Détection de communautés dans les grands graphes de terrain

Rushed Kanawati

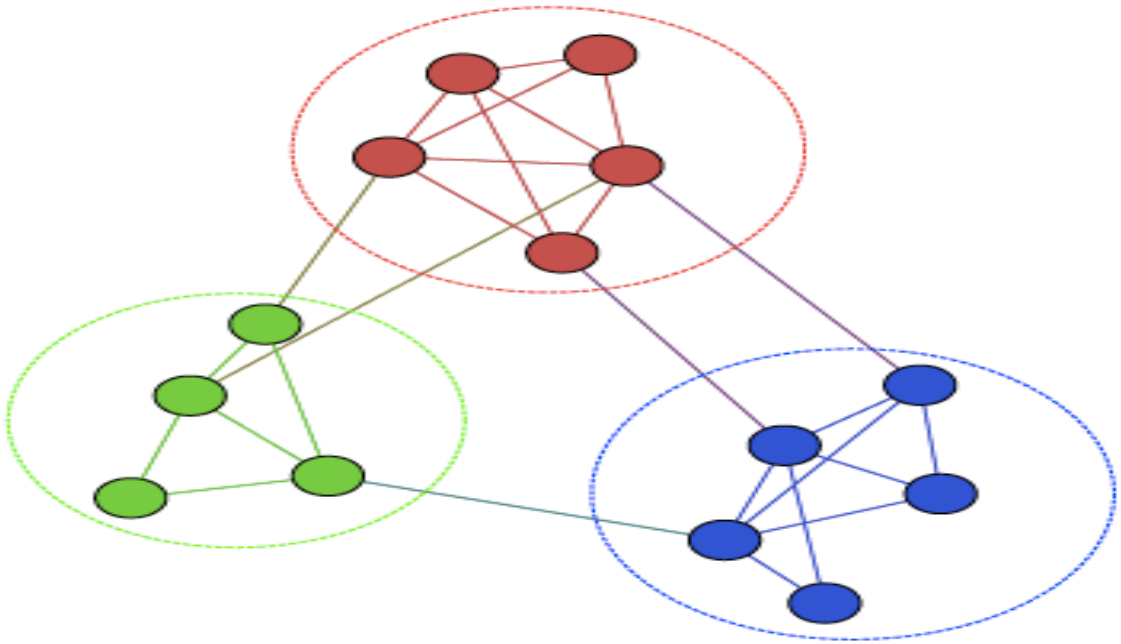
LIPN, CNRS UMR 7030
Université Paris Sorbonne Cité
http://kanawati.fr
kanawati@univ-paris13.fr

Plan

Communauté ?

Définition

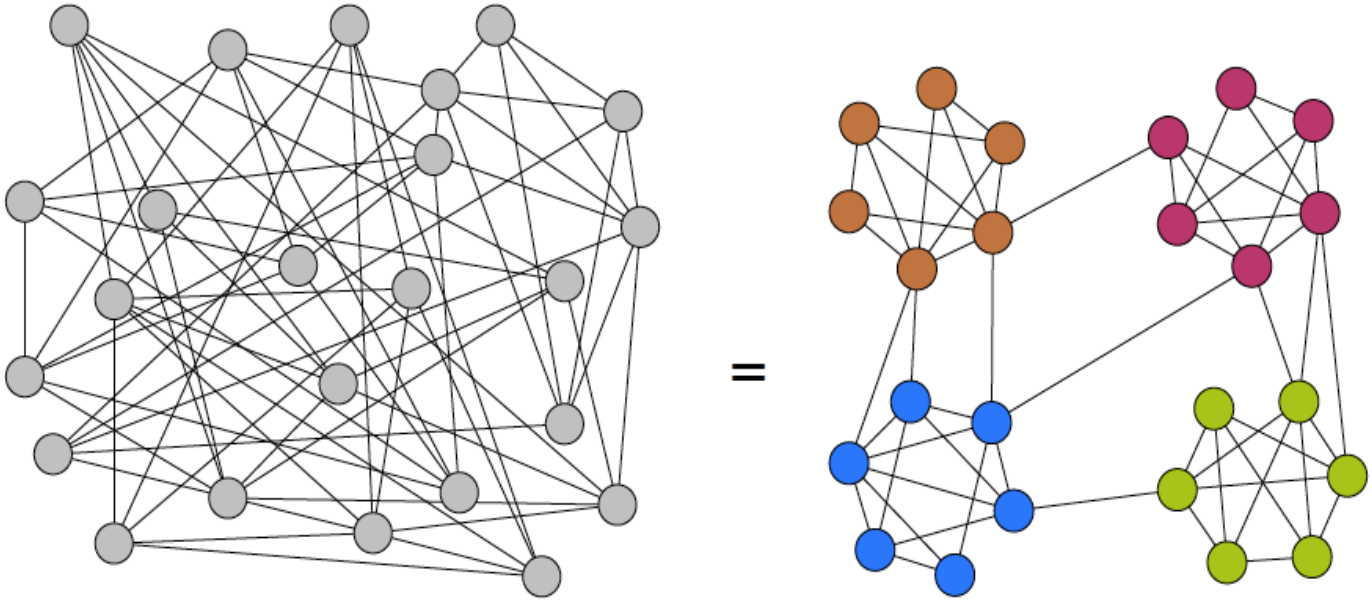
Un sous-graphe dense faiblement lié au reste du graphe.



Communauté ?

Definition

Un sous-graphe dense faiblement lié au reste du graphe.



Intérêts

- Réseaux sociaux : identification de groupes d'amis
- Web : ensemble de pages web traitant un même thème,
- Réseaux biologiques : un ensemble de gènes dédiés à une même fonction,
- etc.

Application

- Parallélisation
- Visualisation
- Compression

Problèmes

- **Identification de communauté** : délimiter la communauté d'un nœud.
- **Partitionnement du graphe** en N sous-graphes (communautés non chevauchantes).
- **Detection de communautés** (potentiellement chevauchantes)

Fonction de qualité : exemples I

La modularité locale R [Clauset05]

$$R = \frac{B_{in}}{B_{in}+B_{out}}$$

B_{in}=len(g.es.select(_within=B))

B_{out}=len(g.es.select(_between=(B,S)))

Fonction de qualité : exemples II

La modularité locale M

$$M = \frac{D_{in}}{D_{out}}$$

D_{in}=len(g.es.select(_within=D))

D_{out}=len(g.es.select(_between=(B,S)))

Fonction de qualité : exemples III

La modularité locale L

$L = \frac{L_{in}}{L_{ex}}$ où :

$$L_{in} = \frac{\sum_{i \in D} \|\Gamma(i) \cap D\|}{\|D\|}$$

$$L_{ex} = \frac{\sum_{i \in B} \|\Gamma(i) \cap S\|}{\|B\|}$$

Approches

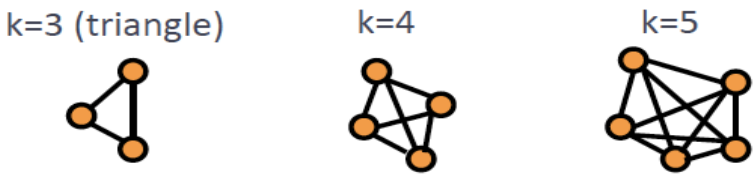
- *Approches centrées groupes* où des nœuds sont regroupés en communautés en fonction de propriétés topologiques partagées.
- *Approches centrées réseau* où la structure globale du réseau est examinée pour la décomposition du graphe en communautés.
- *Approches centrées propagation* qui appliquent souvent une procédure d'émergence de la structure communautaire par échange de messages entre nœuds voisins.
- *Approches centrées graines* où la structure communautaire est construite autour d'un ensemble de nœuds choisis d'une manière informée.

Approches centrées groupes

- Recherche de groupes denses :
  - Cliques maximales
  - γ-dense quasi clique.
  - K-core : sous-graphe connexe maximal dans lequel le degré de chaque nœud est supérieur ou égale à k
  - ...
- Adéquat pour graphes denses
- Souvent NP-Complet.

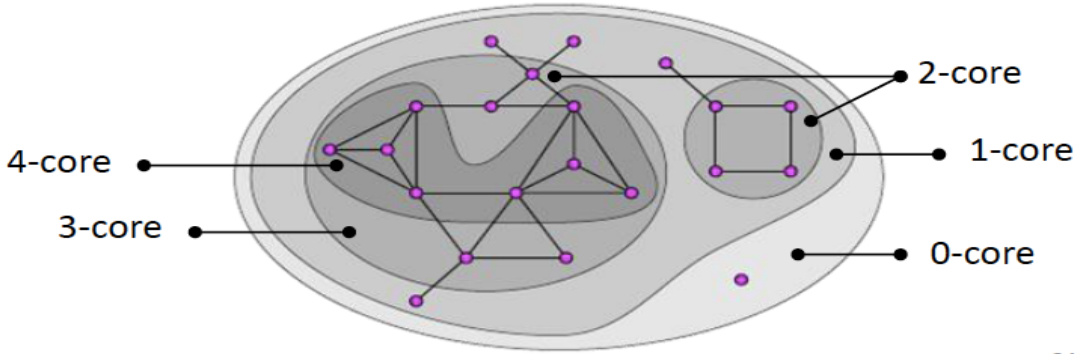
Group-based approches

- k-clique



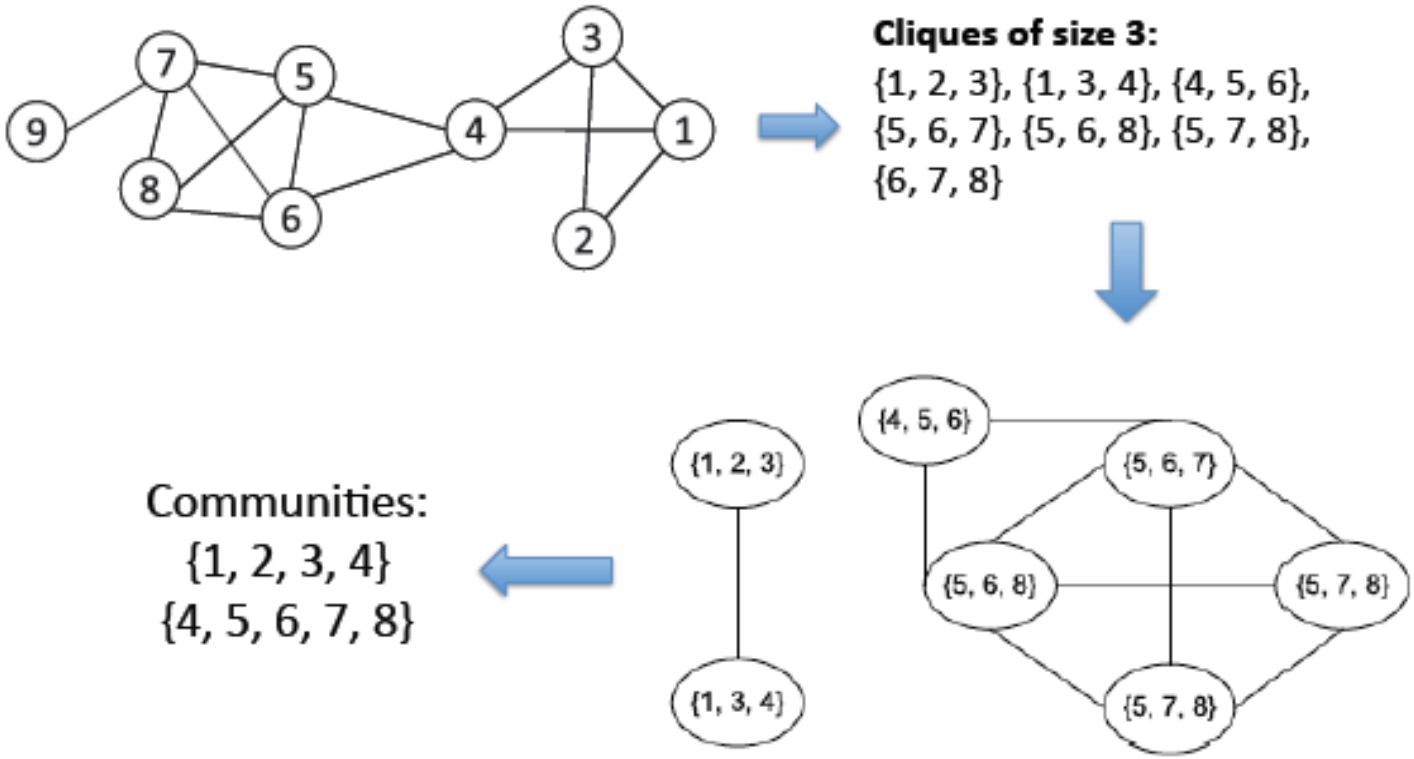
- N-clique

- k-core



from Symeon Papadopoulos, Community Detection in Social Media, CERTH-ITI, 22 June 2011

Example: Clique percolation



Suits fairly dense graph.

Approches centrées Réseau

- Algorithme de clustering
- Approches d'optimisation
- Approches basées sur la propagation locale
- Approches centrées *graine*

Approche de clustering : Classification hiérarchique

Chaque sommet = communauté.

On itère jusqu'au avoir une seule communauté :

> Calcule les **distances** entre chaque 2 communautés
> Fusionner les deux communautés les plus proches

On obtient un dendrogramme de communautés.

Distance entre communautés : La distance minimale, maximale, ou moyenne entre deux sommets des 2 communautés. Distance entre barycentres des communautés.

Approches d'optimisation

- Définition d'une fonction de qualité de partition
- Application d'approches d'optimisation.

Fonction de qualité : La modularité

La modularité

$$Q(\mathcal{P}) = \frac{1}{2m} \sum_{c \in \mathcal{P}} \sum_{i,j \in c} (A_{ij} - \frac{d_i d_j}{2m}) \quad (1)$$

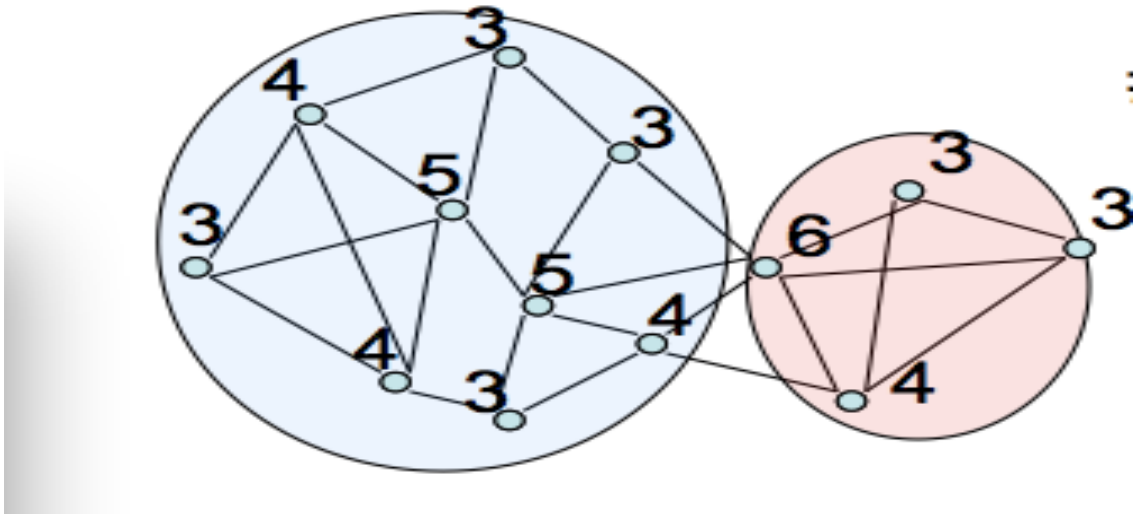


Figure: Exemple de calcul de la modularité : $Q = \frac{(15+6)-(11.25+2.56)}{25} = 0.275$

Optimisation : Algorithmique génétique

Une population de solutions : chaque individu est représenté par une séquence de gènes

Evolution de la population par mécanismes de : **croisement**, **mutation** et **sélection naturelle**.

Gène : vecteur d'affectation de communauté aux sommets.

Mécanismes de sélection : Modularité.

Intérêt : Parrallélisation facile.

Approches séparatives

`igraph.Graph.community_edge_betweenness`

Principe : A chaque itération retirer un lien *inter-communuaté*

Différents critères pour identifier un lien inter-communauté

Algo. de Girvan et Newman :*la centralité d'intermédiarité*. Complexité $\mathcal{O}(m^2 n)$.

Algo. de Fortunato : *centralité d'information*. L'efficacité d'un graph E est donné par la moyenne de $\frac{1}{d_{ij}}$, La centralité d'une arête (i, j) est donné par la diminution de E en retirant ce lien. Complexité $\mathcal{O}(m^3 n)$.

Approche Agglomérative I

`igraph.community_multilevel`

Chaque nœud est associé à une communauté

Répéter jusqu'à stabilisation :

pour chaque nœud i évaluer le gain de modularité ΔQ si i rejoint la communauté d'un noœud voisin j

$\Delta Q = (\frac{\sum_{in}+2k_{i,in}}{2m} - (\frac{\sum_{tot}+k_i}{2m})^2) - (\frac{\sum_{in}}{2m} - (\frac{\sum_{tot}}{2m})^2 - (\frac{k_i}{2m})^2)$

$\sum_{in}$ est la somme des poids des liens dans la communauté cible,

Approche Agglomérative II

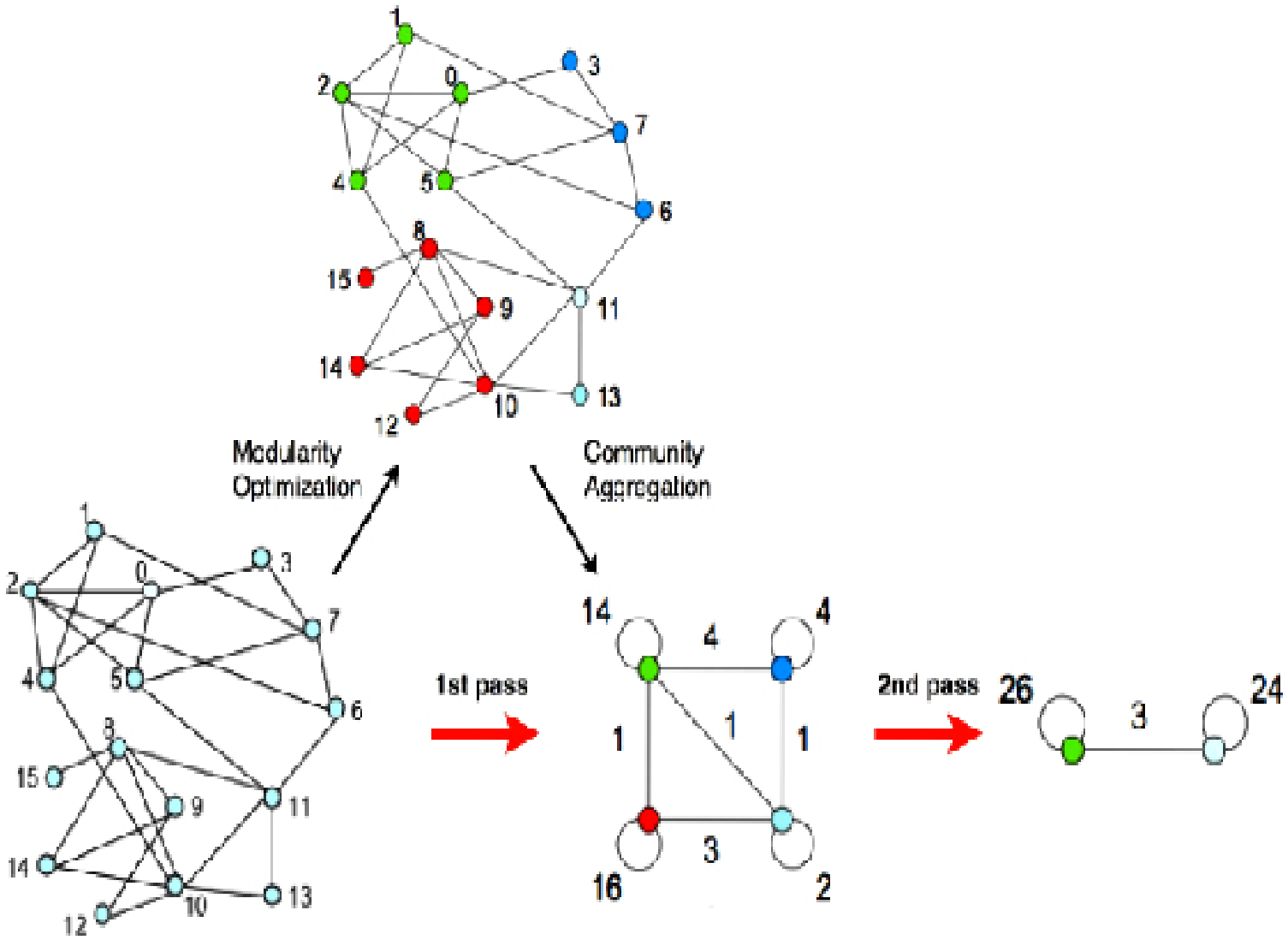
$\sum_{tot}$ est la somme des liens adjacents aux nœuds de la communauté cible,
$k_{i,in}$ est la somme des poids des liens reliant i à des nœuds de la communauté cible.

(suite Répéter jusqu'à stabilisation)

i est ajouté á la communauté qui maximise le gain si ce gain est positif.
Remplacer le graphe actuel par le graphe de connexion de communautés calculé comme suit :

Les nœuds sont les communautés
Le poids d'un lien (u, v) est la somme des poids des liens reliant tous les nœuds de u aux nœuds de v

Approche Agglomérative III



Optimisation de la modularité: Limites

Hypothèses

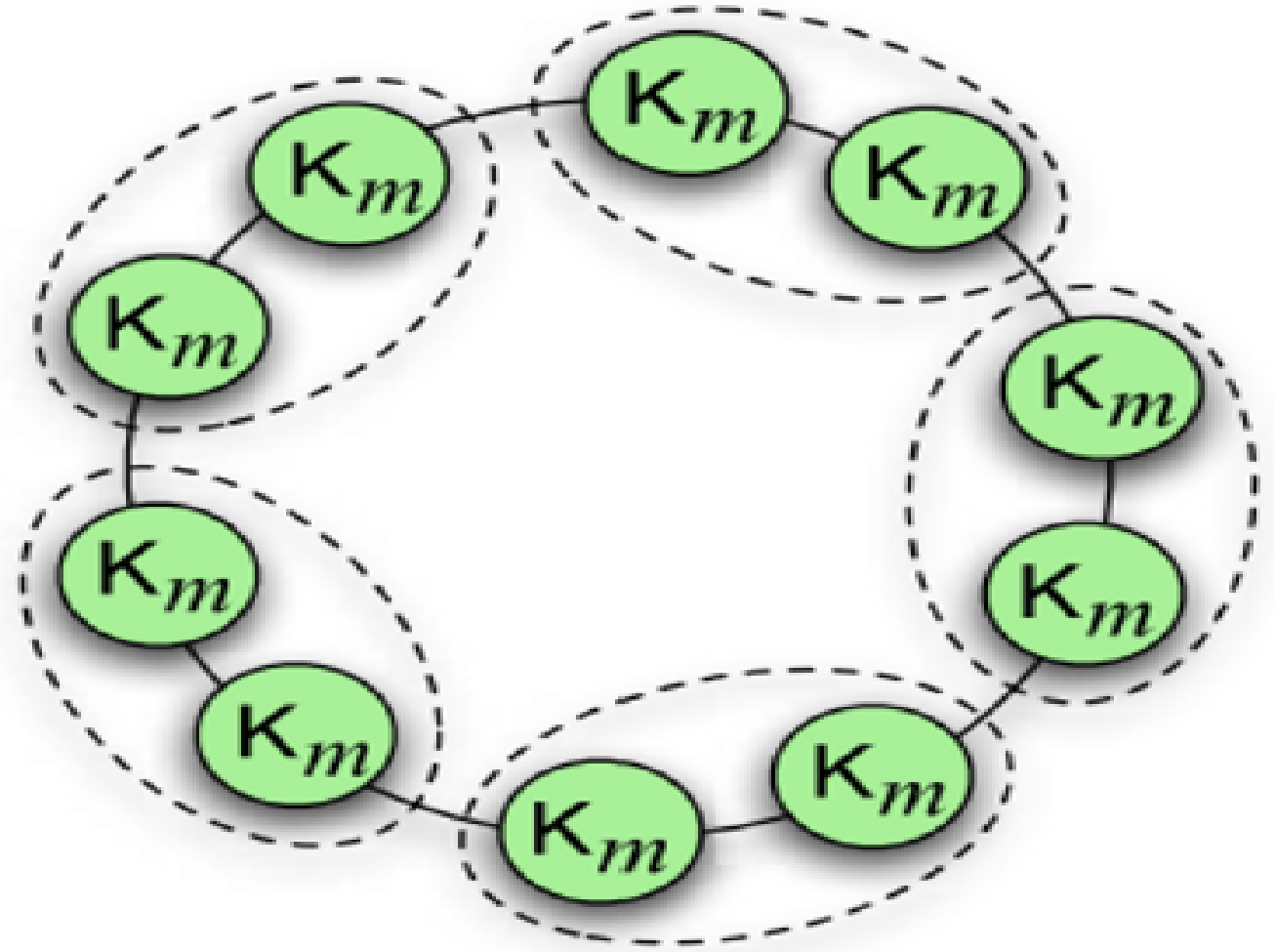
La meilleure partition est celle qui maximise la modularité

Si un graphe a une structure communautaire alors il est possible de trouver une partition precise qui avec une modularité maximales.

Les partitions ayant des modularités élevées sont similaires entre elles.

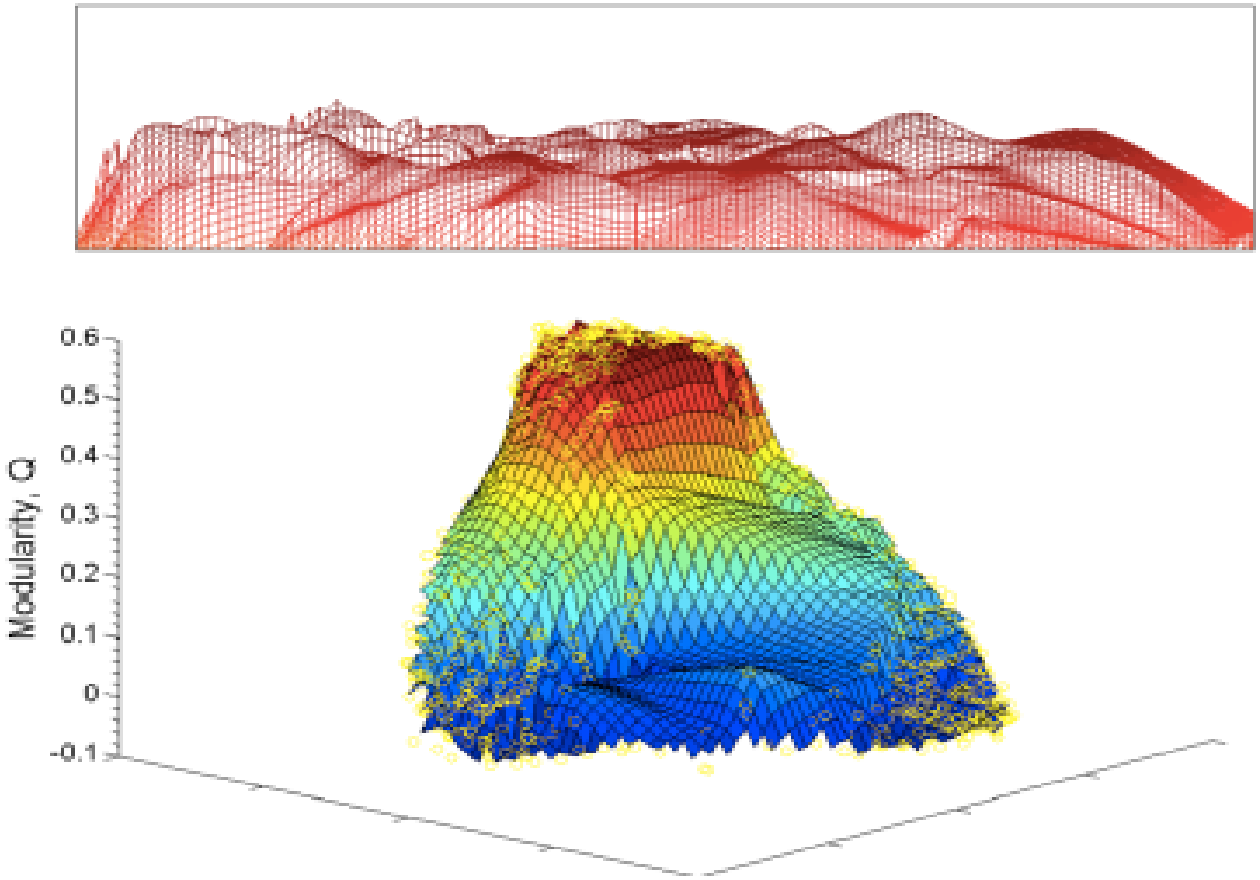
Les trois hypothèses sont fausses [GdMC10, LF11].

Modularité: limite de la résolution



$m = 3$; $Max_Q = 0.675$ tandis que la modularité de la partition naturelle

Distribution de la modularité



Approches de propagation

Algorithm 1 Label propagation

Require: $G =< V, E >$ a connected graph,

1: Initialize each node with unique label l_v
2: **while** Labels are not stable **do**
3: **for** $v \in V$ **do**

$$l_v = \arg\max_{l} |\Gamma^l(v)| \quad /* \text{ random tie-breaking } */$$

4: **end for**
5: **end while**
6: **return** communities from labels

$\Gamma^l(v)$: set of neighbors having label l

Approche de propagation d'étiquettes

`igraph.community_label_propagation`

1. Chaque nœud est attribué une étiquette unique.
2. Chaque nœud propage son étiquette à ses voisins.
3. A réception, chaque nœud adopte l'étiquette majoritaire.
4. En cas de conflit, un nœud choisit une étiquette aléatoirement.
5. Deux approches : Propagation synchrone et propagation asynchrone.

Problème :

Stabilité des partitions retrouvées.

Solution : calcul de *cœurs de communautés*

Label propagation

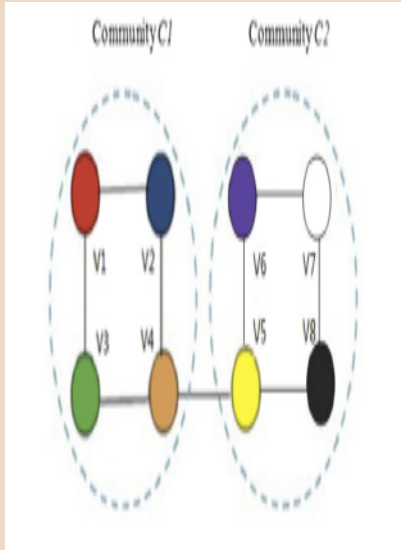
Avantages

- Complexité : $\mathcal{O}(m)$
- Simplicité de Paralléllisation

Inconvénients

- Problème de convergence, d'oscillation.
- Problème de stabilité

  Différentes exécutions donnent différentes solutions pour un même réseau



Label propagation: Convergence

- Asynchronous label update, [RAK07]
- Semi-synchronous label update (graph coloring + propagation by color) [CG12].

- Accentuer le problème de instabilité !
- Plus difficile à paralléliser.

Robust Label propagation

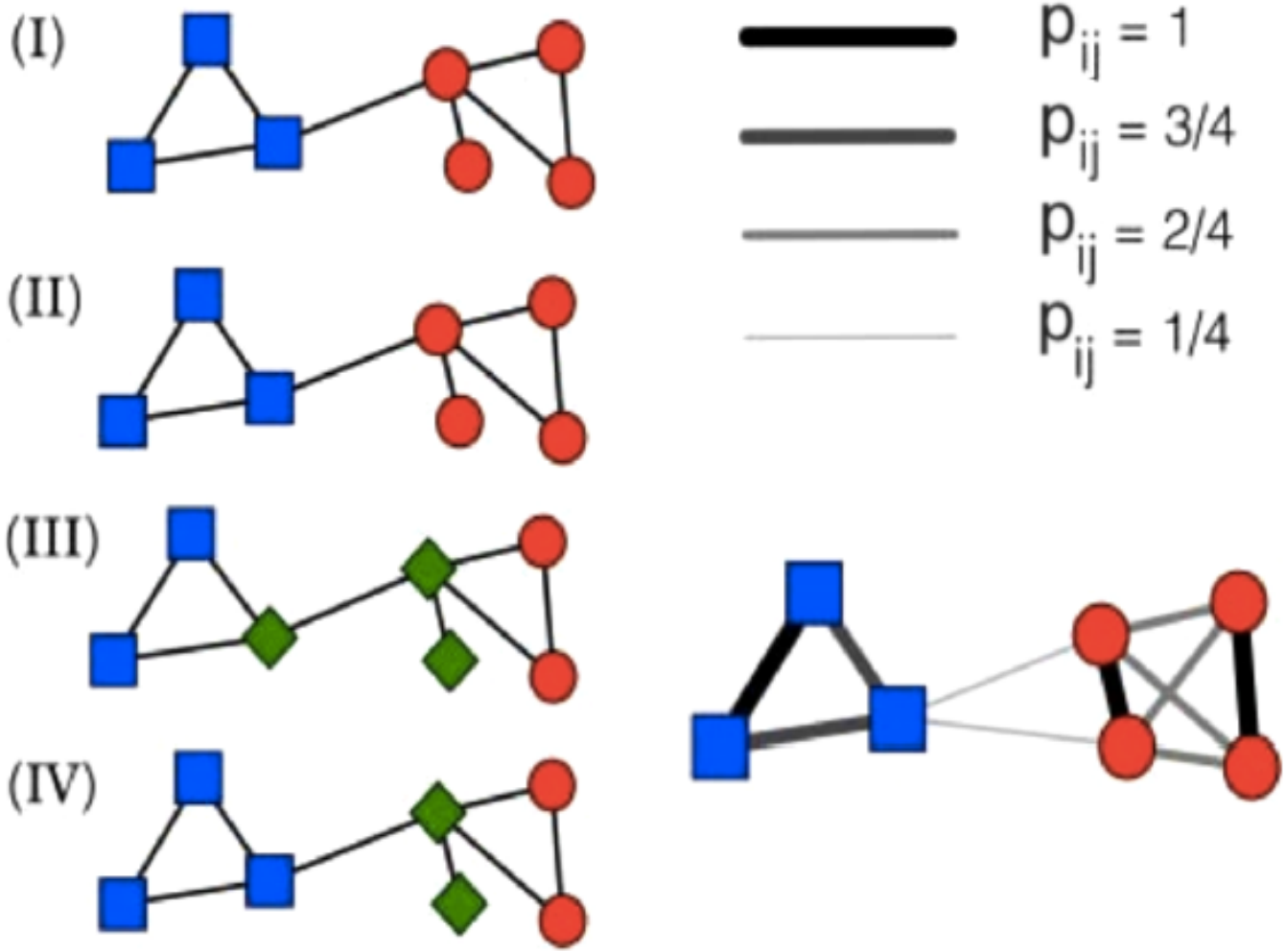
- *Label hop attenuation* [LHLC09]
- *Balanced label propagation* [SB11].

- Multiplex approach !
- **Ensemble clustering** $\rightarrow$ cœurs de communautés [OGS10, SG12, LF12].

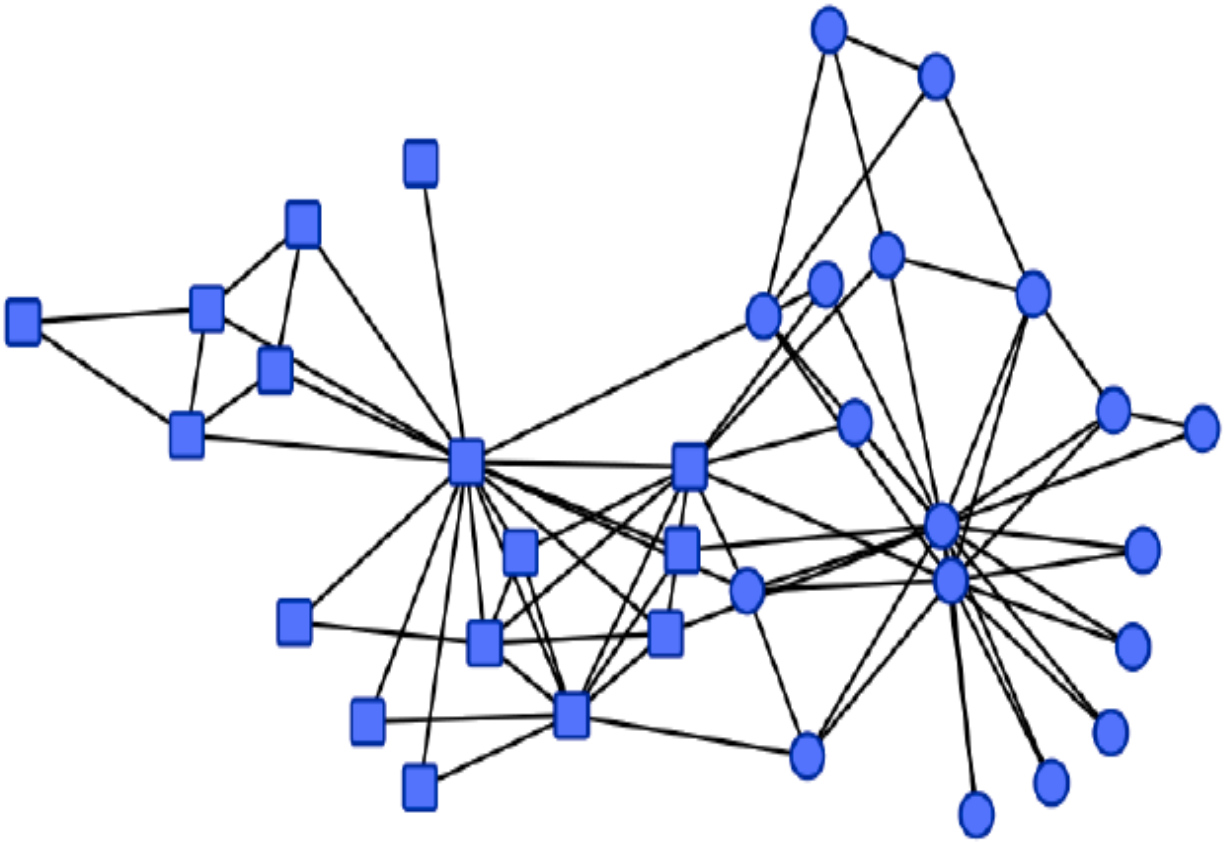
Cœurs de communautés I

- Etant donné N partitions d'un graphe G de n nœuds
- Calculer la matrice $n \times n$ de consensus $C = c_{ij}$ où c_{ij} est la fréquence de co-occurrence des nœuds i; j dans une même communauté.
- C est la matrice d'adjacence du graphe de consensus.

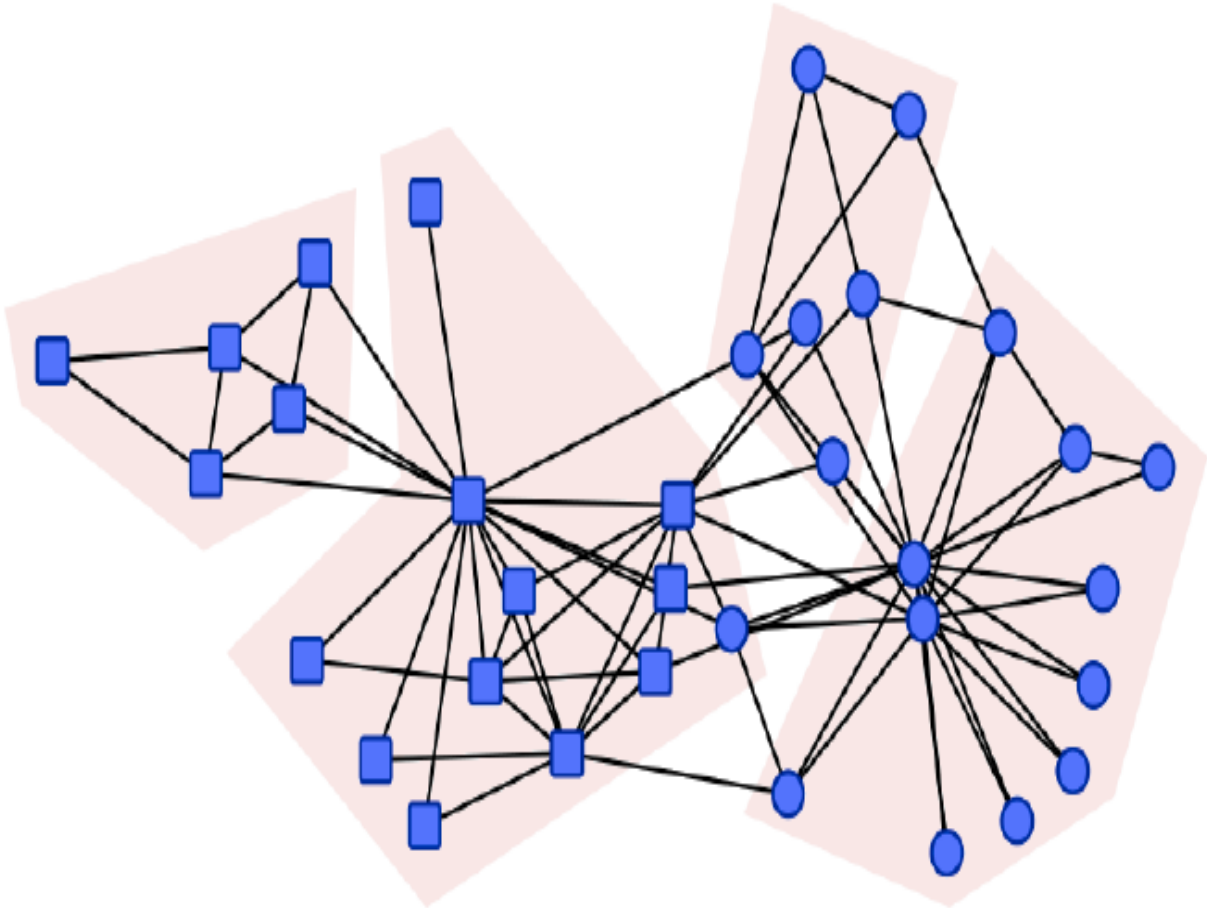
Cœurs de communautés II



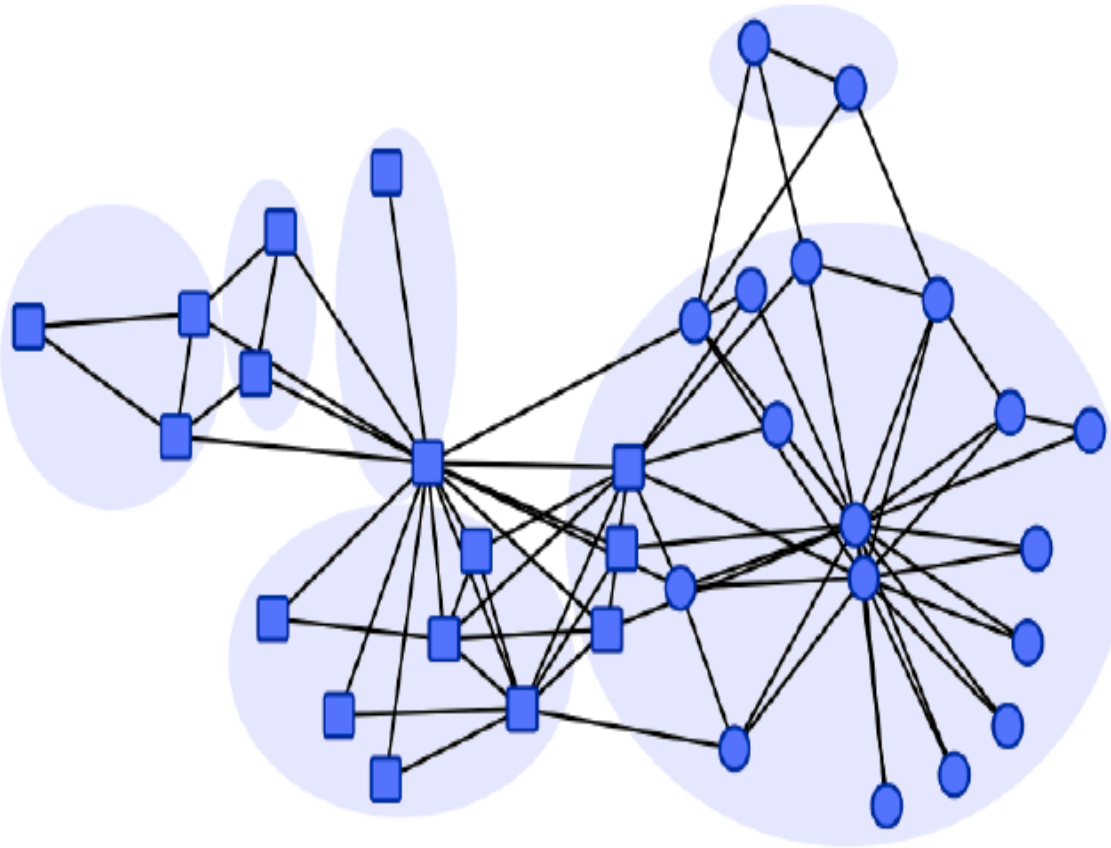
Exemple : Club de Karaté de Zachary I



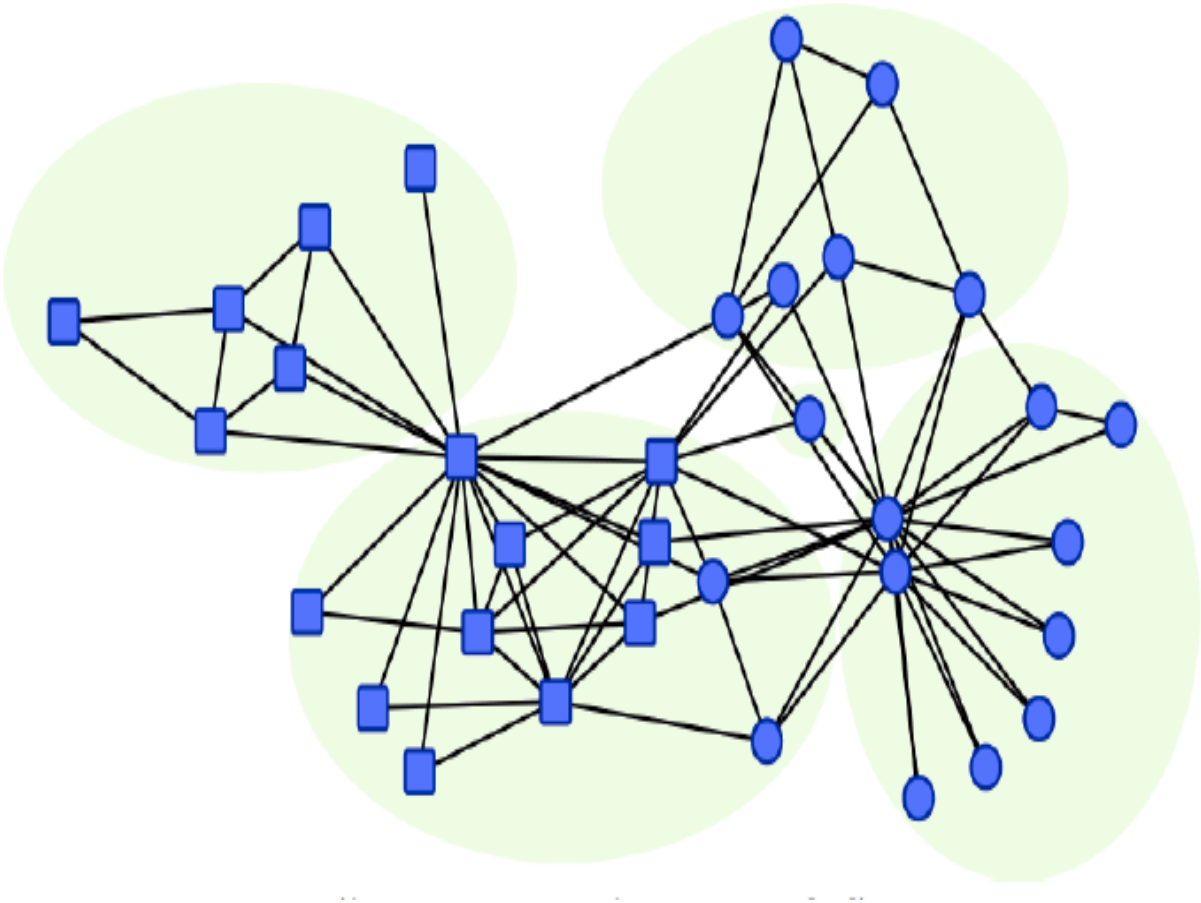
Exemple : Club de Karaté de Zachary II



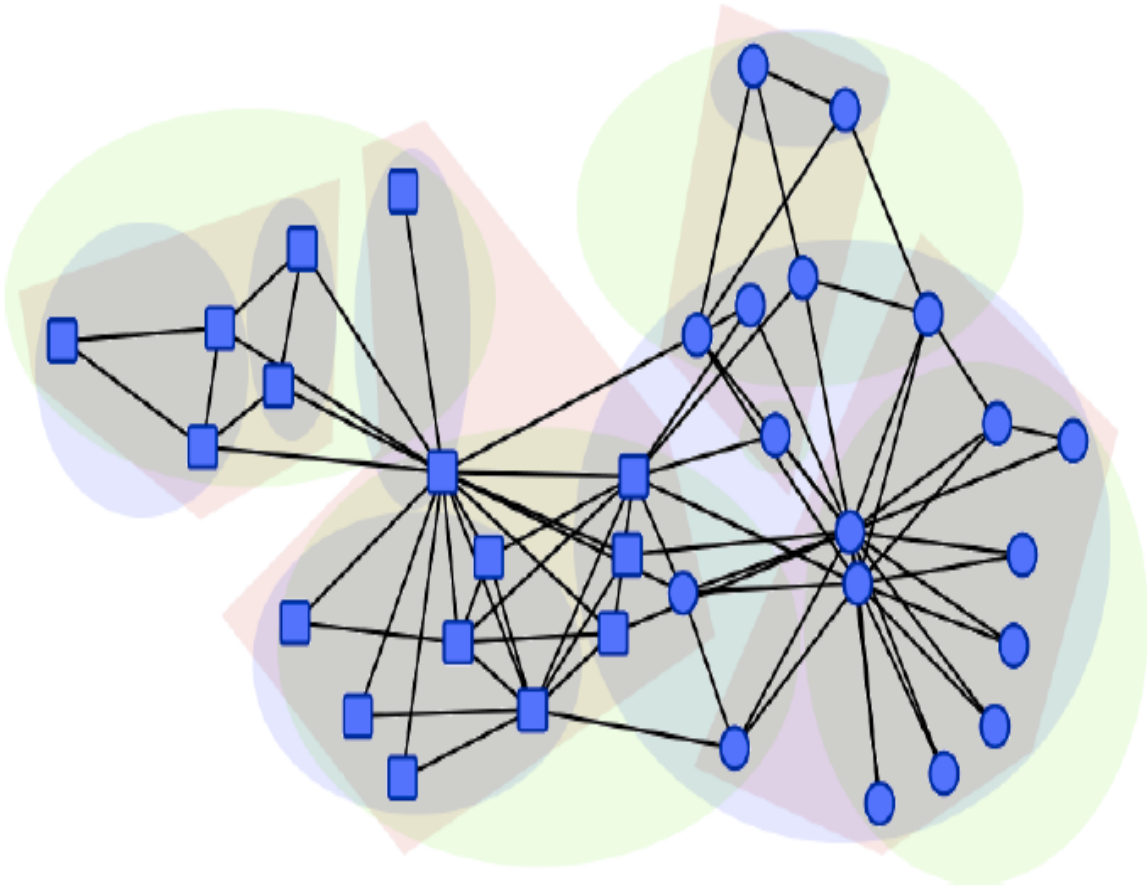
Exemple : Club de Karaté de Zachary III



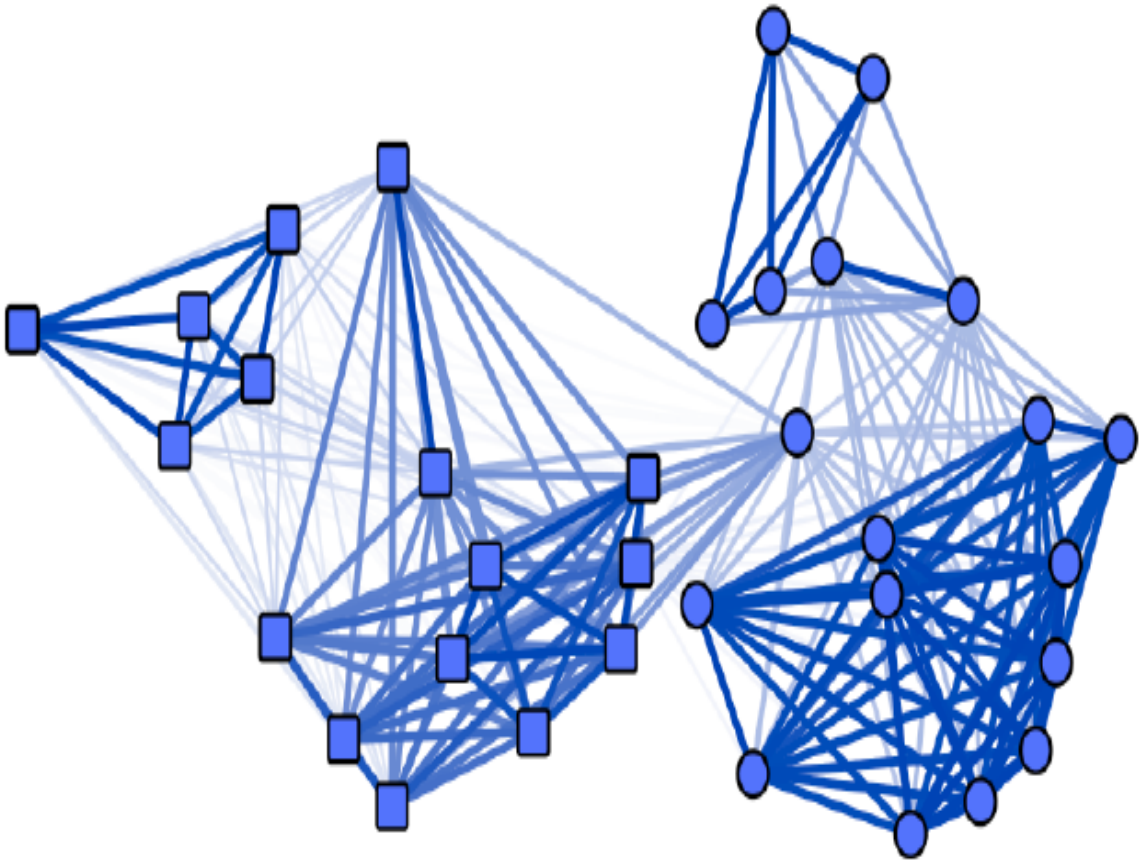
Exemple : Club de Karaté de Zachary IV



Exemple : Club de Karaté de Zachary V



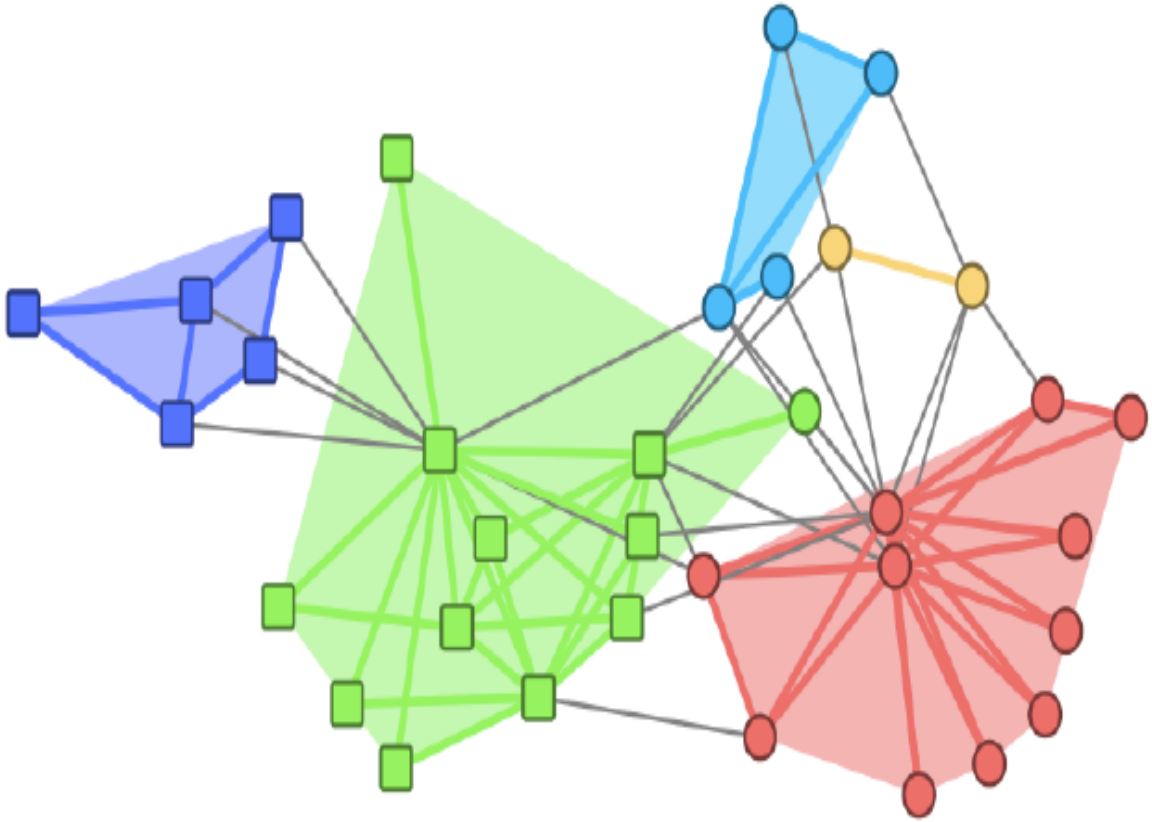
Exemple : Club de Karaté de Zachary VI



Cœurs de communautés

- Etant donné un seuil $\alpha \in [0, 1]$
- Retirer du graphe de consensus les liens dont le poids $\leq \alpha$
- Les composantes connexes du graphe résultat sont **les cœurs de communautés**

Exemple : Club de Karaté de Zachary



Label propagation preprocessing (EPP)

1. Appliquer l'algorithme LP n fois
2. Calculer les cœurs de communautés pour $\alpha = 1$
3. Réduire le graphe en substituant chaque communauté par un simple nœud.
4. Appliquer un algorithme de détection de communautés sur le graphe réduit.
5. Expansion des résultats.

Approches centrées graine

Algorithm 2 General seed-centric community detection algorithm

Require: $G =< V, E >$ a connected graph,

1: $\mathcal{C} \leftarrow \emptyset$
2: $S \leftarrow$ **compute_seeds(G)**
3: **for** $s \in S$ **do**
4: $\quad C_s \leftarrow$ **compute_local_com(s,G)**
5: $\quad \mathcal{C} \leftarrow \mathcal{C} + C_s$
6: **end for**
7: **return** **compute_community($\mathcal{C}$)**

Seed-centric approaches

Table: Characteristics of major seed-centric algorithms

Algorithm	Seed Nature	Seed Number	Seed selection	Local Com.	Com. computation
Leaders-Followers [SZ10]	Single	Computed	informed	Agglomerative	-
Top-Leaders [KCZ10]	Single	Input	Random	Expansion	-
PapadopoulosKVS12	Subgraph	Computed	Informed	Expansion	-
WhangGD13	Single	Computed	informed	Expansion	-
BollobasR09	Subgraph	Computed	informed	expansion	-
Licod [Kan11]	Set	Computed	Informed	Agglomerative	-
Yasca [Kan14]	Single	Computed	Informed	Expansion	Ensemble clustering

L'algorithme LICOD

1. Déterminer l'ensemble des Leaders $\mathcal{L}$
2. Répéter jusqu'à stabilisation ou max fois :

   Chaque $x \in V$ trie les communautés $c \in \mathcal{C}$ dans un ordre décroissant selon le degré d'appartenance P_x^0
   Pour chaque $x \in V$, calculer
   $P_x^t = FusionVotes(P_y^{t-1} y \in \{X\} \cup \Gamma(x))$
   Recalculer $\mathcal{L}$
3. Retourner pour chaque nœud la communauté placée en tête de vecteur de préférence.

LICOD: Identification de Leaders

Pour chaque $x \in V$, calculer les propriétés suivantes $prop(x)$:

- Propriété locale: centralité de degré
- Propriétés globales: centralité de proximité, centralité de d'intermédiarité (?)
- Autre: PageRank

Soit $F_x = \{y \in \Gamma(x) : prop(x) > prop(y)\}$

x est un Leader si $\frac{|F_x|}{|\Gamma(x)|} > \sigma \in [0, 1]$

LICOD: Degré d'appartenance

Le degré d'appartenance d'un nœud v à une communauté c est donnée par l'inverse du plus court chemin SP liant v à un des leaders de c:
$appartenance(v, c) = \frac{1}{(\min_{x \in COM(c)} SP(v,x))+1}$

LICOD: Fusion des votes

Application des méthode de votes:

- Méthodes de condercet non-consistantes: Majority, Borda
- Méthodes de condercet consistantes: Kemeny, local Kemeny

Exemple:

$A > B > C > D$
$A > B > C > D$
$B > C > A > D$
$D > A > B > C$
$B > D > A > C$

Borda $\rightarrow B > A > D > C > E$

Kemeny $\rightarrow A > B > C > D > E$

The YASCA algorithm

Algorithm 3 The Yasca community detection algorithm

Require: $G =< V, E >$ a connected graph,

1: $\mathcal{C} \leftarrow \emptyset$
2: $S \leftarrow$ **compute_seeds(G)** /* diverse seed nodes */
3: **for** $s \in S$ **do**
4: $C_s \leftarrow$ **compute_local_com(s,G)**
5: $\mathcal{C} \leftarrow \mathcal{C} + (C_s, \overline{C_s})$
6: **end for**
7: **return** **Ensemble_Clustering($\mathcal{C}$)**

Selection de graine

Diversité guidée par les attributs topologiques

Diversité basée sur la centralisé (Degrés , betweenness, . . . , etc)

Nœuds d'articulation + nœuds centrals dans le **bi-connected core**

Bi-connected core

Composant bi-connecté : A sous-graphe maximal qui reste connexe après la suppression de n'importe quel nœud.

A bi-connected core: Un graphe connexe maximal après la suppression de tous les compensates bi-connecté de taille 1.

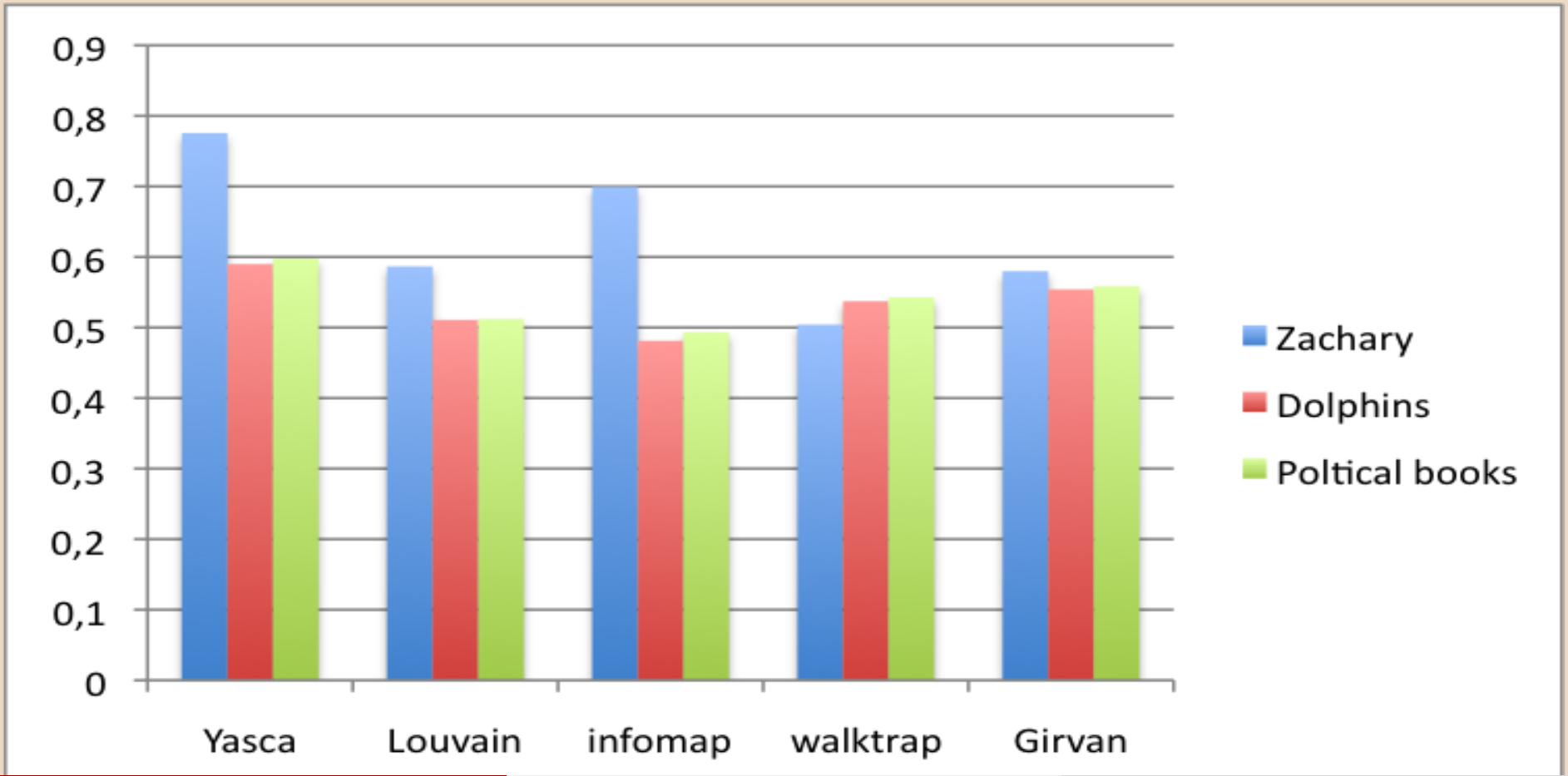
Pont: Un composnat bi-connecté de taille 1 connexe à un bi-connected core

Nœud d'articulation : L'intersection d'un pont et d'un bi-connected core.

Yasca: Résultats

Comparative Results on benchmark networks : NMI

Nœus d'articulation + top15% nodes centrales dans le bi-connected-core



Evaluation des algorithmes de détection de communautés

3 grandes approches :

- Qualités structurelles des communautés retrouvées.
- Comparaison avec une vérité de terrain.
- Evaluation orientés tâches

Fonctions d'évaluation structurelle

La qualité de $\mathcal{C} = \{S_1, \ldots, S_i\}$ est donnée par :

$$Q(\mathcal{C}) = \frac{\sum_i f(S_i)}{|\mathcal{C}|} \tag{2}$$

$f()$ est une fonction de qualité d'une communauté.

4 familles de fonctions de *scoring* :

- Fonctions basées sur la connectivité interne.
- Fonctions basées sur la connectivité externe.
- Fonctions hybrides
- Fonctions informées par des modèles.

Fonctions d'évaluation de la connectivité interne

Densité interne : $f(S) = \frac{2 \times m_S}{n_S \times (n_S - 1)}$

Degrés moyen : $f(S) = \frac{2 \times m_S}{n_S}$

FOMD: Fraction over Median Degree : $f(s) = \frac{|\{u : u \in S, |(u,v), v \in S| > d_m\}|}{n_S}$,
d_m est le médian de degrés des nœuds dans V

TPR: Triangle Participation Ratio : $\frac{|\{u \in S\} : \exists v, w \in S : (u,v), (w,v), (u,w) \in E|}{n_S}$

Fonctions d'évaluation de la connectivité externe

Expansion : $f(S) = \frac{C_S}{n_S}$

Cut : $f(S) = \frac{C_s}{n_S \times (N - n_S)}$

Fonctions hybrides

Conductance : $f(S) = \frac{C_S}{2m_S + C_S}$

MAX-ODF : Out Degree Fraction : $f(S) = max_{u \in S} \frac{|\{(u,v) \in E, v \notin S\}|}{d(u)}$

AVG-ODF : $f(S) = \frac{1}{n_S} \times \sum_{u \in S} \frac{|\{(u,v) \in E, v \notin S\}|}{d(u)}$

Fonctions guidées par un modèle

La modularité : $f(s) = \frac{1}{4}(m_S - E(m_S))$

$E(m_S)$ le nombre attendu de liens dans un graphe aléatoire ayant la même distribution de degrés.

Comparaison avec une véritié de terrain

Principe : calculer une similarité (ou distance) entre une partition calculée et une partition de référence.

La partition de référence peur être :

- annotée par un expert
- ou générée par un modèle.

Deux types de mesures :

- Mesures basées sur le compte des accords.
- Mesures basées sur la théorie de l'information.

Mesures basées sur le comptes des accords

Soient U, V deux partitions d'un graphe G

Indice de Rand :

a pairs placés dans une même communauté selon U et V

b pairs placés en même communauté selon U et en différents communauté selon V

c pairs placés en même communauté selon V et en différents communauté selon V

d pairs placées en différentes communauté selon U et selon V.

$rand = \frac{a+d}{a+b+c+d}$

$ARI = \frac{C_n^2(a+d)-[(a+b)(a+c)+(c+d)(b+d)]}{(C_n^2)^2-[(a+b)(a+c)+(c+d)(b+d)]}$

E(ARI)=0

rappel : $C_n^k = \frac{n!}{k!(n-k)!}$

L'information mutuelle

Rappel : L'information mutuelle entre deux variables aléatoires X, Y est : $I(X,Y) = H(X) + H(Y) - H(X,Y)$

$H(X) = \sum_{i=1}^{n_x} p(x_i) \times log(\frac{1}{p(x_i)})$

Normalisation : $NMI(U,V) = \frac{I(U,V)}{\sqrt{H(U)H(V)}}$

Problème de vérité de terrain

Existence de quelques graphes de benchmark (Zachary, Football, . . . , etc.),

Graphe de très petits tailles.

Difficile de trouver de grands graphes avec vérité de terrain.

Benchmarks artificiels : Le modèle LFR

But : Génération d'un graphe composé d'un ensemble de k communautés plus au moins interconnectés entre elles.

Algorithme :

Soit N le nombre de nœuds du graphe à générer. La distribution de degrés est tirée selon une loi de puissance de paramètre γ tel que le degrés d'un nœud est dans un intervalle $[K_{min}, K_{max}]$.

μ est un paramètre de connexion entre communauté : $\mu \in [0, 1]$

La distribution des tailles de communautés suit une autre lois de puissance de paramètre β.

Par itération : assigner un nœud à une communauté aléatoirement choisi de sorte que la taille de la communauté soit supérieur au **degrés interne** du nœud.

Si la communauté élue est complet on exclue un de ses membres choisi d'une manière aléatoire. Le nœud exclu devint un nœud non assigné.

L'algorithme s'arrête lorsque tous les nœuds sont assignés à des

Bibliography I


Gennaro Cordasco and Luisa Gargano, *Label propagation algorithm: a semi-synchronous approach*, IJSNM **1** (2012), no. 1, 3–26.


B. H. Good, Y.-A. de Montjoye, and A. Clauset., *The performance of modularity maximization in practical contexts.*, Physical Review **E** (2010), no. 81, 046106.


Rushed Kanawati, *Licod: Leaders identification for community detection in complex networks*, SocialCom/PASSAT, 2011, pp. 577–582.


———, *Yasca: An ensemble-based approach for community detection in complex networks*, COCOON (Zhipeng Cai, Alex Zelikovsky, and Anu G. Bourgeois, eds.), Lecture Notes in Computer Science, vol. 8591, Springer, 2014, pp. 657–666.


Reihaneh Rabbany Khorasgani, Jiyang Chen, and Osmar R Zaiane, *Top leaders community detection approach in information networks*, 4th SNA-KDD Workshop on Social Network Mining and Analysis (Washington D.C.), 2010.

Bibliography II


Andrea Lancichinetti and Santo Fortunato, *Limits of modularity maximization in community detection*, CoRR **abs/1107.1** (2011).


———, *Consensus clustering in complex networks*, Sci. Rep. **2** (2012).


Ian X.Y. Leung, Pan Hui, Pietro Lio, and Jon Crowcroft, *Towards real-time community detection in large networks*, Phys. Phy. E. **79** (2009), no. 6, 066107.


Michael Ovelgönne and Andreas Geyer-Schulz, *Cluster cores and modularity maximization*, ICDM Workshops, 2010, pp. 1204–1213.


Usha N. Raghavan, Roka Albert, and Soundar Kumara, *Near linear time algorithm to detect community structures in large-scale networks*, Physical Review E **76** (2007), 1–12.

Bibliography III


Lovro Subelj and Marko Bajec, *Robust network community detection using balanced propagation*, European Physics Journal B **81** (2011), no. 3, 353–362.


Massoud Seifi and Jean-Loup Guillaume, *Community cores in evolving networks*, WWW (Companion Volume) (Alain Mille, Fabien L. Gandon, Jacques Misselis, Michael Rabinovich, and Steffen Staab, eds.), ACM, 2012, pp. 1173–1180.


D Shah and T Zaman, *Community detection in networks: The leader-follower algorithm*, Workshop on Networks Across Disciplines in Theory and Applications, NIPS, 2010.