Reformulation of a locally optimal heuristic for modularity maximization

Alberto Costa, Sonia Cafieri, Pierre Hansen

Reformulation of a locally optimal heuristic for modularity maximization

Alberto Costa[1], Sonia Cafieri[2], Pierre Hansen[1,3]

[1] LIX, École Polytechnique, F-91128 Palaiseau, France
costa@lix.polytechnique.fr
[2] École Nationale de l'Aviation Civile, F-31055 Toulouse, France
sonia.cafieri@enac.fr
[3] GERAD, HEC, 3000 chemin de la Côte-S.te-Catherine, H3T 2A7 Montréal, Canada
pierre.hansen@gerad.ca

Keywords : *binary decomposition, clustering, modularity maximization, reformulation.*

1 Introduction

A network, or graph, $G = (V, E)$ consists of a set of vertices $V = \{1, \ldots, n\}$ and a set of edges $E = \{1, \ldots, m\}$ connecting vertices. One of the most studied problems in the field of complex systems is to find communities, or clusters, in networks. A community consists of a subset S of the vertices of V where inner edges connecting pairs of vertices of S are more dense than cut edges connecting vertices of S to vertices of $V \backslash S$. Many criteria have been proposed to evaluate partitions of V into communities. The best known of them appears to be the modularity, defined as follows by Newman and Girvan [9]:

$$Q = \sum_c Q_c = \sum_c \left(\frac{m_c}{m} - \frac{{D_c}^2}{4m^2} \right), \tag{1}$$

where Q_c is the modularity of the cluster c, m_c is the number of edges with both end vertices within the cluster c, D_c is the sum of the degrees of the vertices in the cluster c, and m is the number of edges of the whole network. The modularity is the difference between the fraction of edges within communities and the expected fraction of such edges in a random graph having the same distribution of degrees than the graph under study. In order to find a good partition into communities for a given network, according to Newman and Girvan one should maximize its modularity. This is a strongly NP-hard problem [3].

A few exact algorithms [1, 6, 10] and many heuristics have been proposed for network modularity maximization. They consist in divisive and agglomerative hierarchical clustering approaches [5, 8], as well as exact or approximate partitioning ones. In this paper, we focus on a recent locally optimal heuristic based on a hierarchical divisive approach [4]. We propose several ways to reformulate the model of [4] in order to accelerate the resolution by reducing efficiently the number of variables and constraints. Computation results are reported for a series of real-world problems from the literature in which the different reformulations are compared. It appears that computing times are very substantially reduced.

2 Initial model

The model used in the framework of the hierarchical divisive heuristic proposed in [4] to split a cluster (V_c, E_c) into two clusters maximizing the modularity, and based on the one proposed in [10], is the following:

$$
\begin{aligned}
\max \quad & \frac{1}{m}\left(m_1 + m_2 - \frac{1}{2m}\left({D_1}^2 + \frac{{D_c}^2}{2} - D_1 D_c\right)\right) && (2)\\
\text{s.t.} \quad & X_{i,j,1} \leq Y_i \quad \forall (v_i, v_j) \in E_c && (3)\\
& X_{i,j,1} \leq Y_j \quad \forall (v_i, v_j) \in E_c && (4)\\
& X_{i,j,2} \leq 1 - Y_i \quad \forall (v_i, v_j) \in E_c && (5)\\
& X_{i,j,2} \leq 1 - Y_j \quad \forall (v_i, v_j) \in E_c && (6)\\
& m_s = \sum_{(v_i,v_j)\in E_c} X_{i,j,s} \quad \forall s \in \{1, 2\} && (7)\\
& D_1 = \sum_{v_i \in V_c} k_i Y_{i,1} && (8)\\
& Y_i \in \{0, 1\} \quad \forall v_i \in V_c && (9)\\
& X_{i,j,s} \geq 0 \quad \forall (v_i, v_j) \in E_c, \forall s \in \{1, 2\}, && (10)
\end{aligned}
$$

where the variable $X_{i,j,s}$ is equal to 1 if the edge (v_i, v_j) is inside the community s (i.e., both vertices v_i and v_j are inside the community s) and 0 otherwise, Y_i is equal to 1 if the vertex v_i is inside the community 1, and 0 otherwise, and k_i is the degree of the vertex v_i; note that D_c is a parameter, and it is known before solving the problem.

3 Reformulations

3.1 Power of two reformulation

The heuristic proposed in [4] works by recursively splitting a cluster into two clusters in an optimal way (in the sense that the computed bipartition corresponds to the best possible modularity). The model is a quadratic integer programming one, with a convex relaxation. The only non-linear term is ${D_1}^2$. The usual Branch-and-Bound approach implemented in CPLEX [7] is to relax the integrality constraints, solve the continuous quadratic program obtained and then branch. Alternately, one may linearize ${D_1}^2$ by replacing it with its expansion in power of two, as proposed for mixed-integer quadratic programming in [2]:

$$D_1 = \sum_{i=0}^{t} 2^i a_i, \quad a_i \in \{0, 1\}. \tag{11}$$

Therefore, the term ${D_1}^2$ in (2) can be written as:

$${D_1}^2 = \sum_{l=0}^{t} 2^l a_l \cdot \sum_{h=0}^{t} 2^h a_h = \sum_{l=0}^{t}\sum_{h=0}^{t} 2^{l+h} a_l a_h = \sum_{l=0}^{t}\sum_{h=0}^{t} 2^{l+h} R_{lh} = \sum_{l=0}^{t} 2^{2l} a_l + \sum_{l=0}^{t}\sum_{h<l} 2^{l+h+1} R_{lh}, \tag{12}$$

where R_{lh} is the linearization variable for $a_l a_h$; hence, we have to adjoin the following constraints to our model:

$$
\begin{aligned}
& R_{lh} \geq a_l + a_h - 1, \quad \forall l \in \{0, \ldots, t\}, \forall h \in \{0, \ldots, l-1\}\\
& R_{lh} \geq 0, \quad \forall l \in \{0, \ldots, t\}, \forall h \in \{0, \ldots, l-1\}.
\end{aligned}
$$

To estimate t, recall that the maximum value which can be assumed by D_1 is the sum of the degrees of all the vertices in the current cluster, that is D_c. Moreover, from (11) the maximum possible value for D_1 is $2^{t+1} - 1$. Hence, t can be computed as:

$$2^{t+1} - 1 \geq D_c \quad \Rightarrow \quad t = \lceil \log_2(D_c + 1) - 1 \rceil. \tag{13}$$

3.2 Change of variables

The model of [4] uses variables assigning edges or vertices to a specific community. When bipartitioning, as there are only two communities to be determined at each iteration, one can use other variables $S_{i,j}$, associated with the fact that the two end vertices v_i and v_j of an edge belong to the same cluster or not (i.e., $S_{i,j} = 1$ if $Y_i = Y_j$, and 0 otherwise). This leads to the following reformulation:

$$
\begin{aligned}
\max \quad & \frac{1}{m}\left(\sum_{(v_i,v_j)\in E_c}(2S_{i,j} - Y_i - Y_j) + |E_c| - \frac{1}{2m}\left({D_1}^2 + \frac{{D_c}^2}{2} - D_1 D_c\right)\right) && (14)\\
\text{s.t.} \quad & S_{i,j} \le Y_i \quad \forall (v_i, v_j) \in E_c && (15)\\
& S_{i,j} \le Y_j \quad \forall (v_i, v_j) \in E_c && (16)\\
& D_1 = \sum_{v_i \in V_c} k_i Y_i && (17)\\
& Y_i \in \{0,1\} \quad \forall v_i \in V_c. && (18)
\end{aligned}
$$

3.3 Symmetry breaking

To avoid considering twice equivalent solutions, one fixes a vertex to belong to the first (or second) community. It appears that the vertex with largest degree is a good choice.

4 Compact model

Applying all the reformulations presented in the previous sections leads to the following compact model:

$$
\max \quad \frac{1}{m}\left(\sum_{(v_i,v_j)\in E_c}(2S_{i,j} - Y_i - Y_j) + |E_c| - \frac{1}{2m}\left(\sum_{l=0}^{t} 2^{2l} a_l + \sum_{l=0}^{t}\sum_{h<l} 2^{l+h+1} R_{l,h} + \frac{{D_c}^2}{2} - D_1 D_c\right)\right) \tag{19}
$$

$$
\begin{aligned}
\text{s.t.} \quad & S_{i,j} \le Y_i \quad \forall (v_i, v_j) \in E_c && (20)\\
& S_{i,j} \le Y_j \quad \forall (v_i, v_j) \in E_c && (21)\\
& R_{l,h} \ge a_l + a_h - 1 \quad \forall l \le t,\ \forall h < l && (22)\\
& R_{l,h} \ge 0 \quad \forall l \le t,\ \forall h < l && (23)\\
& D_1 = \sum_{l=0}^{t} 2^l a_l && (24)\\
& D_1 = \sum_{v_i \in V_c} k_i Y_v && (25)\\
& Y_g = 0, \quad g = \arg\max\{k_i,\ \forall v_i \in V_c\} && (26)\\
& Y_i \in \{0,1\} \quad \forall v_i \in V_c && (27)\\
& a_l \in \{0,1\} \quad \forall l \le t. && (28)
\end{aligned}
$$

This model has $|V_c|+t+1$ binary variables, $|E_c|+\frac{t^2+t}{2}+1$ continuous variables and $2|E_c|+t^2+t+3$ constraints, while the initial model has $|V_c|$ binary variables, $2|E_c|+3$ continuous variables and $6|E_c|+3$ constraints.

5 Results

Table 1 presents the comparison of computing times for the initial model and the final one. Results have been obtained on a 2.4GHz Intel Xeon CPU of a computer with 24 GB RAM running Linux and CPLEX 12.2 [7]. M denotes the number of clusters, and Q the modularity; computing times are in seconds. Note that slight discrepancies may arise in the values of M and Q; they are due to the fact that optimal bipartitions are not necessarily unique. It appears that the computing time is reduced by a factor of 2 to over 265.

Network			Initial model			Compact model		
	n	m	M	Q	time	M	Q	time
Karate	34	78	4	0.4188	0.32	4	0.4188	**0.16**
Dolphins	62	159	4	0.5265	1.45	4	0.5265	**0.65**
Les misérables	77	254	8	0.5468	4.47	8	0.5468	**0.67**
A00 main	83	135	7	0.5281	0.71	7	0.5281	**0.37**
P53 protein	104	226	7	0.5284	16.82	7	0.5284	**1.55**
Political books	105	441	4	0.5263	16.74	5	0.5244	**2.66**
Football	115	613	10	0.6009	238.47	10	0.6009	**82.21**
A01 main	249	635	15	0.6288	563.41	15	0.6288	**38.12**
USAir97	332	2126	8	0.3596	113545.00	8	0.3596	**428.40**
Netscience main	379	914	20	0.8470	11.83	20	0.8470	**5.24**
S838	512	819	15	0.8166	24.48	15	0.8166	**6.40**
Power	4941	6594	40	0.9394	3952.72	41	0.9396	**567.07**

TAB. 1: Results obtained with the hierarchical divisive heuristic using respectively the original formulation and the compact reformulation.

References

[1] D. Aloise, S. Cafieri, G. Caporossi, P. Hansen, S. Perron and L. Liberti. Column generation algorithms for exact modularity maximization in networks. *Physical Review E*, 82(4), 046112, American Physical Society, 2010.

[2] A. Billionnet, S. Elloumi and A. Lambert. Extending the QCR method to general mixed-integer programs. *Mathematical Programming A*, doi:10.1007/s10107-010-0381-7, Springer, 2010.

[3] U. Brandes, D. Delling, M. Gaertler, R. Görke, M. Hoefer, Z. Nikoloski, and D. Wagner. On Modularity Clustering. *IEEE Transactions on Knowledge and Data Engineering*, 20(2):172-188, IEEE, 2008.

[4] S. Cafieri, P. Hansen, L. Liberti. Locally optimal heuristic for modularity maximization of networks. *Physical Review E*, 83(5):056105, American Physical Society, 2011.

[5] A. Clauset, M. E. J. Newman and C. Moore. Finding and evaluating community structure in very large networks. *Physical Review E*, 70(6), 066111, American Physical Society, 2004.

[6] M. Grötschel and Y. Wakabayashi. A cutting plane algorithm for a clustering problem. *Mathematical Programming*, 45(1), 59-96, Springer, 1989.

[7] IBM. ILOG CPLEX 12.2 *User's Manual*, IBM, 2010.

[8] M. E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences of the U.S.A.*, 103(23), 8577-8582, 2006.

[9] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69(2):026113, American Physical Society, 2004.

[10] G. Xu, S. Tsoka and L. G. Papageorgiou. Finding community structures in complex networks using mixed integer optimisation. *The European Physical Journal B - Condensed Matter and Complex Systems*, 60(2), 231-239, European Physical Society, 2007.