Algorithmique des réseaux sociaux **2013/2014**

Cours 4 — 4 Novembre

Enseignant: Marc Lelarge *Scribe: Marc Heinrich*

Pour information

- Page web du cours
  `http://www.di.ens.fr/~lelarge/soc.html`

4.1 Partition de graphe et Laplacien

Problème : On cherche à partitionner l'ensemble des sommets en groupes de telle sorte que le nombre d'arête entre chaque groupe soit minimisé. Si on se fixe un nombre k de groupe, alors il existe un algorithme pour résoudre ce problème en :

$$\mathcal{O}(n^{k^2})$$

Par contre, si k est donné en entrée, alors c'est un prolbème NP-complet. Si on spécifie k sommets, et que l'on demande une k-coupe séparant ces sommets, c'est également un problème NP-complet.

4.1.1 Méthode spectrale

On note $\mathbf{A}$ la matrice d'adjacence du graphe définie par :

$$\mathbf{A}_{i,j} = \begin{cases} 1 & si\ (i,j) \in E \\ 0 & sinon \end{cases}$$

On se restreint au cas où le graphe est non orienté, dans ce cas, la matrice $\mathbf{A}$ est symétrique. On considère le cas de deux communautés. Le nombre d'arêtes entre deux groupes est alors donné par la formule :

$$R = \frac{1}{2} \sum_{i,j \neq groupes} \mathbf{A}_{i,j}$$

Il s'agit d'essayer de minimiser cette quantité. On note alors $\mathbf{s}$ le vecteur donné par :

$$\mathbf{s}_i = \begin{cases} +1 & \text{si } i \text{ est dans le groupe 1} \\ -1 & \text{si } i \text{ est dans le groupe 2} \end{cases}$$

On a :

$$\begin{aligned}
\mathbf{s}\mathbf{s}^t &= n \\
\frac{1}{2}\left(1 - \mathbf{s}_i\mathbf{s}_j\right) &= \begin{cases} 1 & \text{si } i,j \text{ sont dans des groupes différents} \\ 0 & \text{sinon} \end{cases} \\
R &= \frac{1}{4}\sum_{i,j}(1 - \mathbf{s}_i\mathbf{s}_j)\mathbf{A}_{i,j} \\
&= \frac{1}{4}\sum_{i,j} s_i(d(i)\delta_{i,j} - \mathbf{A}_{i,j})s_j \\
&= \frac{1}{4}\mathbf{s}^t\mathbf{L}\mathbf{s}
\end{aligned}$$

Avec $d(i) = \sum_j A_{ij}$ le degré du sommet i et $\mathbf{L}$ la matrice donnée par $\mathbf{L}_{i,j} = d(i)\delta_{i,j} - \mathbf{A}_{i,j}$

$\mathbf{L}$ est une matrice symétrique, appelée matrice Laplacienne, et on a :

$$\mathbf{L}_{i,j} = \begin{cases} d(i) & si\ i = j \\ -1 & si\ i \neq j\ et (i,j) \in E \\ 0 & sinon \end{cases}$$

Comme :

$$\begin{aligned}
\forall \mathbf{x},\ \mathbf{x}^t\mathbf{L}\mathbf{x} &= \sum_{i,j} \mathbf{x}_i\mathbf{L}_{i,j}\mathbf{x}_j \\
&= \sum_i \mathbf{x}_i^2 d(i) - \sum_{i\neq j,\ (i,j)\in E} \mathbf{x}_i\mathbf{x}_j \\
&= \sum_{i<j,\ (i,j)\in E} (\mathbf{x}_i - \mathbf{x}_j)^2 \\
&\geq 0
\end{aligned}$$

Donc le spectre de $\mathbf{L}$ est inclus dans $\mathbf{R}_+$. On note alors $0 \leq \lambda_1 \leq \cdots \leq \lambda_n$ les valeurs propres de la matrice $\mathbf{L}$, et $\mathbf{v}_1, \cdots \mathbf{v}_n$ les vecteurs propres orthonormés associés. On décompose $\mathbf{s}$ dans la base orthonormée $(\mathbf{v}_i)_i$:

$$\begin{aligned}
\mathbf{s} &= \sum_{i=1}^{n} a_i\mathbf{v}_i \qquad avec\ a_i = \mathbf{v}_i^t\mathbf{s} \\
\sum_{i=1}^{n} a_i^2 &= n \qquad \text{du fait de la normalisation } \mathbf{s}^t\mathbf{s} = n
\end{aligned} \tag{4.1}$$

Avec ces notations, on a alors :

$$\begin{aligned} 4R &= \sum_i a_i \mathbf{v}_i^t \mathbf{L} \sum_j a_j \mathbf{v}_j \\ &= \sum_{i,j} a_i a_j \lambda_j \delta_{i,j} \\ &= \sum_i \lambda_i a_i^2 \end{aligned}$$

On veut donc minimiser R sous la contrainte (4.1).

On a $\lambda_1 = 0$ avec pour vecteur propre $\mathbf{v}_1 = \begin{pmatrix} 1 & \cdots & 1 \end{pmatrix}^t$. Plus précisément, le calucl précédent de $\mathbf{x}^t\mathbf{L}\mathbf{x}$ nous permet de voir que 0 est valeur propre avec une multiplicité égale au nombre de composantes connexes du graphe.

La plus petite valeur propre non nulle (*i.e.*λ_2 si le graphe est connecté) est appelée la **connectivité algébrique**.

Commentaire: Si on prend $\mathbf{s} = \begin{pmatrix} 1 & \cdots & 1 \end{pmatrix}^t$, alors $R = 0$ et la coupe que l'on obtient est la coupe triviale.

Pour résoudre ce problème, on fixe les tailles des groupes n_1 et $n_2 = n - n_1$. Dans ce cas, on a :

$$\left(\mathbf{v}_1^t \mathbf{s}\right)^2 = \frac{(n_1 - n_2)^2}{n}$$

En posant $\mathbf{s}' = \mathbf{s} - a_1\mathbf{v}_1$, pour minimiser R, il faudrait prendre $\mathbf{s}'$ colinéaire à $\mathbf{v}_2$ (appelé parfois le vecteur de Fiedler). Ceci n'est généralement pas possible à cause des contraintes entières sur les composantes de $\mathbf{s}$.

4.1.2 Heuristique

Nous présentons maintenant une heuristique pour résoudre le problème de minimisation de R.

$$\begin{aligned} |\mathbf{v}_2^t \mathbf{s}| &= \left| \sum_i \mathbf{v}_i^{(2)} \mathbf{s}_i \right| \\ &\leq \sum_i \left| \mathbf{v}_i^{(2)} \right| \end{aligned}$$

Avec égalité lorsque tous les termes de la somme ont le même signe. Cela suggère de prendre :

$$\mathbf{s} = \begin{cases} +1 & si\ \mathbf{v}_2^{(i)} \geq 0 \\ -1 & sinon \end{cases} \tag{4.2}$$

Cependant, on ne respecte pas les contraintes sur n_1 et n_2 que l'on s'est fixé plus haut. La solution est alors d'ordonner les composantes de $\mathbf{v}_2$ par ordre croissant, et au choix de prendre les n_1 plus grandes pour le groupe 1, ou les n_1 plus petites. Cela donne deux coupes possibles, il suffit alors de les comparer et de prendre la meilleur.
Commentaire: Le problème de cette méthode est qu'à priori, on ne connait pas la taille des communautés que l'on recherche. Une solution est de prendre directement pour vecteur **s** celui donné par (**??**).

4.2 Modularité et optimisation spectrale

4.2.1 Modularité

On va présenter une autre méthode pour résoudre notre problème, qui cette fois respecte en plus les spécifications suivantes :
- ne pas imposer de tailles des communautés à l'avance
- ne pas biaiser vers les plus forts degrés.

Pour cela, on introduit une nouvelle notion, la **modularité** définie par :

$$Q = \left(\begin{array}{c}\text{nombre d'arêtes}\\ \text{dans les communautés}\end{array}\right) - \left(\begin{array}{c}\text{nombre moyen}\\ \text{de telles arêtes}\end{array}\right)$$

Il s'agit maintenant de maximiser la modularité, c'est à dire de trouver les sous-graphes pour lesquels on a beaucoup plus d'arêtes que ce que l'on attendait. La notion de nombre moyen d'arêtes nécessite que l'on choisisse un modèle statistique pour le graphe. Si tout ce que l'on connait sur le graphe est la suite des degrés, alors on prend comme modèle un multigraphe tiré alétoirement parmi tous les graphes qui possèdent cette suite de degrés. Pour chaque sommet, on trace toutes les demi-arêtes qui partent de ce sommet. On peut alors contruire le multi-graphe en choisissant un appariement des demi-arêtes aléatoirement et uniformément. On peut également choisir successivement le sommet opposé à une demi-arête uniformément parmi ceux qui sont possibles. Cette deuxième manière de faire permet d'obtenir un graphe ayant la même distribution.

Avec ce deuxième modèle, il est facile de voir que la probabilité qu'une demi-arête u fixée soit connectée avec le sommet $v \neq u$ est : $\frac{d(v)}{2m-1}$. Par linéarité, le nombre moyen d'arrêtes entre les sommets u et $v \neq u$ est alors :

$$\frac{d(u)d(v)}{2m-1}$$

Un calcul similaire montre que le nombre moyen de boucles est $\frac{d(v)(d(v)-1)}{2m-1}$. Pour une communauté fixée S, le nombre moyen d'arêtes internes est alors :

$$\frac{1}{2}\frac{d(\mathcal{S})(d(\mathcal{S})-1)}{2m-1} \simeq \frac{d(\mathcal{S})^2}{4m} \qquad \text{avec } d(\mathcal{S}) = \sum_{v\in\mathcal{S}} d(v)$$

Pour une partition $\mathcal{P} = \{\mathcal{S}_1, \cdots, \mathcal{S}_k\}$, on défini la modularité par :

$$Q(\mathcal{P}) = \frac{1}{m}\sum_{i=1}^{k} |e(\mathcal{S}_i)| - \frac{1}{4m} d(\mathcal{S}_i)^2$$

Problème d'optimisation :
Il faut trouver une partition $\mathcal{P}$ qui maximise $Q(\mathcal{P})$ (k n'est pas spécifié). C'est un problème NP-difficile, et il n'existe pas d'algorithme avec des guaranties démontrées.

4.2.2 Cas de deux communautés

On reprend la notation précédente pour le vecteur $\mathbf{s}$. On a :

$$\begin{aligned} \frac{1}{2}\left(\mathbf{s}_i\mathbf{s}_j + 1\right) &= \begin{cases} 1 & \text{si } (i,j) \text{ sont dans le même groupe} \\ 0 & \textit{sinon} \end{cases} \\ |e(\mathbf{s}_1)| + |e(\mathbf{s}_2)| &= \sum_{i,j} \mathbf{A}_{i,j}\frac{1}{2}\left(\mathbf{s}_i\mathbf{s}_j + 1\right) \end{aligned}$$

$$d(\mathbf{s}_1)^2 + d(\mathbf{s}_2)^2 = \sum_{i,j} d(i)d(j)\frac{1}{2}\left(\mathbf{s}_i\mathbf{s}_j + 1\right)$$

Il en résulte, en utilisant le fait que $\sum_{i,j}\frac{d(i)d(j)}{2m} = 2m = \sum_{i,j}\mathbf{A}_{i,j}$

$$\begin{aligned} Q &= \frac{1}{4m}\sum_{i,j}\left(\mathbf{A}_{i,j} - \frac{d(i)d(j)}{2m}\right) \\ &= \frac{1}{4m}\mathbf{s}^t\mathbf{B}\mathbf{s} \end{aligned}$$

avec $\mathbf{B}$ une matrice symétrique définie par $\mathbf{B}_{i,j} = \mathbf{A}_{i,j} - \frac{d(i)d(j)}{2m}$. $\mathbf{B}$ est appelé la matrice de modularité, et 0 est valeur propre de $\mathbf{B}$ associée au vecteur propre $\left(\ 1, \ \cdots \ , 1\ \right)$
On note :

$$\beta_1 \geq \cdots \geq \beta_n \quad et \quad \mathbf{u}_1, \cdots \mathbf{u}_n$$

les valeurs propres, avec leurs vecteurs propres associés. Alors $Q = \sum_i a_i^2\beta_i$

Heuristique : prendre $\mathbf{s}_i = \begin{cases} 1 & si\ \mathbf{u}_1^{(i)} \geq 0 \\ -1 & sinon \end{cases}$

4.2.3 Cas de plusieurs communautés

On définit cette fois ci une matrice $n \times c$, avec c le nombre de communautés par :

$$\begin{aligned} \mathbf{S} &= (\mathbf{s}_1, \cdots \mathbf{s}_c) \\ \mathbf{S}_{i,j} &= \begin{cases} 1 & i \text{ est dans le groupe } j \\ 0 & sinon \end{cases} \\ \mathbf{s}_i \mathbf{s}_j &= 0 \quad si\ i \neq j \\ Tr(\mathbf{S}^t\mathbf{S}) &= n \end{aligned}$$

On a maintenant (en ignorant la normalisation en $\frac{1}{2m}$ faite dans le cas de deux communautés) :

$$\begin{aligned} Q &= \sum_{i,j=1}^{n} \sum_{k=1}^{c} \mathbf{B}_{i,j}\mathbf{S}_{i,k}\mathbf{S}_{k,j} \\ &= Tr(\mathbf{S}^t\mathbf{B}\mathbf{S}) \end{aligned}$$

En prenant $\mathbf{B} = \mathbf{U}\mathbf{D}\mathbf{U}^t$ avec $\mathbf{D}$ une matrice diagonale, on est ramené comme ci-dessus à essayer de maximiser une quantité de la forme :

$$Q = \sum_{i=1}^{n} \sum_{k=1}^{c} \beta_i (\mathbf{u}_i^t \mathbf{s}_k)^2$$

où on peut maintenant choisir le paramètre c.