Détection de communautés en ligne dans des graphes dynamiques

Yves Darmaillac [1] & Sébastien Loustau [2]

[1] *Laboratoire de Mathématiques et de leurs Applications - UMR CNRS 5142, Avenue de l'Université, 64013 Pau cedex, France*
Artfact-Online, Technopole Hélioparc, 2 Avenue du President Pierre Angot, 64000 Pau
email : yves.darmaillac@univ-pau.fr

[2] *Artfact-Online, Technopole Hélioparc, 2 Avenue du President Pierre Angot, 64000 Pau*
email : sebastien.loustau@learnation.eu

Résumé. Nous présentons un nouvel algorithme de détection de communautés qui maintient dynamiquement une structure de communautés dans un réseau de grande taille qui se modifie dans le temps. L'algorithme maximise l'indice de modularité grâce à une segmentation hiérarchique, obtenue par une méthode de Monte Carlo par Chaîne de Markov. Il est intéressant de voir l'algorithme comme une application dynamique de l'algorithme de Louvain (voir Blondel, Guillaume, Lambiotte et Lefebvre (2008)) où l'étape d'agrégation est remplacée par un modèle probabiliste hiérarchique.

Mots-clés. Algorithmes stochastiques, Apprentissage et classification, Grande dimension, Données massives, Statistique computationnelle.

Abstract. We introduce a novel algorithm of community detection that maintains dynamically a community structure of a large network that evolves with time. The algorithm maximizes the modularity index thanks to the construction of a randomized hierarchical clustering based on a Monte Carlo Markov Chain (MCMC) method. Interestingly, it could be seen as a dynamization of Louvain algorithm (see Blondel, Guillaume, Lambiotte et Lefebvre (2008)) where the aggregation step is replaced by the hierarchical instrumental probability.

Keywords. Stochastic algorithms, Learning and clustering, High dimension, Big data, Computational statistics.

1 Introduction

Community detection applications include social sciences, biology and complex systems, such as the world-wide-web, protein-protein interactions, or social networks (see Fortunato (2010) for a thorough exposition of the topic). To tackle this problem, spectral approaches have been introduced in Newman (2006). For large graphs, a class of algorithms that maximize a quality index called modularity was introduced by Newman and

Girvan (2004). Unfortunately, exact modularity optimization is NP-hard (see Brandes, Delling, Gaerther Gorke, Hoefer, Nikoloski and Wagner (2008)). An efficient solution for static graphs has been proposed by Blondel, Guillaume, Lambiotte et Lefebvre (2008) known as Louvain algorithm. Görke, Maillard, Staudt and Wagner (2010) proposed a greedy dynamic version of Louvain.

2 Notations and preliminary study

2.1 Notations

Let $G = (V, E)$ be an undirected and -possibly- weighted graph where V is the set of N vertices or nodes and E the set of edges (i, j), for $i, j \in \{1, \dots, N\}$. We denote by $A \in \mathcal{M}_N(\mathbb{R})$ the corresponding symmetric adjacency matrix where entry A_{ij} denotes the weight assigned to edge (i, j). The degree of a node i is denoted k_i and $m := |E| = \frac{1}{2}\sum_i k_i$. We call $C \in \mathcal{C}$ a *coloration* of graph (V, E) any partition $C = \{c_1, \dots, c_k\}$ of V where for any $i = 1, \dots, k$, $c_i \subseteq V$ is a set of nodes of G. Moreover, with a slight abuse of notation, $C(i) \in \{1, \dots, k\}$ denotes the community of vertex i based on partition C.

With these notations, the modularity $C \mapsto Q^C$ of a given graph (V, E) is given by :

$$Q^C = \frac{1}{2m} \sum_{i,j \in V^2} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta\big(C(i), C(j)\big) \tag{1}$$

where δ is the Kronecker delta. Roughly speaking, modularity compares fraction of edges that falls into communities of C with its expected counterpart, given a purely random rewiring of edges which respect to nodes degrees $(k_i)_{i \in V}$.

2.2 Metropolis Hasting Algorithm

Community detection algorithms based on modularity index try to maximize the so-called modularity. In this contribution, we use a MH algorithm as follows:

1. Initialization $\lambda > 0$, $C^{(0)}$.
2. For $k = 1, \dots, N$:
3. Draw $C' \sim p(\cdot|C^{(k-1)})$ where $p(\cdot|C^{(k-1)}) \in \mathcal{P}(\mathcal{N}^{C^{(k-1)}})$ is the proposal distribution over $\mathcal{N}^{C^{(k-1)}}$, a neighborhood of $C^{(k-1)}$.
4. Update $C^{(k)} = C'$ with acceptance ratio :

$$\rho = 1 \wedge \left(r_{C^{(k-1)} \to C'} \frac{\exp\left(\lambda Q^{C'}\right)}{\exp\left(\lambda Q^{C^{(k-1)}}\right)} \right), \text{ where } r_{C \to C'} := p(C^{(k-1)}|C')/p(C'|C^{(k-1)}). \tag{2}$$

The above algorithm satisfies the so-called detailed balance condition for any proposal p and then produces a Markov chain with invariant probability density f such that:

$$f(C)dC \approx \exp\left(\lambda Q^C\right) dC.$$

The major issue is then to define a particular neighborhood $\mathcal{N}^C$ and a relevant proposal $p(\cdot|C)$ in order to achieve convergence in a manageable time. An important issue is to tune parameter $\lambda > 0$ in MH algorithm. In our experimental studies, we use a value of λ of order $m^{-1/2}$. Adaptive choices of λ have been investigated in the literature (see for instance Cesa-Bianci and Lugosi (2006)).

2.3 Neighborhood definition

In what follows, given $C \in \mathcal{C}$, the neighborhood $\mathcal{N}^C$ consists of all coloration C' equals to C except for one node $i \in V$. Then two cases arises:

- i joins an existing community $c \in C$ such that $c \neq C(i)$,
- a new single node community c_{new} is created by i.

2.4 Proposal distribution

The prior $p(\cdot|C)$ is defined as :

$$p(\cdot|C) = \alpha p_1(\cdot|C) + (1-\alpha)\, p_2(\cdot|C), \tag{3}$$

where $\alpha \in (0, 1)$ and $p_1(\cdot|C)$ and $p_2(\cdot|C)$ are defined as follows :

1. $p_1(\cdot|C)$ is equivalent to draw a first node i uniformly over V and a second one j uniformly among the others,
2. $p_2(\cdot|C)$ is equivalent to draw a first node i uniformly over $\mathcal{F}^C$ and a second one j proportionally to $k^C_{i,C(j)}$ with the constraint $C(i) \neq C(j)$.

In both cases, to derive C', we use the mapping Φ^C and state $C' = \Phi^C(i, C(j))$.

2.5 Modularity gain

Last step is to compute the likelihood in (2). For this purpose, we introduce the quantity:

$$\Delta Q^{C \to C'} := Q^{C'} - Q^C.$$

It is easy to see from (1) that when an isolated node i joins an existing community $c \in C$, we have:

$$\Delta Q^{C \to C'} = \Delta Q_+^{C \to C'} := \frac{1}{m} \sum_{j \in V} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(C(j), c) = \frac{1}{m} \left(k^C_{i,c} - \frac{k_i k^C_c}{2m}, \right) \tag{4}$$

where $k_c^C = \sum_{j \in V} k_j \delta(C(j), c)$ is the total weight of community $c \in C$.

Symmetrically, when a node i leaves its community $C(i)$ to form a new single node community:

$$\Delta Q^{C \to C'} = \Delta Q_-^{C \to C'} := -\frac{1}{m} \sum_{j \in V} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(C(j), C(i))$$
$$= -\frac{1}{m} \left(k_{i,C(i)}^C - A_{ii} - \frac{k_i}{2m} (k_{C(i)}^C - k_i) \right) \tag{5}$$

Note that in (1) every term is summed twice due to the symmetry of the adjacency matrix. This explains the $1/m$ factor instead of $1/2m$ in (4) and (5).

Next, the change of modularity incurred by our proposal is :

$$\Delta Q^{C \to C'} = \begin{cases} \Delta Q_-^{C \to C'} & \text{if } C' = \Phi^C(i, C(i)) \\ \Delta Q_-^{C \to C'} + \Delta Q_+^{C \to C'} & \text{otherwise,} \end{cases} \tag{6}$$

3 Aggregation

Aggregation may accelerate convergence when communities are "big", i.e. when there is notably less communities than nodes. Aggregation consists of building a new graph which nodes are the communities of the former graph, after the optimization step as described in Section 2.2.

In our framework, we propose to use the same kind of prior than (3) to a family of aggregated graphs in order to move entire communities rather than a single node. To achieve this, we introduce a set of hierarchical graphs and hierarchical priors as follows.

Let $L \geq 1$ an integer. The construction of a family of aggregated graphs $(E_l, V_l)_{l=1}^L$ is done iteratively as follows. Let $G_1 := (E_1, V_1) = (E, V)$. Then, given G_l for $1 \leq l \leq L - 1$, we define a $(l+1)^{\text{th}}$ aggregated graph as $G_{l+1} = (E_{l+1}, V_{l+1})$ where $V_{l+1} := \{v_1^{(l+1)}, \ldots, v_{N_{l+1}}^{(l+1)}\}$ is a partition of V_l and E_{l+1} is computed thanks to G_l as the following aggregation step:

1. if $i \neq j$, the edge between $v_i^{(l+1)}$ and $v_j^{(l+1)}$ in G_{l+1} is equal to the sum of all edges between nodes of G_l contained in $v_i^{(l+1)}$ and nodes of G_l contained in $v_j^{(l+1)}$,
2. if $i = j$, loop i in E_{l+1} is equal to the sum of all edges between nodes of G_l contained in $v_i^{(l+1)}$.

Moreover, We denote by $A^{(l)} \in \mathcal{M}_{|V_l|}(\mathbb{R})$ the corresponding symmetric adjacency matrix where entry $A_{ij}^{(l)}$ denotes the weight assigned between vertices $v_i^{(l)}$ and $v_j^{(l)}$ in G_l. The degree of a node i is denoted $k_i^{(l)}$ and $m_l := |E_l| = \frac{1}{2} \sum_i k_i^{(l)}$. We call $C_l \in \mathcal{C}_l$ a *coloration* of level l of graph G_l any partition $C_l = \{c_{1,l}, \ldots, c_{k,l}\}$ of V_l where for any $i = 1, \ldots, k$, $c_{i,l} \subseteq V_l$ is a set of nodes of G_l. Moreover, $C_l(i)$ denotes the community of vertex i based on C_l. Moreover, we denote by $\mathrm{map}^{C_l} : V^{(l)} \to V^{(l+1)}$ the mapping of all nodes of G_l in G_{l+1} that groups all nodes of a same community according to C_l in a single node of G_{l+1}. For instance, if for some i we have $c_{i,l} = \{v_1^{(l)}, \ldots, v_r^{(l)}\}$, then $\mathrm{map}^{C_l}(v) = \mathrm{map}^{C_l}(v')$ for any $v, v' \in c_{i,l}$.

Finally, the decision to find the community of $i \in V$ thanks to $(C_l, G_l)_{l=1}^L$ is made of the following computation:

$$C(i) := C_L \left(\mathrm{map}^{C_{L-1}} \circ \cdots \circ \mathrm{map}^{C_1}(i)\right),$$

where $C_L(v)$ stands for the community of $v \in V^{(L)}$.

4 Dynamic Metropolis Hasting graph clustering

The purpose of this section is to adapt the previous algorithm to the dynamic graph clustering problem. The challenge is to maintain a clustering for a sequence of graphs $(G_t)_{t\geq 1}$, where G_t is derived from G_{t-1} by applying a small number of local changes.

The following algorithm describes the online procedure. Grossly speaking, the principle of the algorithm is to run at each new observation t the MH iterations from the endpoint of step $t-1$ at each level $l = 1, \ldots, L$ simultaneously.

---

1. Initialization $\lambda > 0$, $L \geq 1$, $(G_l^{(0,0)}, C_l^{(0,0)})_{l=1}^L$,$N(0) = 0$.
2. For $t = 1, \ldots, T$:
3. $(C^{(t,0)}, G^{(t,0)}) := (C^{(t-1,N(t-1))}, G^{(t-1,N(t-1))})$
4. For $k = 1, \ldots, N(t)$:
5. Draw $C' \sim p$ where $p(\cdot | C_l^{(t,k-1)}, G_l^{(t)}) \in \mathcal{P}(\otimes_{l=1}^L \mathcal{C}_l)$.
6. If C' has been proposed by the l^{th} prior $p^{(l)}$ for some $l = 1, \ldots, L$ update $C^{(t,k)} = C'$ with Metropolis ratio :
$$\rho = 1 \wedge \left(r_{C^{(t,k-1)} \to C'} \frac{\exp\left(\lambda Q^{C'_l}\right)}{\exp\left(\lambda Q^{C_l^{(t,k-1)}}\right)} \right), \text{ where } r_{C^{(t,k-1)} \to C'} := p(C_l^{(t,k-1)} | C'_l)/p(C'_l | C_l^{(t,k-1)}). \tag{7}$$
7. If C' has been accepted and has used the l^{th} prior $p^{(l)}$ for some $l = 1, \to, L$, maintain $(G_{k'})_{k'=l+1}^L$ as follows:
8. For $k' = l, \ldots, L-1$
9. Update $V_{k'+1}$ thanks to $\mathrm{map}^{C_{k'}^{(t,k)}}$,
10. Update $E_{k'+1}$ thanks to the aggregation step define above.

Bibliographie

[1] M. EJ Newman and M. Girvan (2004), Finding and evaluating community structure in networks, Physical review E, 69(2):026113.

[2] M. EJ Newman (2006), Finding community structure in networks using the eigenvectors of matrices, Phys. Rev. E (3), 74(3):036104, 19.

[3] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner (2008). On modularity clustering. IEEE Trans. on Knowl. and Data Eng., 20(2):172-188.

[4] V. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre (2008), Fast unfolding of communities in large networks, Journal of statistical mechanics: theory and experiment, 2008(10):P10008.

[5] R. Görke, P. Maillard, C. Staudt, and D. Wagner (2010), Modularity driven clustering of dynamic graphs, in Proceedings of the 9th International Conference on Experimental Algorithms, Springer-Verlag, SEA'10, pages 436-448.

[6] S. Fortunato (2010), Community detection in graphs. Physics Reports, 486 (3):75-174.

[7] N. Cesa-Bianci and G. Lugosi (2006), Prediction, learning, and games. Cambridge university press.