A Lagrangian Relaxation Algorithm for Modularity Maximization Problem

Inaba, Kotohumi
University of Tsukuba

Izunaga, Yoichi
University of Tsukuba

Yamamoto, Yoshitsugu
University of Tsukuba

出版情報：Operations research proceedings. 2014, pp.249-255, 2016-02-21. Springer
バージョン：
権利関係：

A Lagrangian Relaxation Algorithm for Modularity Maximization Problem

Kotohumi Inaba, Yoichi Izunaga and Yoshitsugu Yamamoto

Abstract The modularity proposed by Newman and Girvan is one of the most common measure when the nodes of a graph are grouped into communities consisting of tightly connected nodes. We formulate the modularity maximization problem as a set partitioning problem, and propose an algorithm based on the Lagrangian relaxation. To alleviate the computational burden, we use the column generation technique.

1 Introduction

As social network services grow, clustering on graphs has been attracting more attention. Since Newman and Girvan [5] proposed the modularity as a graph clustering measure, modularity maximization problem became one of the central subjects of research. Most of the solution methods proposed so far are heuristic algorithms due to its *NP*-hardness, which was shown by Brandes *et al.* [2], while few exact algorithms have been proposed.

Aloise *et al.* [1] formulated the problem as a set partitioning problem, which has to take into account all, exponentially many, nonempty subsets of the node set. Therefore one cannot secure the computational resource to hold the problem when the number of nodes is large. Their algorithm is based on the linear programming relaxation, and uses the column generation technique. Although it provides a tight upper bound of the optimal value, it can suffer a high degeneracy due to the set partitioning constraints.

Kotohumi Inaba
University of Tsukuba, Ibaraki 305-8573, Japan, e-mail: s1320486@sk.tsukuba.ac.jp

Yoichi Izunaga
University of Tsukuba, Ibaraki 305-8573, Japan, e-mail: s1130131@sk.tsukuba.ac.jp

Yoshitsugu Yamamoto
University of Tsukuba, Ibaraki 305-8573, Japan, e-mail: yamamoto@sk.tsukuba.ac.jp

In this paper, based on the set partitioning formulation, we propose a Lagrangian relaxation algorithm, and apply the column generation technique in order to alleviate the computational burden. We also report on some computational experiments.

2 Modularity Maximization Problem

Let $G=(V,E)$ be an undirected graph with the set $V=\{1,2,\ldots,n\}$ of n nodes and the set $E=\{1,2,\ldots,m\}$ of m edges. We say that $\Pi=\{C_1,C_2,\ldots,C_k\}$ is a *partition* of V if $V=\bigcup_{p=1}^{k}C_p$, $C_p\cap C_q=\emptyset$ for any distinct p and q, and $C_p\neq\emptyset$ for any p. Each member C_p of a partition is called a *community*. For $i,j\in V$ let e_{ij} be the (i,j) element of the adjacency matrix of graph G, and d_i be the degree of node i, and $\pi(i)$ be the index of community which node i belongs to, i.e., $\pi(i)=p$ means $i\in C_p$. Then *Modularity*, denoted by $Q(\Pi)$, of a partition Π is defined as

$$Q(\Pi)=\frac{1}{2m}\sum_{i\in V}\sum_{j\in V}\left(e_{ij}-\frac{d_id_j}{2m}\right)\delta(\pi(i),\pi(j)),$$

where δ is the Kronecker delta. *Modularity Maximization problem*, (MM) for short, is the problem of finding a partition of V that maximizes the modularity $Q(\Pi)$.

Let $\mathscr{P}$ denote the family of all nonempty subsets of V. Note that $\mathscr{P}$ is composed of 2^n-1 subsets of V. Introducing a binary variable z_C for each $C\in\mathscr{P}$, a partition Π is represented by the (2^n-1)-dimensional binary vector $z=(z_C)_{C\in\mathscr{P}}$ defined as

$$z_C=\begin{cases}1 & \text{when } C\in\Pi\\ 0 & \text{otherwise.}\end{cases}$$

For each $i\in V$ and $C\in\mathscr{P}$ we define a constant a_{iC} to describe whether node i belongs to C, i.e., $a_{iC}=1$ when $i\in C$ and $a_{iC}=0$ otherwise. The column $a_C=(a_{iC},\ldots,a_{nC})^\top$ is called the incidence vector of community C, i.e., $C=\{i\in V\mid a_{iC}=1\}$. For each $C\in\mathscr{P}$, let f_C be

$$f_C=\frac{1}{2m}\sum_{i\in V}\sum_{j\in V}w_{ij}a_{iC}a_{jC},$$

where $w_{ij}=(e_{ij}-d_id_j/2m)$. The constant f_C represents the contribution of community C to the objective function when C is selected as a member of the partition Π. Thus (MM) is formulated as the following integer programming (P):

$$(P)\quad\left|\begin{array}{ll}\text{maximize} & \displaystyle\sum_{C\in\mathscr{P}}f_Cz_C\\ \text{subject to} & \displaystyle\sum_{C\in\mathscr{P}}a_{iC}z_C=1\quad(\forall i\in V)\\ & z_C\in\{0,1\}\qquad(\forall C\in\mathscr{P})\end{array}\right.$$

We call the first set of constraints *set partitioning constraints*.

3 Lagrangian Relaxation and Lagrangian Dual Problem

The problem (P) is a difficult problem due to both its integrality and the set partitioning constraints. The well-known technique in order to obtain the useful information about the solution of (P) is *Linear Programming relaxation*, LP relaxation for short. Although LP relaxation provides a tight upper bound of the optimal value of (P), it usually suffers the high degeneracy due to the set partitioning constraints. To overcome this degeneracy, several techniques have been proposed in the literature, for example [3, 4]. In this paper we employ the *Lagrangian relaxation* instead of LP relaxation. Here we will give a brief review of Lagrangian relaxation and Lagrangian dual problem.

We relax the set partitioning constraints and add them to the objective function as a penalty with Lagrangian multiplier vector $\lambda = (\lambda_1, \ldots, \lambda_n)^\top$, and obtain the following Lagrangian relaxation problem $(LR(\lambda))$ with only the binary variable constraints:

$$(LR(\lambda)) \quad \left| \begin{array}{ll} \text{maximize} & \displaystyle\sum_{C \in \mathscr{P}} f_C z_C + \sum_{i \in V} \lambda_i (1 - \sum_{C \in \mathscr{P}} a_{iC} z_C) \\ \text{subject to} & z_C \in \{0,1\} \qquad (\forall C \in \mathscr{P}). \end{array} \right.$$

Let $\gamma_C(\lambda) = f_C - \sum_{i \in V} \lambda_i a_{iC}$, then the objective function of $(LR(\lambda))$ is written as

$$L(z, \lambda) = \sum_{C \in \mathscr{P}} \gamma_C(\lambda) z_C + \sum_{i \in V} \lambda_i.$$

For a given multiplier vector λ, we can obtain an optimal solution $z(\lambda)$ of $(LR(\lambda))$ by simply setting $z_C(\lambda) = 1$ if $\gamma_C(\lambda) > 0$, and $z_C(\lambda) = 0$ otherwise. We denote the optimal value of $(LR(\lambda))$ by $\omega(LR(\lambda))$, then $\omega(LR(\lambda))$ provides an upper bound of $\omega(P)$ for any λ. The problem of finding the best upper bound of $\omega(P)$ is called the Lagrangian dual problem (LRD), which is given as:

$$(LRD) \quad \left| \begin{array}{ll} \text{minimize} & \omega(LR(\lambda)) \\ \text{subject to} & \lambda \in \mathbb{R}^n. \end{array} \right.$$

One of the most commonly used method for this problem is the subgradient method. This method uses the subgradient $d(\lambda) = (d_i(\lambda))_{i \in V}$ at λ, defined by $d_i(\lambda) = 1 - \sum_{C \in \mathscr{P}} a_{iC} z_C(\lambda)$ for $i \in V$, and updates the Lagrangian multiplier vector to the direction of $d(\lambda)$ with a step size μ. We employ the well-known Armijo rule to determine the step size μ.

4 Proposed Algorithm

As we discussed in the previous section, the optimal solution $z(\lambda)$ can be obtained by checking the sign of $\gamma_C(\lambda)$. However it is hard to compute all of $\gamma_C(\lambda)$ owing to the huge number of variables. The number of variables which are positive at an optimal solution of (P) is at most the number of nodes, hence we need only a small number of variables. Therefore we use the column generation technique in order to alleviate the computation burden. Namely, we start the algorithm with a small number of variables and gradually add variables as the computation goes on.

We consider a small subfamily $\mathscr{C}$ of $\mathscr{P}$ and deal with the following subproblem $(P(\mathscr{C}))$:

$$(P(\mathscr{C})) \quad \left| \begin{array}{ll} \text{maximize} & \displaystyle\sum_{C\in\mathscr{C}} f_C z_C \\ \text{subject to} & \displaystyle\sum_{C\in\mathscr{C}} a_{iC} z_C = 1 \ (\forall i \in V) \\ & z_C \in \{0,1\} \quad (\forall C \in \mathscr{C}). \end{array} \right.$$

We denote the the Lagrangian relaxation problem and the Lagrangian dual problem corresponding to $(P(\mathscr{C}))$ by $(LR(\mathscr{C},\lambda))$ and $(LRD(\mathscr{C}))$, respectively. Let $\lambda(\mathscr{C})$ be an optimal solution of $(LRD(\mathscr{C}))$. Since the variables z_C for $C \in \mathscr{P}\setminus\mathscr{C}$ are not considered in the problem $(LR(\mathscr{C},\lambda(\mathscr{C})))$, the optimal solution $z(\lambda(\mathscr{C}))$ is not necessarily optimal to $(LR(\lambda(\mathscr{C})))$. When $\gamma_C(\lambda(\mathscr{C})) \le 0$ for all $C \in \mathscr{P}\setminus\mathscr{C}$, $z(\lambda(\mathscr{C}))$ is an optimal solution of $(LR(\mathscr{C},\lambda(\mathscr{C})))$. On the other hand $\gamma_C(\lambda(\mathscr{C})) > 0$ holds for some $C \in \mathscr{P}\setminus\mathscr{C}$, adding this C to $\mathscr{C}$ can lead to an improvement of the optimal value of $(LR(\mathscr{C},\lambda(\mathscr{C})))$, i.e., $\omega(LR(\mathscr{C}',\lambda(\mathscr{C}))) > \omega(LR(\mathscr{C},\lambda(\mathscr{C})))$ where $\mathscr{C}' = \mathscr{C}\cup\{C\}$. Note that $\lambda(\mathscr{C})$ is not necessarily an optimal solution of $(LRD(\mathscr{C}'))$, hence we solve the problem $(LRD(\mathscr{C}'))$ again to obtain an optimal Lagrangian multiplier $\lambda(\mathscr{C}')$ by the subgradient method.

According to the formulation of Xu *et al.* [6], the problem of finding C that maximizes $\gamma_C(\lambda)$ is formulated as the problem $(AP(\lambda))$ with a quadratic concave objective function:

$$(AP(\lambda)) \quad \left| \begin{array}{lll} \text{maximize} & \dfrac{1}{m}\displaystyle\sum_{r=1}^{m} x_r - \dfrac{1}{4m^2}\left(\sum_{i\in V} d_i y_i\right)^2 - \sum_{i\in V}\lambda_i y_i & \\ \text{subject to} & x_r \le y_i & (\forall r = \{i,j\} \in E) \\ & x_r \le y_j & (\forall r = \{i,j\} \in E) \\ & x_r \in \{0,1\} & (\forall r \in E) \\ & y_i \in \{0,1\} & (\forall i \in V). \end{array} \right.$$

For each edge $r = \{i,j\} \in E$, a binary variable x_r is equal to 1 when both end nodes i, j of edge r belong to the community that maximizes $\gamma_C(\lambda)$, and for each $i \in V$ a variables y_i is equal to 1 when node i belongs to the community and 0 otherwise.

From the above discussion, our proposed algorithm is given as follows.

Algorithm LCG

Step 1 : Let $\mathscr{C}$ and λ be an initial family of nonempty subsets of V and an initial multiplier vector, respectively.
Step 2 : Solve $(LRD(\mathscr{C}))$ to obtain a near optimal solution λ and the objective value $\omega(LRD(\mathscr{C}))$ by the subgradient method.
Step 3 : Solve $(AP(\lambda))$ and set y^* be an optimal solution.
Step 4 : If $\omega(AP(\lambda)) \leq 0$, then set $\mathscr{C}^* := \mathscr{C}$ and $\omega^* := \omega(LRD(\mathscr{C}))$.
Output $\mathscr{C}^*$ and ω^*, and terminate.
Step 5 : Otherwise set $C := \{i \in V \mid y_i^* = 1\}$ and increment $\mathscr{C} := \mathscr{C} \cup \{C\}$. Return to Step 2.

When this algorithm terminates, we construct the problem $(P(\mathscr{C}^*))$ from the obtained $\mathscr{C}^*$, and solve $(P(\mathscr{C}^*))$ by an IP solver.

The following proposition shows that we can obtain an upper bound of $\omega(P)$ at each iteration of the algorithm.

Proposition 1. *Let t be an upper bound of the number of communities at an optimal solution of (P). Then $\sum_{i\in V} \lambda_i + t \cdot \omega(AP(\lambda))$ is an upper bound of $\omega(P)$ for any $\lambda \in \mathbb{R}^n$.*

If the difference between the upper bound and $\omega(LRD(\mathscr{C}))$ is small, we can stop the algorithm even if $\omega(AP(\lambda)) \leq 0$ does not hold.

5 Computational Experiments

We report the computational experiment with Algorithm LCG. The experiment was performed on a PC with an Intel Core i7, 3.20 GHz processor and 12.0 GB of memory. We implemented the algorithm in Python 2.7, and used Gurobi 5.6.2 as the IP solver. We solved the benchmark instances provided by DIMACS. The size and the known optimal value of each instance is given in Table 1.

Table 1 Instances

name	n	m	$\omega(P)$
Karate	34	78	0.4198
Dolphins	62	159	0.5285
Football	115	613	0.6046

Table 2 Computational results of Algorithm LCG

instance	$\lvert\mathscr{C}^*\rvert$	ω^*	$\omega(P(\mathscr{C}^*))$	Gap (%)	Time (s)
Karate	62	0.4198	0.4198	0.000	7
Dolphins	112	0.5302	0.5222	1.192	37
Football	192	0.6054	0.6043	0.049	34509

We set $\mathscr{C}$ initially to the family of all singletons, i.e., $\mathscr{C} = \{\{1\}, \ldots, \{n\}\}$, and set an initial multiplier vector $\lambda = 0$. Table 2 shows the results of the proposed algorithm for each instance. The columns $|\mathscr{C}^*|$ and $\omega(P(\mathscr{C}^*))$ represent the cardinality of the final family of $\mathscr{C}^*$ and the optimal value of $(P(\mathscr{C}^*))$, respectively. The column Gap indicates relative gap defined by

$$\text{Gap} = \left(\frac{\omega(P) - \omega(P(\mathscr{C}^*))}{\omega(P)} \right) \times 100.$$

The column Time indicates the computation time in seconds.

From Table 2, we observe that Algorithm LCG solved Karate to optimality and failed to solve the others, but the Gap was less than 2%. Moreover the number of $|\mathscr{C}^*|$ is quite small.

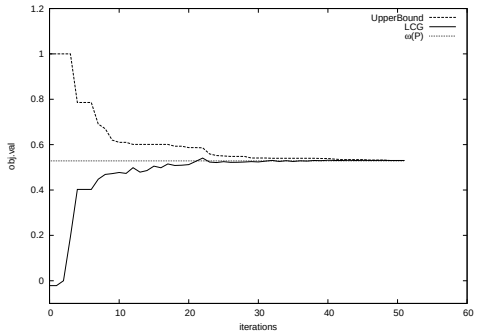

Fig. 1 $\omega(LRD(\mathscr{C}))$ vs. iterations for Dolphins

Fig. 1 shows $\omega(LRD(\mathscr{C}))$ and the upper bound in Proposition 1 at each iteration of the algorithm for the instance Dolphins. We set t to the optimal number of communities in calculating an upper bound of $\omega(P)$. $\omega(LRD(\mathscr{C}))$ rapidly increases at an early stage, and increases slowly as the algorithm goes on. Since we observed the similar results in other instances, we omitted the figures of the others.

References

1. D. Aloise, S. Cafieri, G. Caporossi, P. Hansen, L. Liberti, and S. Pellon, "Column generation algorithms for exact modularity maximization in networks," *Physical Review*, E.82, 2012.
2. 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, pp.172-188, 2008.
3. M. Boschetti, A.Minggozzi, and S.Ricciardelli, "A dual ascent procedure for the set partitioning problem," *Discrete Optimization*, 5, pp.735-747, 2008.
4. O. du Merle, D. Villeneuve, J. Desrosiers, and P. Hansen, "Stabilized column generation," *Discrete Mathematics*, 194, pp.229-237, 1999.
5. M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review*, E.69, 2004.
6. G.Xu, S.Tsoka and L.Papageorgiou, "Finding community structures in complex networks using mixed integer optimization," *The European Physical Journal*, B.50, pp.231-239, 2007.