity Q as a sum over communities of their modularities [1]:

$$Q = \sum_s [a_s - e_s],$$

where a_s is the fraction of all edges that lie within community s and e_s is the expected value of the same quantity in a graph in which the vertices have the same degrees but edges are placed at random. Modularity can then be written equivalently as:

$$Q = \sum_s \left[\frac{m_s}{m} - \left(\frac{d_s}{2m} \right)^2 \right], \tag{1}$$

where m_s denotes the number of edges in community s, i.e., the subgraph induced by V_S, and d_s denotes the sum of degrees k_i of the vertices of community s. Since we aim to find a bipartition, only two sub-modules of the original community have to be considered, i.e. $s \in \{1, 2\}$. We can express the sum of degrees d_2 of vertices belonging to the second community as a function of the sum of degrees d_1 of vertices belonging to the first one:

$$d_2 = d_t - d_1, \tag{2}$$

where d_t is the sum of degrees in the community to be bipartitioned and it is equal to $2m$ when one aim to bipartition the original network, containing all the entities. We rewrite (1) for $s \in \{1, 2\}$, using (2):

$$\begin{aligned} Q &= \frac{m_1 + m_2}{m} - \frac{d_1^2}{4m^2} - \frac{d_2^2}{4m^2} = \\ &= \frac{m_1 + m_2}{m} - \frac{d_1^2}{4m^2} - \frac{d_t^2 + d_1^2 - 2d_t d_1}{4m^2} = \\ &= \frac{m_1 + m_2}{m} - \frac{d_1^2}{4m^2} - \frac{d_t^2}{4m^2} + \frac{d_t d_1}{2m^2}. \end{aligned} \tag{3}$$

We then introduce binary variables X_{r1}, X_{r2} and Y_{i1}, Y_{i2} to model assignment of vertices and edges to the two communities of the bipartition. These variables are defined as follows:

$$X_{rs} = \begin{cases} 1 \text{ if edge } \ r \text{ belongs to module } s \\ 0 \text{ otherwise} \end{cases} \tag{4}$$

for $r = 1, 2, \ldots m$ and $s = 1, 2$ and

$$Y_{is} = \begin{cases} 1 \text{ if vertex } i \text{ belongs to module } s \\ 0 \text{ otherwise} \end{cases} \tag{5}$$

for $i = 1, 2, \ldots n$ and $s = 1, 2$.
We impose that any edge $r = \{v_i, v_j\}$ with end vertices indiced by i and j can only belong to community s if both of its end vertices belong also to that community:

$$\begin{aligned} X_{r1} \leq Y_{i1} \quad \forall r = \{v_i, v_j\} \in E \\ X_{r1} \leq Y_{j1} \quad \forall r = \{v_i, v_j\} \in E \end{aligned} \tag{6}$$

and

$$\begin{aligned} X_{r2} \leq 1 - Y_{i1} \quad \forall r = \{v_i, v_j\} \in E \\ X_{r2} \leq 1 - Y_{j1} \quad \forall r = \{v_i, v_j\} \in E. \end{aligned} \tag{7}$$

Furthermore, we exploit the following expressions in terms of variables X and Y for the number of edges of each of the two communities and the sum of vertex degrees of the first one:

$$m_s = \sum_r X_{rs} \quad \forall s \in \{1, 2\}, \tag{8}$$

$$d_1 = \sum_{i \in V_1} k_i Y_{i1}. \tag{9}$$

The sum of vertex degrees of the first community only is exploited, because of expression (2).

Maximizing modularity (3) subject to constraints (6)-(7) and (8)-(9) gives a mixed-integer program that can be solved by `CPLEX` [27].

Note that in Xu et al.'s model a number of other constraints are imposed on the variables. For example, constraints are used to express that community s can be nonempty only is community $s-1$ is so and lower and upper bounds on the cardinality of the modules are imposed. Furthermore, symmetry-breaking constraints avoid the computation of alternative equivalent solutions.

2.2 Improving partitioning by merging and splitting

Given a partition of a network in communities, we aim to improve the partition computing a new one with a high value of modularity. As we apply our approach as post-processing to available heuristics for modularity maximization, the initial partition is represented by the solution provided by the considered heuristic. A new partition is obtained in a sequence of steps, which act on the current communities by splitting and merging.

First, for each community of the original partition, we split it into two subcommunities by applying the exact algorithm for bipartition described in subsection 2.1 and we check if the modularity value corresponding to the obtained bipartition is higher than the one of the original community. It is worth noticing that this comparison makes sense because of the definition of modularity of a partition as sum of modularities of its communities. If the new modularity value is higher than the one of the original community, such community is replaced by the two new communities, otherwise the two obtained communities are discarded and the original one is kept. When all the original communities have been checked, a new partition is obtained with a high modularity if at least one bipartition has been accepted.

A next step consists in trying to merge pairs of communities if this is worthwhile for modularity. For each pair of communities, we consider the new community containing all vertices of this pair and we check if the larger community has a modularity value higher than the one corresponding to the two original communities. If this is the case, the new large community is kept in place of the other two. Otherwise, if merge is not beneficial, we try to split the merged community using again the exact algorithm presented in subsection 2.1. As before,

the two communities resulting from the bipartition are kept if they correspond to an increased value of modularity.

A sketch of our algorithm is given in Alg. 1.

Algorithm 1

/* ncl = number of communities of the partition found by a heuristic */
/* CL_i = community of the partition found by a heuristic, $\forall i = \{1, \ldots, ncl\}$ */
Require: $V, E, ncl, CL_i \; \forall i = \{1, \ldots, ncl\}$
$ncl_{split} \leftarrow 0$
for all $i \leq ncl$ **do**
  split CL_i into CL_1, CL_2 using algorithm in subsection 2.1
  if $Q(CL_1) + Q(CL_2) > Q(CL_i)$ **then**
    replace CL_i with CL_1, CL_2
  else
    keep CL_i
  end if
  $ncl_{split} \leftarrow$ number of communities of the new partition
end for
$ncl_{merge+split} \leftarrow ncl_{split}$
for all $i \leq ncl_{split}$ **do**
  $listcl \leftarrow$ list of pairs of communities (CL_j, CL_k), $j, k \in \{1, \ldots, ncl_{split}\}$
  while $listcl \neq \emptyset$ **do**
    select a pair of communities CL_j, CL_k from $listcl$
    merge CL_j and CL_k into CL_m
    if $Q(CL_m) > Q(CL_j) + Q(CL_k)$ **then**
      replace CL_j, CL_k with $CL_m = CL_j \cup CL_k$
    else
      split CL_m into CL_{m1}, CL_{m2}
      if $Q(CL_{m1}) + Q(CL_{m2}) > Q(CL_m)$ **then**
        replace CL_m with CL_{m1}, CL_{m2}
      else
        keep CL_m
      end if
    end if
    update $listcl$
  end while
end for
$ncl_{merge+split} \leftarrow$ number of communities of the new partition
compute modularity $Q = \sum_{i=1}^{i=ncl_{merge+split}} Q(CL_i)$
return final partition, Q

3 Computational results

We describe the results of some computational experiences carried out applying our strategy, described in subsections 2.1, 2.2, as post-processing to two known

heuristics for modularity maximization, due to Clauset, Newman and Moore [3] and Noack and Rotta [28] respectively, whose implementations are freely available. Clauset et al. [3] proposed in 2004 an efficient agglomerative hierarchical scheme, that for sparse networks has a very low complexity and is considerably faster than previously proposed methods. Noack and Rotta [28] recently presented a comparison of heuristics for modularity maximization and proposed a heuristic based on a single-step coarsening with a multi-level refinement, which is competitive with other methods in the literature. For details about these two heuristic schemes, the reader is referred to [3, 28].

Our computational results have been obtained on some datasets that are often used to evaluate heuristics and algorithms for identification of communities in networks. These datasets correspond to various real-life applications: a social network of dolphins described by Lusseau [29], a network describing interactions among the characters of Hugo's novel *Les Miserables* [30], a network dealing with *p53* protein [31], a network dealing with co-purchasing of political books on `Amazon.com` [32], a network of common adjective and noun adjacencies for the Dickens' novel *David Copperfield*, as described by Newman [33], a network representing the schedule of games between American college football teams in the Fall 2000 [34], a network dealing with connections between US airports [35], a network describing electronic circuits [36], a network dealing with e-mail interchanges between members of a university [37], a network representing the topology of the Western States Power Grid of the United States [38] and a network of authors collaborations [35].

In Table 1 we report, for each dataset, the values of modularity computed by the two considered heuristics and by our proposed approach when applied as post-processing to these methods, together with the optimal value of modularity, when available in the literature. The number of vertices and the number of edges of the datasets are also reported. A comparison in terms of modularity values shows that we are able to improve results provided by the considered heuristics for all the tested datasets, sometimes significantly. A significant improvement is obtained, for example, in the case of Clauset et al.'s solution for `dolphin`, `les miserables` and `usair97` datasets. We transformed solutions for some datasets, i.e. Noack and Rotta's solutions for `dolphin`, `political books` and `football` datasets and Clauset et al.'s solution for `political books` dataset, into optimal ones.

Our approach is based on two moves, a splitting and a merging move, the merging one being in turn used in conjunction with a splitting when merging is not beneficial. These two main steps, which we call `split` and `merge+split` for short, are applied sequentially. In order to evaluate the impact of the two steps, we report in Table 2 the modularity values obtained applying `split` and `merge+split` starting from Clauset et al.'s (*CNM*) solution and from Noack-Rotta's (*NR*) solution. Note that modularity values for `merge+split` are the final results provided by our moves, already shown in Table 1. These results show that the splitting step provides in most cases a significant improvement of the original partition. Examples are given by `dolphin`, `usair97` and `email` datasets

dataset	n	m	Q_{CNM}	Q	Q_{NR}	Q	Q_{opt}
dolphin	62	159	0.49549	0.51958	0.52377	0.52852	0.52852
les miserables	77	254	0.50060	0.54039	0.56001	0.56001	0.56001
p53 protein	104	226	0.52052	0.52621	0.53216	0.53502	0.53513
political books	105	441	0.50197	0.52724	0.52694	0.52724	0.52724
adjnoun	112	425	0.29349	0.29446	0.30729	0.30848	–
football	115	613	0.57728	0.58685	0.60028	0.60457	0.60457
usair97	332	2126	0.32039	0.36161	0.36577	0.36605	–
s838	512	819	0.80556	0.80914	0.81624	0.81656	–
email	1133	5452	0.51169	0.53808	0.57740	0.57773	0.579
power	4941	6594	0.93402	0.93612	0.93854	0.93870	–
erdos02	6927	11850	0.78092	0.78095	0.71552	0.7157	–

Table 1. Results on real-world datasets: modularity values found by the Clauset et al.'s heuristic (Q_{CNM}), by the Noack and Rotta's heuristic (Q_{NR}) and by our proposed approach (Q). Q_{opt} are the optimal modularity values as reported in the literature. n and m are the number of vertices and the number of edges of the networks.

for *CNM*, where an improvement on the second decimal digit of modularity value is obtained. Furthermore, the splitting step provides the optimal solution of `political books` dataset for *NR*. By contrast, this step does not provide for some instances a better partition than the original one, leading to an unchanged modularity value. Examples are given by `adjnoun` and `erdos02` datasets for *CNM* and by `p53 protein`, `usair97` and `s838` datasets for *NR*. This behavior shows the importance of a combined use of both splitting and merging steps.

dataset	*CNM*			*NR*		
	Q_{orig}	Q_{split}	$Q_{merge+split}$	Q_{orig}	Q_{split}	$Q_{merge+split}$
dolphin	0.49549	0.51693	0.51958	0.52377	0.52773	0.52852
les miserables	0.50060	0.50732	0.54039	0.56001	0.56001	0.56001
p53 protein	0.52052	0.52518	0.52621	0.53216	0.53216	0.53502
political books	0.50197	0.52708	0.52724	0.52694	0.52724	0.52724
adjnoun	0.29349	0.29349	0.29446	0.30729	0.30847	0.30848
football	0.57728	0.58232	0.58685	0.60028	0.60237	0.60457
usair97	0.32039	0.36157	0.36161	0.36576	0.36576	0.36605
s838	0.80556	0.80639	0.80914	0.81624	0.81624	0.81656
email	0.51169	0.53761	0.53808	0.57740	0.57741	0.57773
power	0.93402	0.93605	0.93612	0.93854	0.93867	0.93870
erdos02	0.78092	0.78092	0.78095	0.71552	0.71561	0.7157

Table 2. Modularity values corresponding to the partition found by the heuristic (Q_{orig}) and by our approach after the splitting step only (Q_{split}) and after the successive application of the merging and splitting step ($Q_{merge+split}$) for Clauset et al.'s heuristic (*CNM*) and Noack and Rotta's heuristic (*NR*).

Table 3 shows computing time required by our post-processing strategy applied to the two considered heuristics to get an improved solution. As expected, times are roughly increasing with network dimension, even though they depend mostly on the quality of the initial partition and the cardinality of its communities to be handled. Times are in general reasonably moderate, and very short times are spent on most of the tested networks. The optimal partition is found in less than 1 second for `dolphin` dataset starting from *NR* solution and in less than 10 and 16 seconds respectively for `political books` starting from *NR* solution and from *CNM* solution.

dataset	$time_{CNM}$	$time_{NR}$
`dolphin`	0.48	0.75
`les miserables`	0.38	1.17
`p53 protein`	0.58	2.28
`political books`	9.58	15.70
`adjnoun`	0.83	1.68
`football`	0.97	7.57
`usair97`	164342.86	2752.27
`s838`	2.61	11.95
`email`	334662.61	614.20
`power`	49.59	113.80
`erdos02`	147915.81	32778.81

Table 3. Computing time (seconds) required by the proposed approach applied as post-processing to Clauset at al.'s heuristic ($time_{CNM}$) and Noack and Rotta's heuristic ($time_{NR}$). Solutions have been obtained on a 2.4 GHz Intel Xeon CPU of a computer with 8GB RAM shared by three other similar CPU running Linux.

4 Conclusion

This paper describes the application of an approach based on an exact algorithm for bipartitioning a network, in the framework of split and merge movements on communities of a network partition. Computational results obtained on a set of examples from the literature, applying the proposed approach as post-processing to two known heuristics for modularity maximization of networks, show the impact of an exact approach on the improvement of heuristic results.

The presented approach may be further developed including the described moves directly in a local search heuristic as well as in a hierarchical divisive clustering algorithm based on exact bipartitions. Future work will address these points.

References

1. Newman, M., Girvan, M.: Finding and evaluating community structure in networks. Physical Review E **69** (2004) 026133
2. Newman, M.: Fast algorithm for detecting community structure in networks. Physical Review E **69** (2004) 066133
3. Clauset, A., Newman, M., Moore, C.: Finding community structure in very large networks. Physical Review E **70** (2004) 066111
4. Danon, L., Diaz-Guilera, A., Arenas, A.: The effect of size heterogeneity on community identification in complex networks. Journal of Statistical Mechanics **P11010** (2006)
5. Wakita, K., Tsurumi, T.: Finding community structure in mega-scale social networks. Technical Report 0702048v1, arXiv (2007)
6. Blondel, V., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. Journal Statistical Mechanics (P10008) (2008)
7. Guimerà, R., Amaral, A.: Functional cartography of complex metabolic networks. Nature **433** (2005) 895–900
8. Massen, C., Doye, J.: Identifying communities within energy landscapes. Physical Review E **71** (2005) 046101
9. Medus, A., Acuna, G., Dorso, C.: Detection of community structures in networks via global optimization. Physica A **358** (2005) 593–604
10. Lehmann, S., Hansen, L.: Deterministic modularity optimization. European Physical Journal B **60** (2007) 83–88
11. Tasgin, M., Herdagdelen, A., Bingol, H.: Community detection in complex networks using genetic algorithms. arXiv:0711.0491 (2007)
12. Duch, J., Arenas, A.: Community identification using extremal optimization. Physical Review E **72**(2) (2005) 027104
13. Newman, M.: Modularity and community structure in networks. Proceedings of the National Academy of Sciences, USA (2006) 8577–8582
14. Agarwal, G., Kempe, D.: Modularity-maximizing graph communities via mathematical programming. The European Physical Journal B **66**(3) (2008) 409–418
15. Boccaletti, S., Ivanchenko, M., Latora, V., Pluchino, A., Rapisarda, A.: Detecting complex network modularity by dynamical clustering. Physical Review E **75** (2007) 045102
16. Djidjev, H.: A scalable multilevel algorithm for graph clustering and community structure detection. Lecture Notes in Computer Science **4936** (2008)
17. Mei, J., He, S., Shi, G., Wang, Z., Li, W.: Revealing network communities through modularity maximization by a contraction-dilation method. New Journal of Physics **11** (2009) 043025
18. Schuetz, P., Caflisch, A.: Efficient modularity optimization by multistep greedy algorithm and vertex mover refinement. Physical Review E **77** (2008) 046112
19. Niu, Y., Hu, B., Zhang, W., Wang, M.: Detecting the community structure in complex networks based on quantum mechanics. Physica A **387**(24) (2008) 6215–6224
20. Chen, D., Fu, Y., Shang, M.: A fast and efficient heuristic algorithm for detecting community structures in complex networks. Physica A **388**(13) (2009) 2741–2749
21. Sun, Y., Danila, B., Josic, K., Bassler, K.E.: Improved community structure detection using a modified fine-tuning strategy. Europhysics Letters **86** (2009) 28004
22. Ruan, J., Zhang, W.: Identifying network communities with a high resolution. Physical Review E **77** (2008) 016104

23. Fan, Y., Li, M., Zhang, P., Wu, J., Di, Z.: Accuracy and precision of methods for community identification in weighted networks. Physica A **377**(1) (2007) 363–372
24. Kumpula, J., Saramaki, J., Kaski, K., Kertesz, J.: Limited resolution and multiresolution methods in complex network community detection. Fluctuation and Noise Letters **7**(3) (2007) L209–L214
25. Brandes, U., Delling, D., Gaertler, M., Görke, R., Hoefer, M., Nikoloski, Z., Wagner, D.: On modularity clustering. IEEE Transactions on Knowledge and Data Engineering **20**(2) (2008) 172–188
26. Xu, G., Tsoka, S., Papageorgiou, L.: Finding community structures in complex networks using mixed integer optimization. Eur. Physical Journal B **60** (2007) 231–239
27. ILOG: ILOG CPLEX 11.0 User's Manual. ILOG S.A., Gentilly, France. (2008)
28. Noack, A., Rotta, R.: Multi-level algorithms for modularity clustering. Technical Report 0812.4073v2, arXiv (2008)
29. Lusseau, D., Schneider, K., Boisseau, O., Haase, P., Slooten, E., Dawson, S.: The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations. can geographic isolation explain this unique trait? Behavioral Ecology and Sociobiology **54**(4) (2003) 396–405
30. Knuth, D.: The Stanford GraphBase: A Platform for Combinatorial Computing. Addison-Wesley, Reading, MA (1993)
31. Dartnell, L., Simeonidis, E., Hubank, M., Tsoka, S., Bogle, I., Papageorgiou, L.: Self-similar community structure in a network of human interactions. FEBS Letters **579** (2005) 3037–3042
32. Krebs, V.: `http://www.orgnet.com/` (unpublished).
33. Newman, M.E.J.: Finding community structure in networks using the eigenvectors of matrices. Physical Review E **74** (2006) 036104
34. Girvan, M., Newman, M.: Community structure in social and biological networks. Proceedings of the National Academy of Sciences, USA **99**(12) (2002) 7821–7826
35. : `http://vlado.fmf.uni-lj.si/pub/networks/data/`.
36. Lab, U.A.: `http://www.weizmann.ac.il/mcb/UriAlon/`.
37. Guimerà, R., Danon, L., Diaz-Guilera, A., Giralt, F., Arenas, A.: Self-similar community structure in a network of human interactions. Physical Review E **68** (2003) 065103
38. Watts, D., Strogatz, S.: Collective dynamics of 'small-world' networks. Nature **393**(6684) (1998) 409–410