COURS SYRRES

DÉTECTION DE COMMUNAUTÉS

Jean-Loup Guillaume

Détection de communautés

- Objectif :
  - Identifier automatiquement des groupes pertinents.
- Applications :
  - Comprendre la structure des réseaux.
  - Détecter des communautés spécifiques (Pages web, terroristes…)
  - Visualisation.
  - Amélioration de moteurs de recherche, systèmes P2P, …
- Challenges :
  - Nombre de communautés inconnu.
  - Communautés de taille variable.
  - Passage à l'échelle : milliards de sommets.

Qu'est-ce qu'une communauté ?

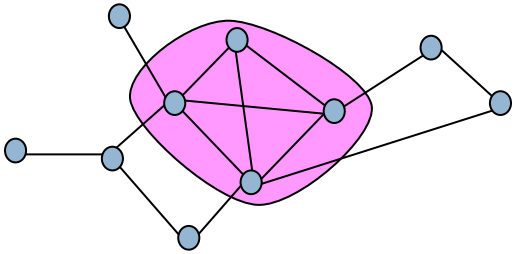
- Un ensemble de sommet d'un graphe qui partagent quelque chose :
  - Amis, collègues, …
  - Personnes avec des intérêts similaires.
  - Pages web avec un même contenu.
  - …

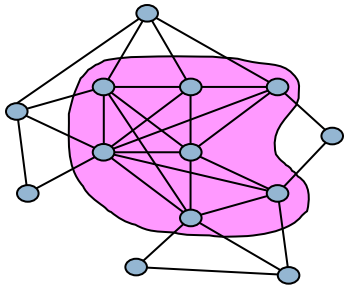
Lien avec la structure du réseau ?

Sous-graphes cohésifs

- Composantes connexes ou k-connexes :
  - Au moins (1) k chemins disjoints entre chaque paire de sommets.

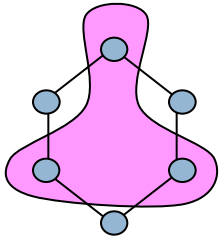
- Cliques :
  - Sous-graphe complètement connecté ou cliques recouvrantes.

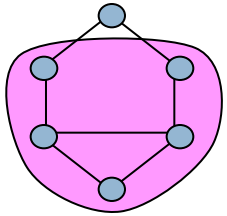




- n-cliques:
  - Sous-graphe G' avec distance inférieure à n dans G.

Peut être déconnecté

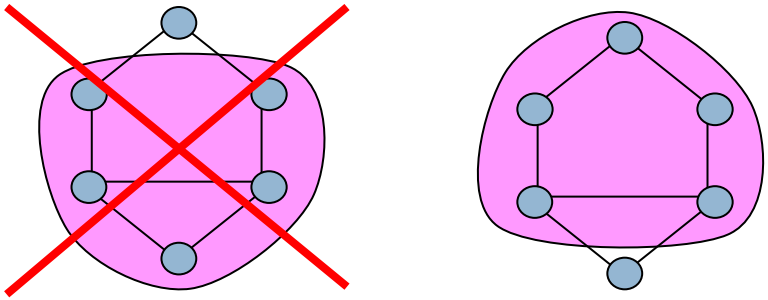




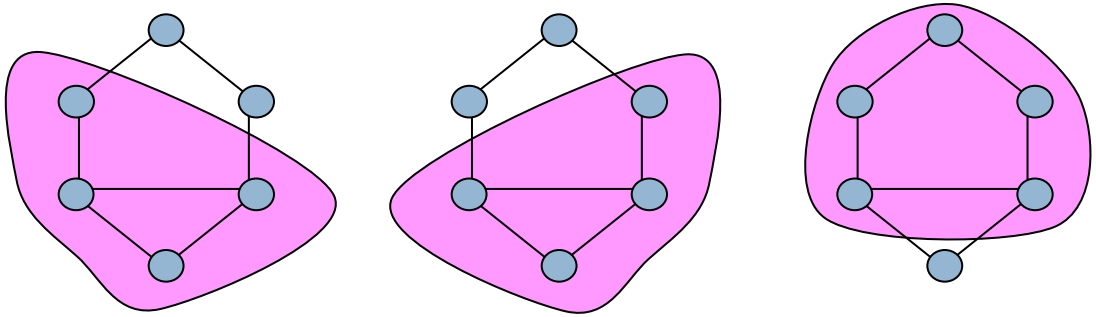
Diamètre peut être>n

Sous-graphes cohésifs (suite)

- n-clan
  - n-clique de diamètre n



- n-club:
  - Sous-graphe maximal de diamètre n.



Sous-graphes cohésifs

- k-degree set (k-core si maximal) :
  - Sous-graphe dans lequel tous les sommets ont degré au moins k.
- k-outdegree set:
  - Sous-graphe tel qu'aucun sommet n'a plus de k liens sortants.
- k-plex:
  - Sous-graphe maximal H t.q. chaque sommet est connecté à au moins |H|-k sommets.

- LS Set:
  - Sous-graphe minimal pour le nombre de liens sortants.
- Alpha set:
  - Sous-graphe t.q. chaque sommet a plus de liens internes que sortant.

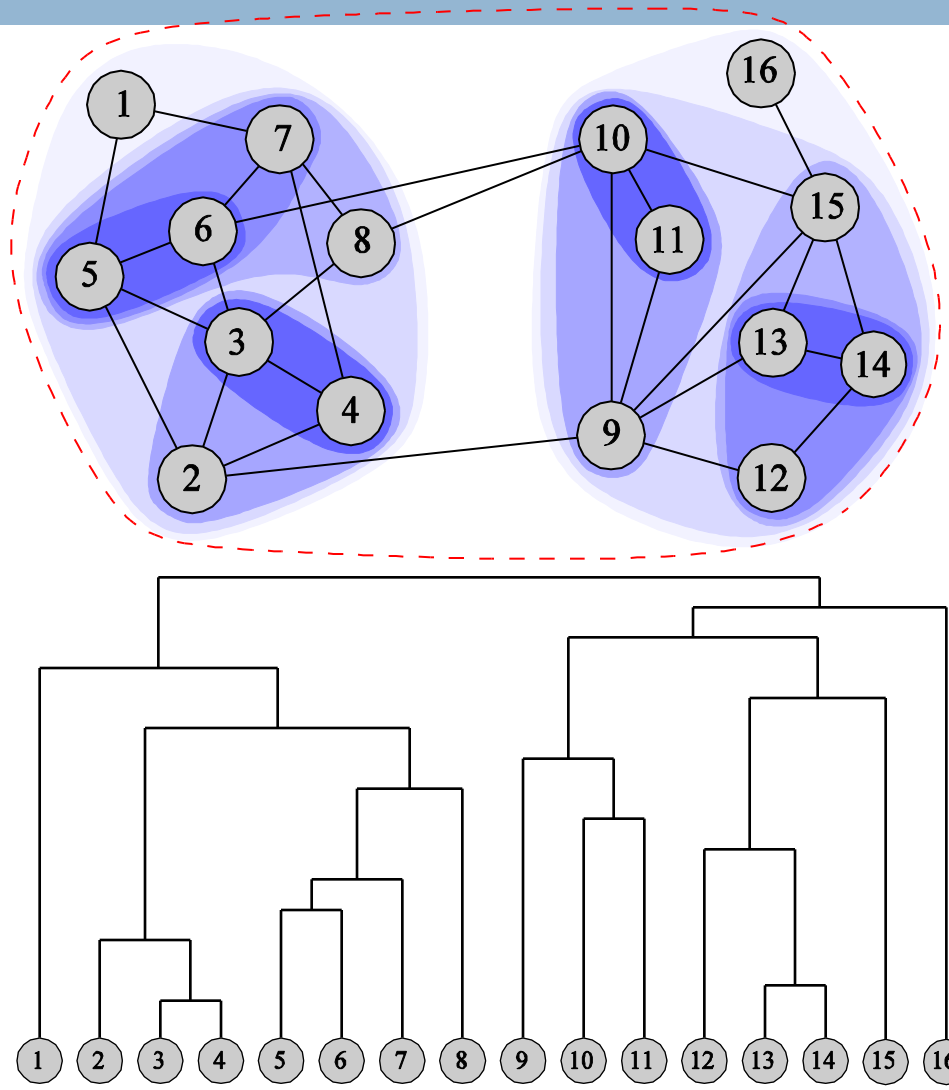
Calcul

- Complexité :
  - Souvent NP-complet : Cliques, k-plex…
  - Parfois polynomial : LS, Lambda sets (n^4 ou moins)…
  - Passage à l'échelle : web ~ 10^{10} sommets !

- Communautés recouvrantes :
  - Un sommet peut être dans plusieurs communautés.
  - Certaines définitions sont recouvrantes : Cliques…
  - Ou pas : K-cores…

Il faut autre chose !

Clustering hiérarchique



Slide de Pascal Pons

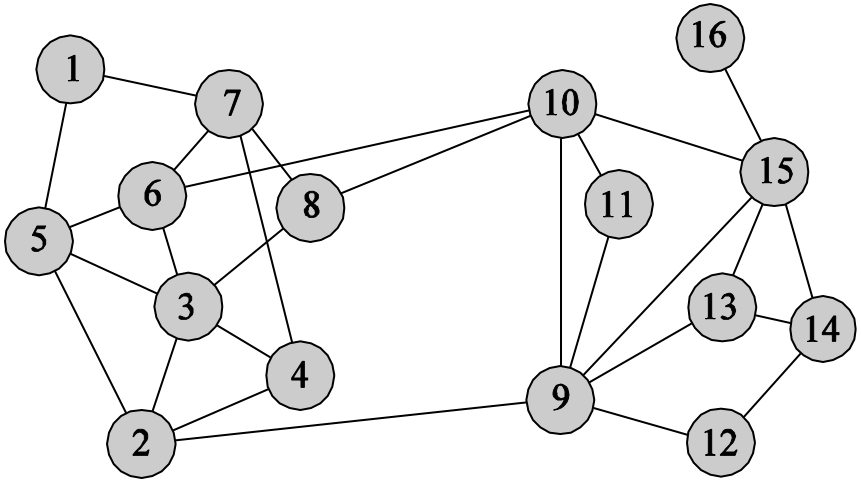
Clustering hiérarchique (suite)

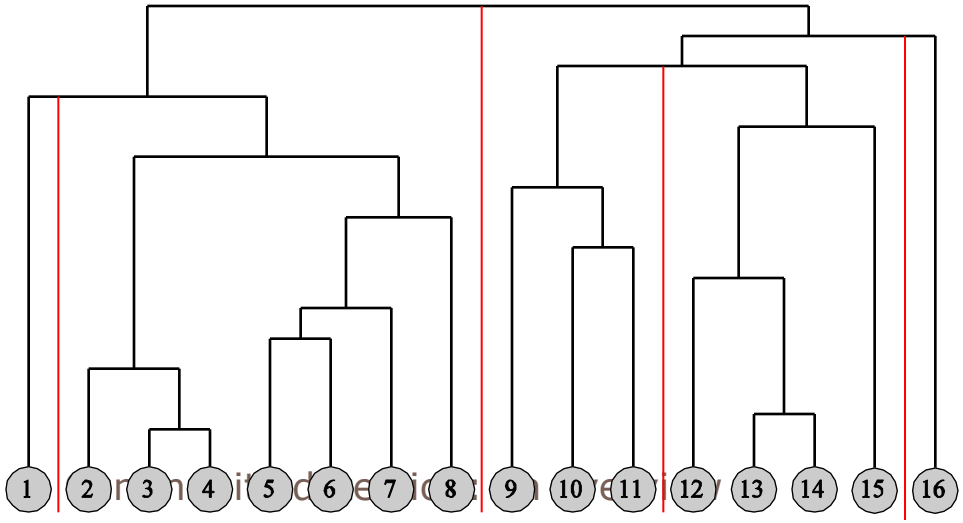
- Algorithme générique :
  1. Chaque sommet est dans une communauté.
  2. Calculer une distance entre chaque paire de communautés.
  3. Fusionner les deux plus proches.
  4. Revenir à 2.

- Distance entre deux sommets ?

- Distance entre communautés = distance entre sommets +
  - Min, max, moyenne, centre de gravité, …

Approches divisives





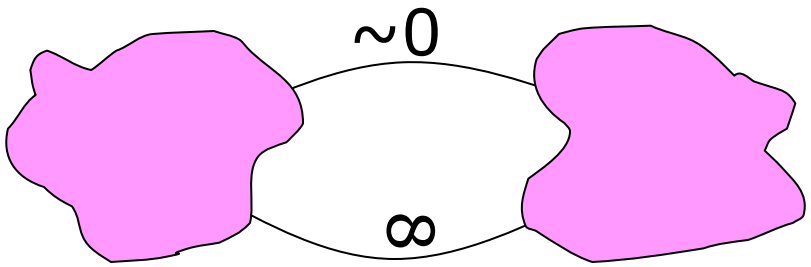
Adapté d'un slide de Pascal Pons

Approches divisives (suite)

- Algorithme générique :
  1. Calculer un score (?) d'inter-communautarisme pour chaque lien.
  2. Supprimer les lien le plus fort.
  3. Revenir à 1.

- Score ?

- Attention : il faut recalculer le score à chaque fois



Qualité d'un algorithme ?

- Validation ad-hoc :
  - On dispose de données pour valider la décomposition.
- Test sur des graphes simples:
  - 4 communautés de 32 sommets
  - Probabilité d'existence de liens interne et externe
  - Est-ce que l'algo arrive à indentifier les 4 groupes ?
- Utilisation de la modularité

LA MODULARITÉ DÉFINITIONS ET LIMITES

Jean-Loup Guillaume

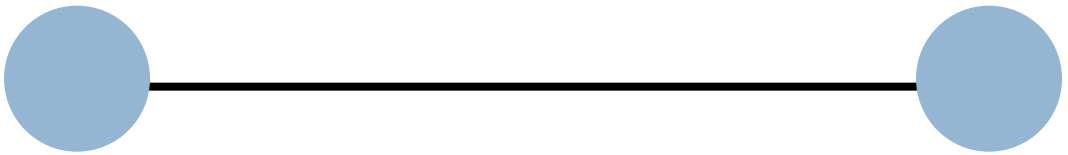
Structure communautaire

- Définition : Une communauté est un sous-graphe dont les sommets sont plus liés entre eux qu'avec le reste du réseau.

- Modularité : une mesure pour comparer le nombre de liens dans un groupe de sommets à ce que l'on attendrait pour un graphe aléatoire similaire.
  - Différence entre
    - Nombre de liens dans un module et
    - Nombre attendu de liens dans un graphe similaire.

Définition

- Probabilité qu'un demi lien soit dans s : $\frac{d_s}{2L}$
- Probabilité qu'un lien soit dans s : $\frac{d_s}{2L} \cdot \frac{d_s}{2L}$
- Nombre attendu de liens dans s : $\frac{d_s}{2L} \cdot \frac{d_s}{2L} \cdot L = \frac{d_s^2}{4L}$



$\frac{d_s}{2L}$ $\frac{d_s}{2L}$

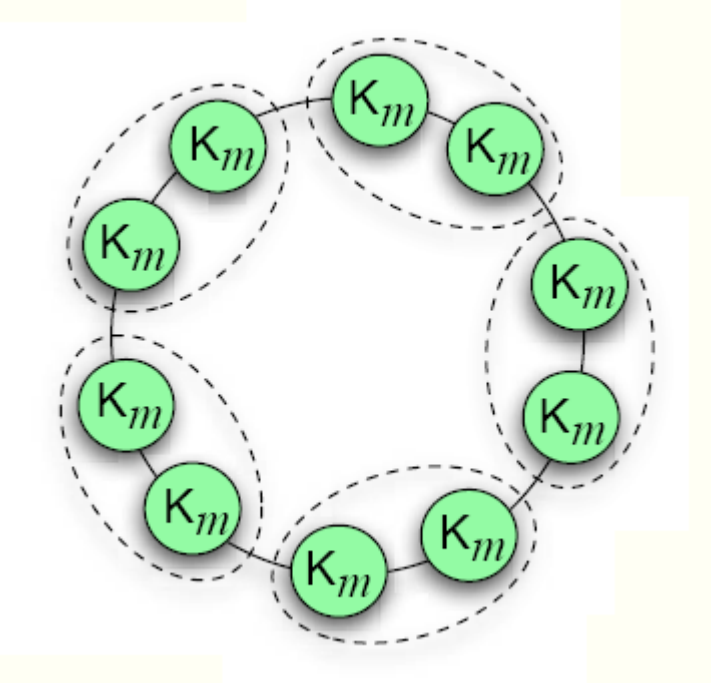
Définition

$$Q = \frac{1}{L}\sum_{s=1}^{m}\left(l_s - \frac{d_s^2}{4L}\right) = \sum_{s=1}^{m}\left[\frac{l_s}{L} - \left(\frac{d_s}{2L}\right)^2\right]$$

- l_s : nombre de liens dans le groupe s
- L : nombre de liens dans le graphe

Limite de résolution

- Anneau de cliques :
  - n cliques.
  - De taille n.



$$Q_{\text{sin}\,gle} = 1 - \frac{2}{m(m-1)+2} - \frac{1}{n}$$

$$Q_{pairs} = 1 - \frac{1}{m(m-1)+2} - \frac{2}{n}$$

Limite de résolution

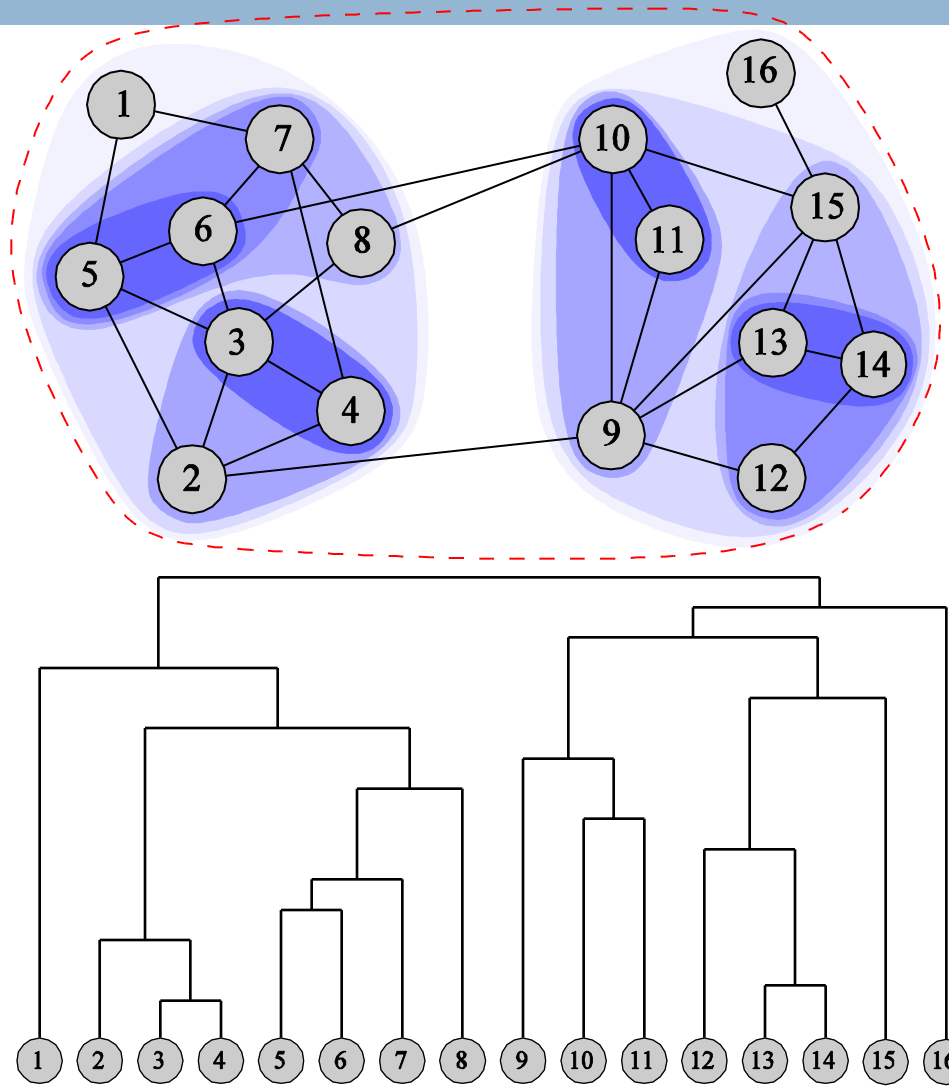
$$Q_{\sin gle} > Q_{pairs} \Leftrightarrow m(m-1)+2 > n$$

- 30 cliques, chacune de taille 5
  - 30>22
  - Qs = 0.876, Qp = 0.888
  - Contre-intuitif
- Peut apparaître à n'importe quelle échelle
  - Plus probable si taille(c) < racine(m)

DIFFÉRENTS ALGORITHMES DE DÉTECTION DE COMMUNAUTÉS

Jean-Loup Guillaume

Clustering hiérarchique



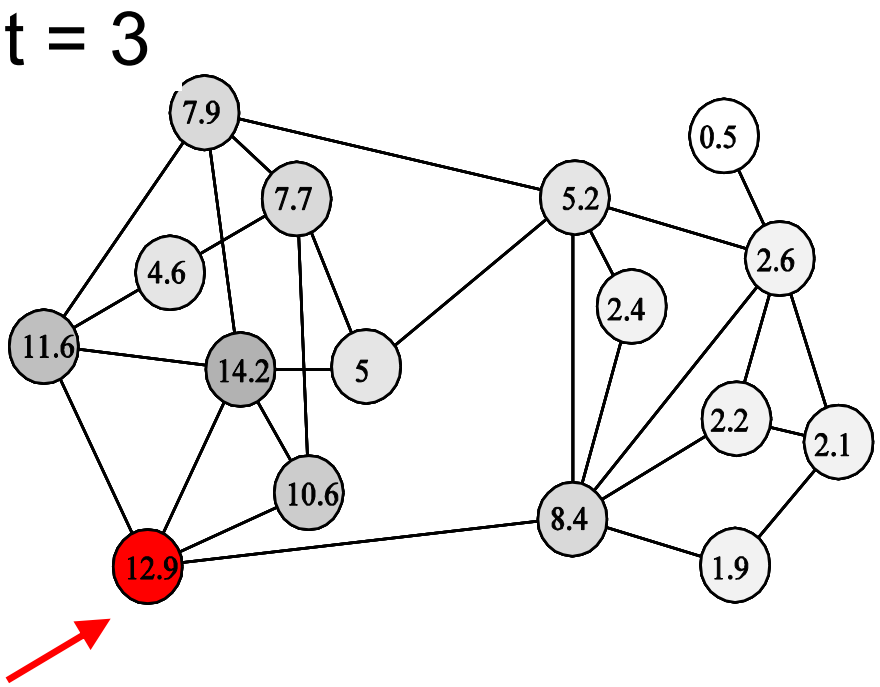
Slide de Pascal Pons

Méthode gloutonne

- À chaque étape fusionner les deux sommets/communautés pour maximiser la modularité.

Chemins aléatoires

- Chemins aléatoires de longueur fixée :
  - Chemins courts : pas assez d'information.
  - Chemins longs : aucune information (proba~degré).



$$r_{ij} = \sqrt{\sum_{k=0}^{n} \frac{\left(P_{ik}^{t} - P_{jk}^{t}\right)^2}{d(k)}}$$

Approches spectrales

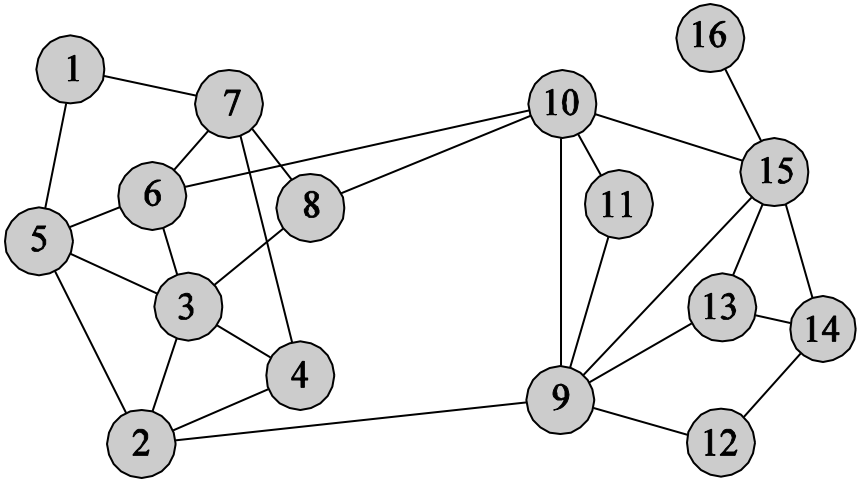
- Laplacien ou Normal matrix de G.

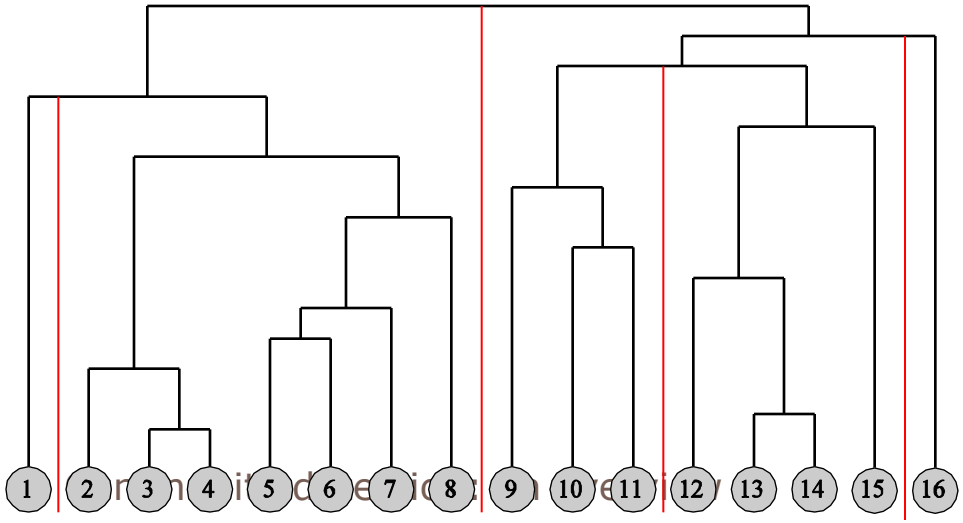
$$L = D - A \qquad N = D^{-1}A$$

- SI les communautés sont claires :
  - Vecteurs propres permettent de trouver les communautés.
- Sinon il faut une distance entre les vecteurs.

- Approches à base de matrices de similarité K :
  - Matrices plus complexes (positive semi définies).
  - Distances simples à partir de K.

Approches divisives





Adapté d'un slide de Pascal Pons

Centralité

- Un lien entre deux communauté sera plus certainement utilisé si l'on cherche un chemin entre deux sommets

- Centralité des plus courts chemins :
  - Nombre de plus courts chemins utilisant un lien donné.

- Centralité des chemins aléatoires :
  - Nombre de fois qu'un lien est utilisé dans un chemin aléatoire.

Efficacité

$$E = \left[\frac{1}{dist(i,j)} \right] = \frac{1}{N.(N-1)} \sum_{i,j} \frac{1}{dist(i,j)}$$

- Centralité

$$c_{\{i,j\}} = \frac{\Delta E_{\{i,j\}}}{E} = \frac{E(G) - E(G \setminus \{i,j\})}{E(G)}$$

Recuit simulé

- A chaque étape, mises à jour aléatoires :
  - Déplacements individuels.
  - Déplacements collectifs : fusion ou séparation
  - Accepté avec probabilité

$$p = \begin{cases} 1 & SI\ Q_a \geq Q_b \\ \exp\left(-\dfrac{Q_b - Q_a}{T}\right) & SINON \end{cases}$$

- Le système est refroidi :
  - T'=T.0.995

Algorithme génétique

- Chromosomes :
  - Partition en communautés (v[1..n]).
  - Initiallement : communautés de sommets connectés.
- Cross-over:
  - Sommets qui appartiennent à c dans C1 vont vers c dans C2.
- Mutation:
  - Un sommet est déplacé aléatoirement.
- Clean up:
  - Un sommet dont les voisins sont ailleurs est mis avec eux.

- Les mauvaises configuration (faible Q) sont enlevées.

Et beaucoup d'autres

- Équations de Kirchhoff
- Wu and Huberman, Eur Phys B 38, 2004
- Modèle de Potts
- Reichardt and Bornholdt, Phys Rev Lett 93, 2004
- …

Complexité

Author	Ref.	Label	Order
Eckmann & Moses	[13]	EM	$O(m\langle k^2\rangle)$
Zhou & Lipowsky	[14]	ZL	$O(n^3)$
Latapy & Pons	[15]	LP	$O(n^3)$
Newman	[24]	NF	$O(n\log^2 n)$
Newman & Girvan	[25]	NG	$O(m^2n)$
Girvan & Newman	[32]	GN	$O(n^2m)$
Guimerà et al.	[27, 43]	SA	parameter dependent
Duch & Arenas	[31]	DA	$O(n^2\log n)$
Fortunato et al.	[33]	FLM	$O(n^4)$
Radicchi et al.	[34]	RCCLP	$O(n^2)$
Donetti & Muñoz	[35, 36]	DM/DMN	$O(n^3)$
Bagrow & Bollt	[37]	BB	$O(n^3)$
Capocci et al.	[38]	CSCC	$O(n^2)$
Wu & Huberman	[39]	WH	$O(n+m)$
Palla et al.	[40]	PK	$O(\exp(n))$
Reichardt & Bornholdt	[41]	RB	parameter dependent

Table from cond-mat/0505245

Conclusion

- Deux approches principales :
  - Similarité entre sommets.
  - Liens inter-communautaires.
  - Mais pas seulement.
- Cas dirigé et pondéré :
  - Trivial la plupart du temps.

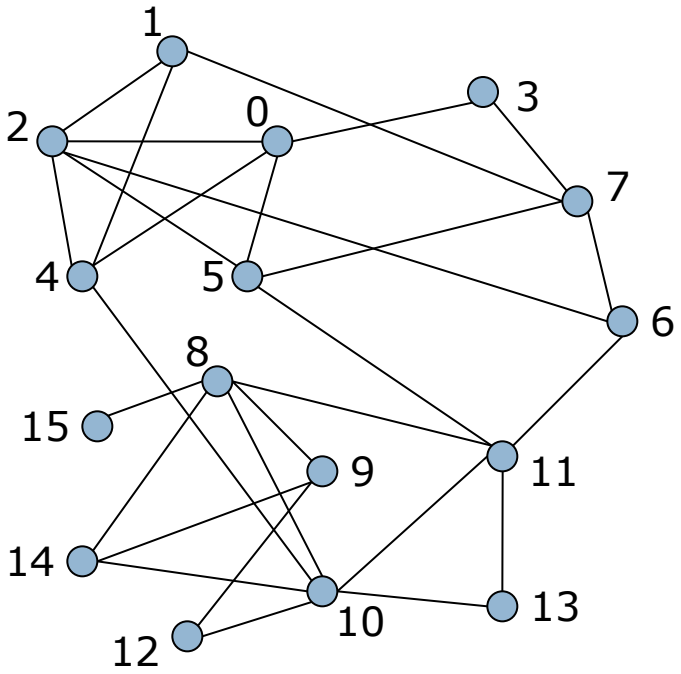
Qualité d'un algorithme ?

Ou qualité d'une partition

- Validation ad-hoc.
- Test sur des graphes simples:
  - 4 communautés de 32 sommets.
  - Probabilité d'existence de liens internes et externes.
- Utilisation de la modularité.
- Utilisation de données réelles.

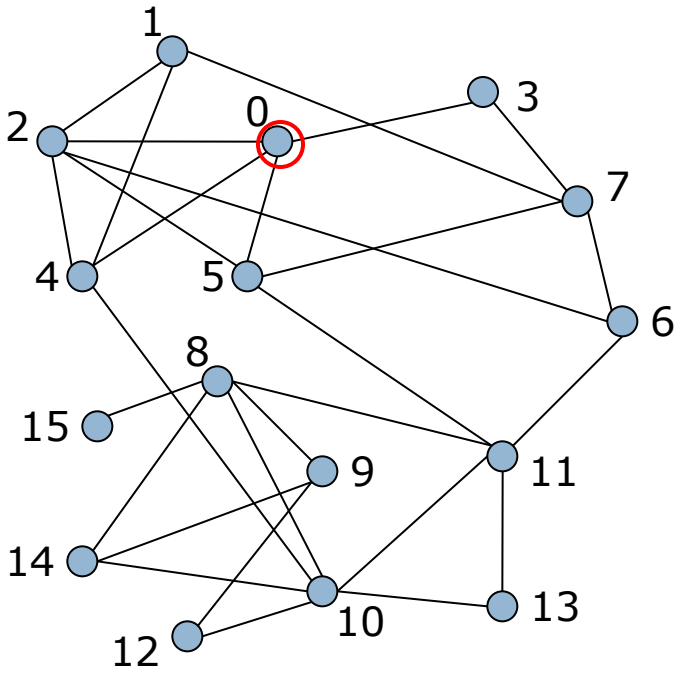
UN ALGORITHME SUPPLÉMENTAIRE

Un exemple

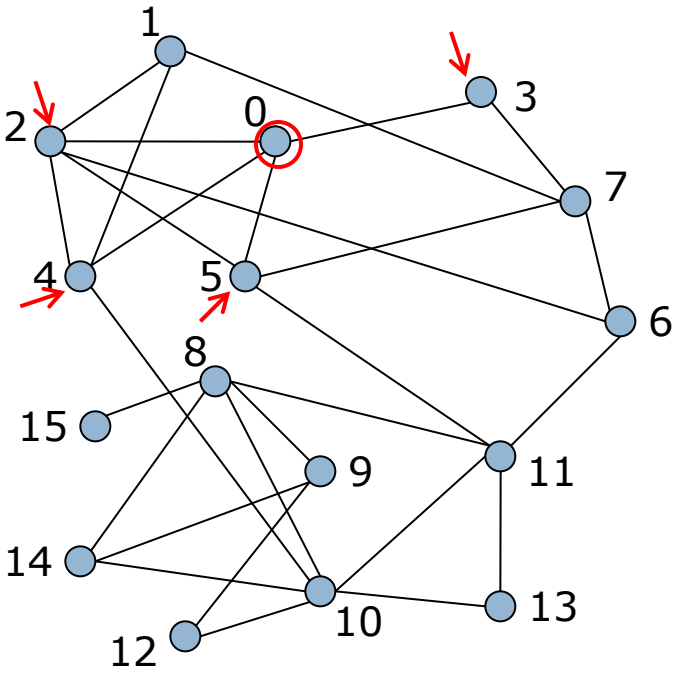


Passe 1 – Itération 1
chaque sommet est isolé

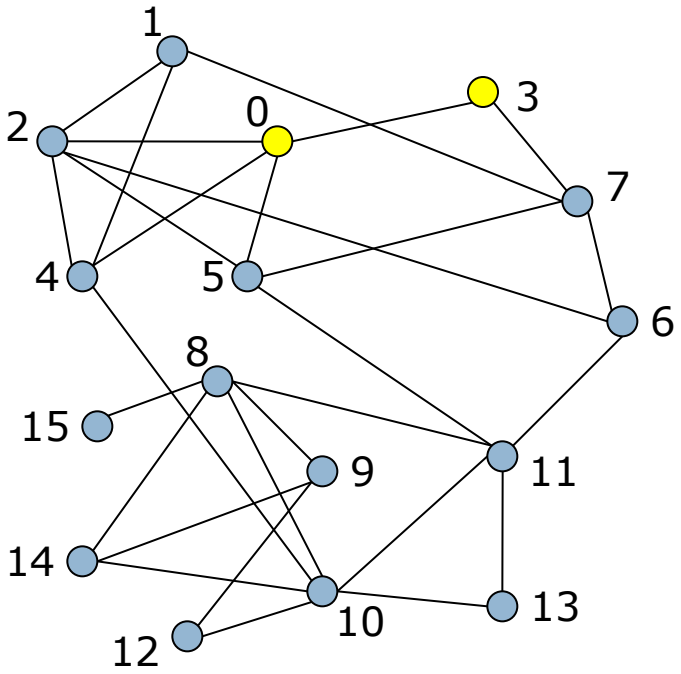
Un exemple



Un exemple

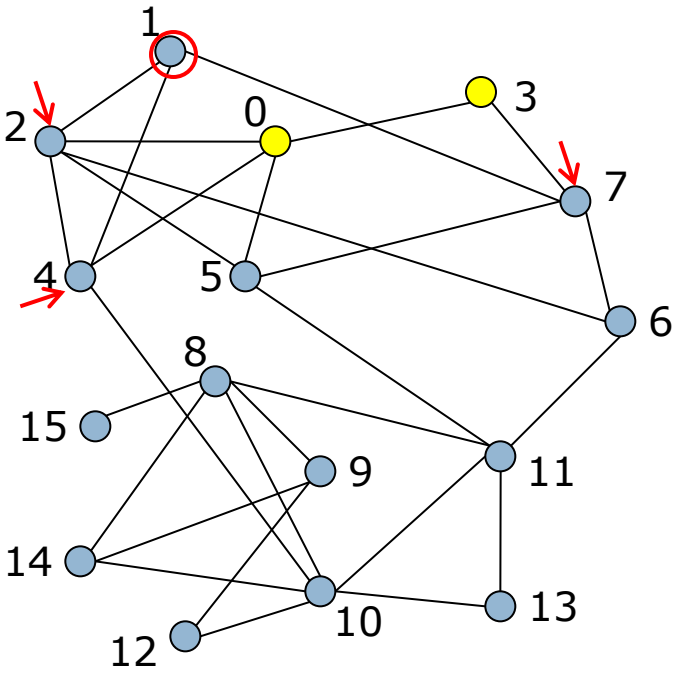


Un exemple



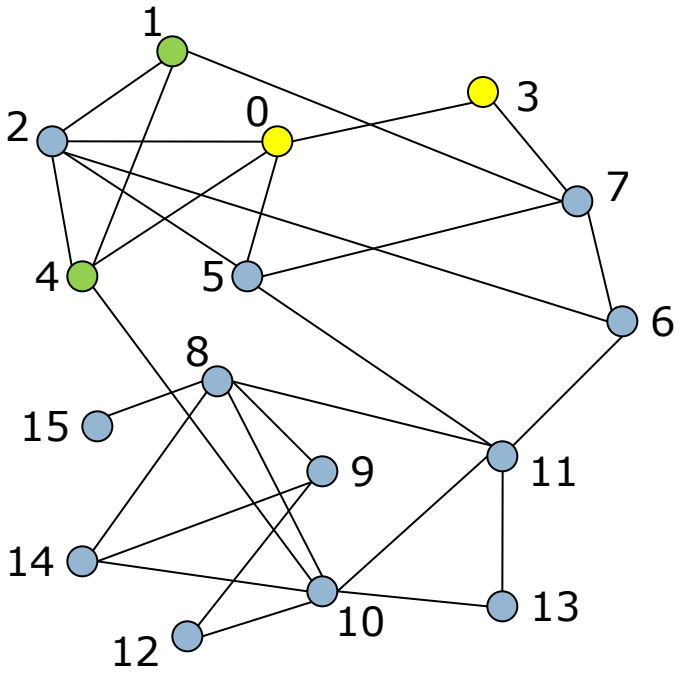
0 -> c[3]

Un exemple



0 -> c[3]

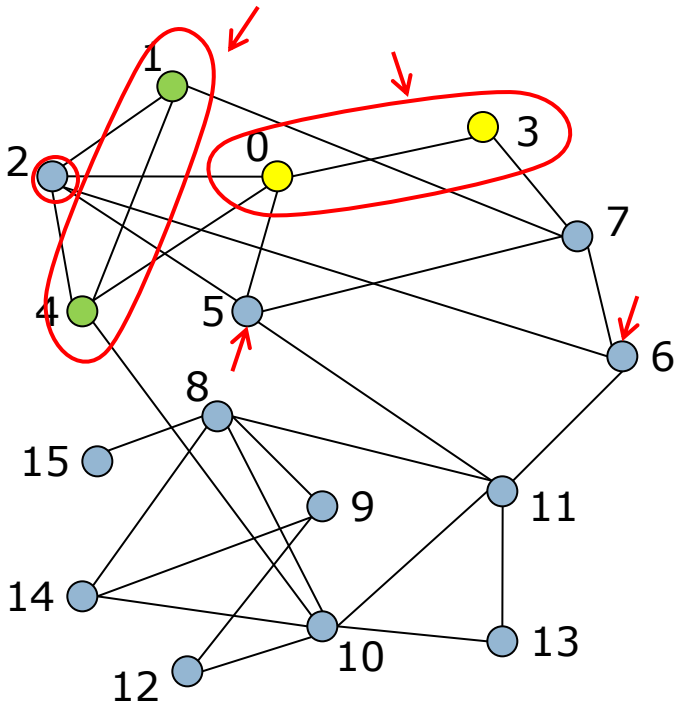
Un exemple



0 -> c[3]

1 -> c[4]

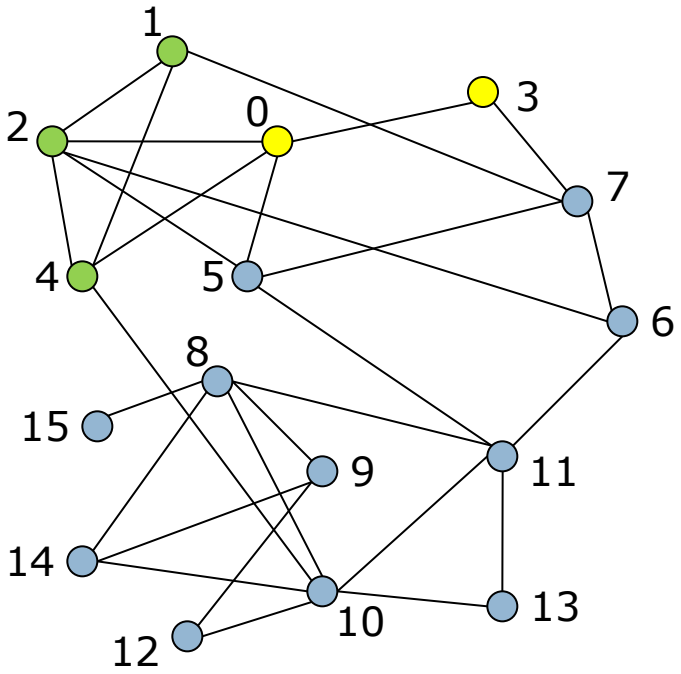
Un exemple



0 -> c[3]

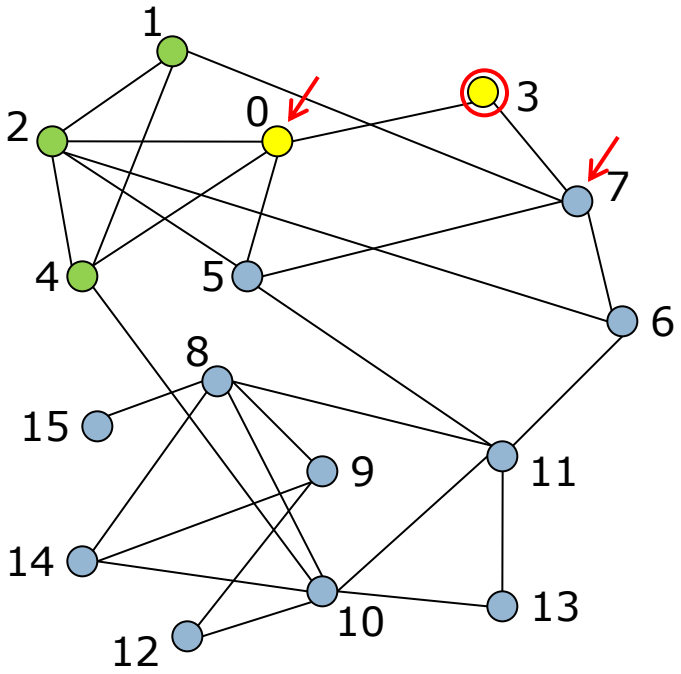
1 -> c[4]

Un exemple



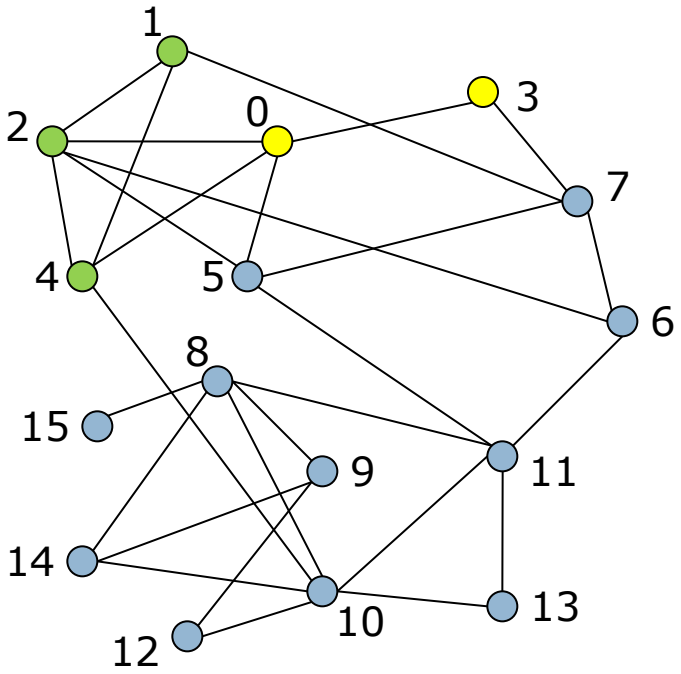
0 -> c[3]
1 -> c[4]
2 -> c[1,4]

Un exemple



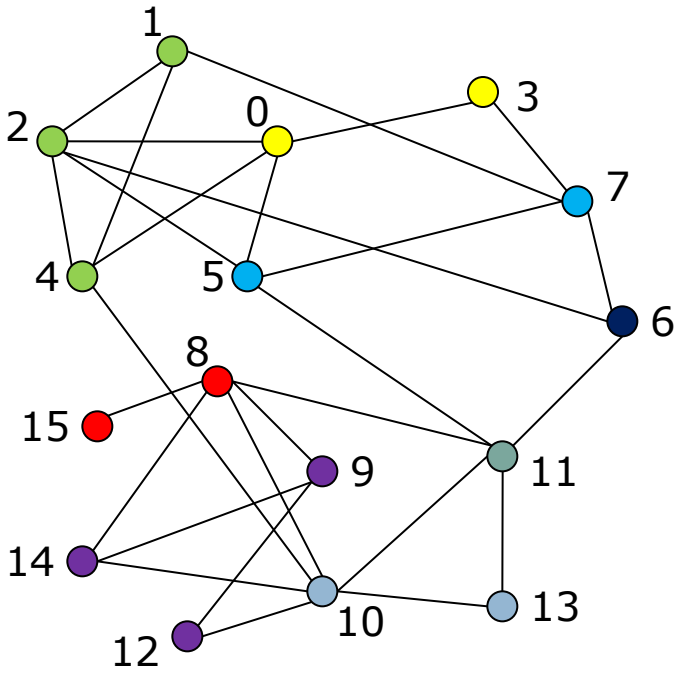
0 -> c[3]
1 -> c[4]
2 -> c[1,4]

Un exemple



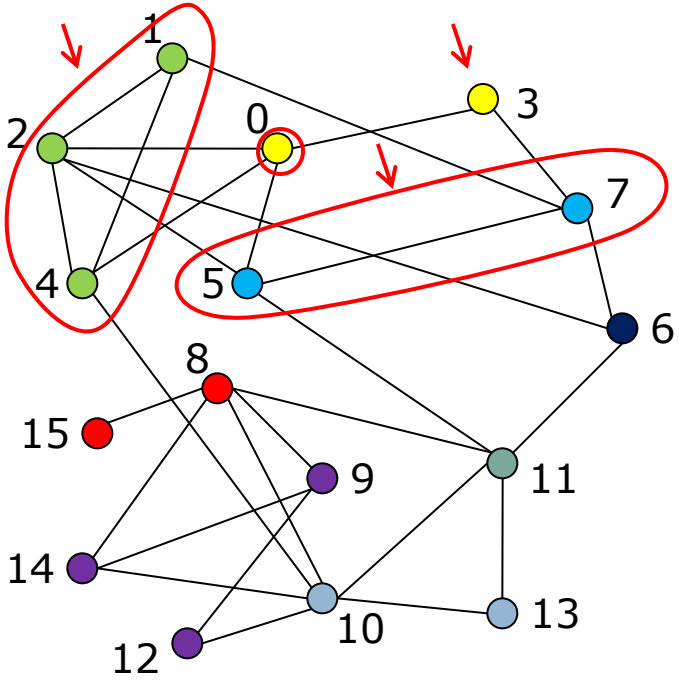
0 -> c[3]
1 -> c[4]
2 -> c[1,4]
3 -> c[0]

Un exemple



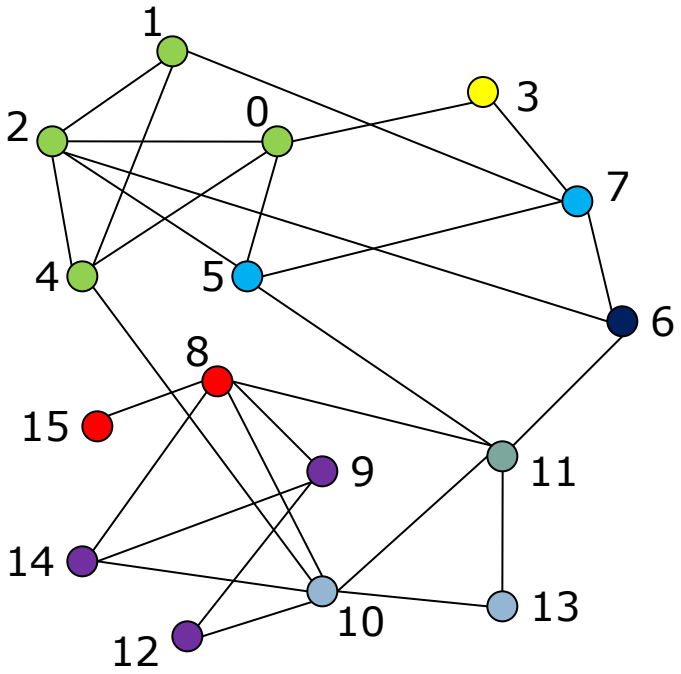
0 -> c[3]
1 -> c[4]
2 -> c[1,4]
3 -> c[0]
4 -> c[1]
5 -> c[7]
6 -> c[11]
7 -> c[5]
8 -> c[15]
9 -> c[12]
10 -> c[13]
11 -> c[10,13]
12 -> c[9]
13 -> c[10,11]
14 -> c[9,12]
15 -> c[8]

Un exemple



Passe 1 – Itération 2

Un exemple

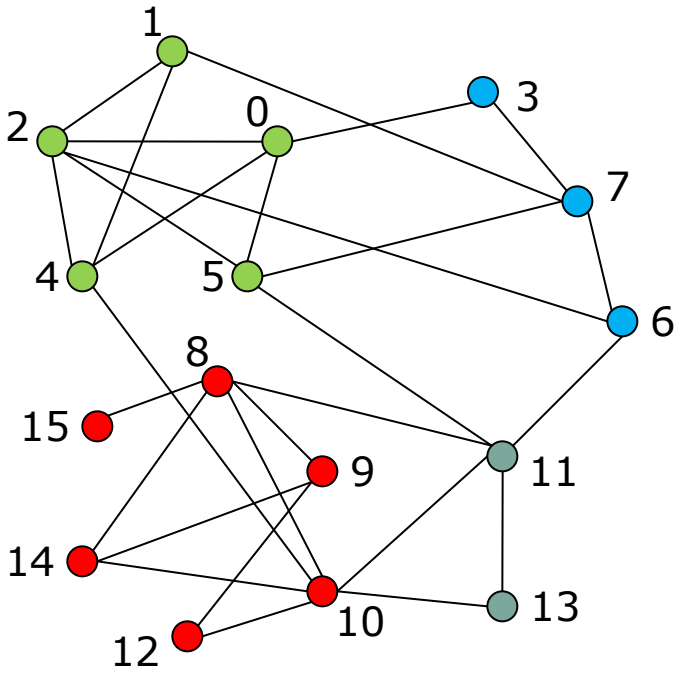


Passe 1 – Itération 2

0 -> c[4]

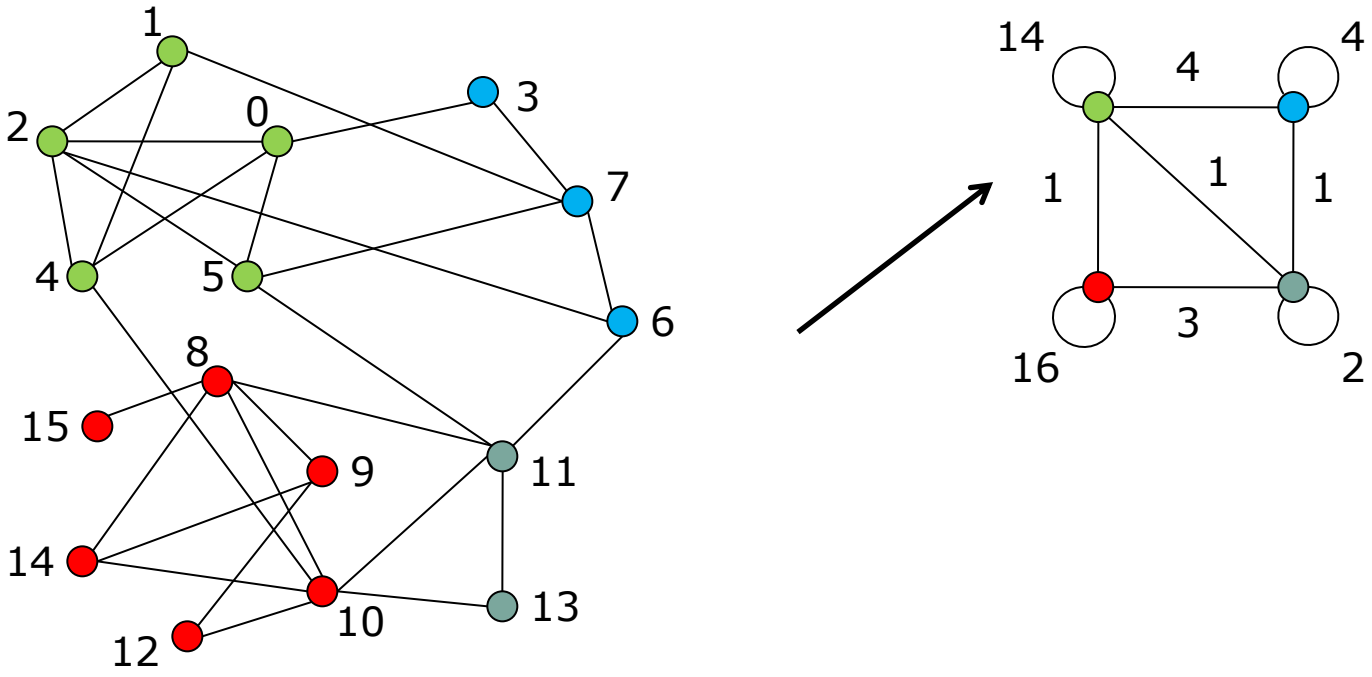
…

Un exemple

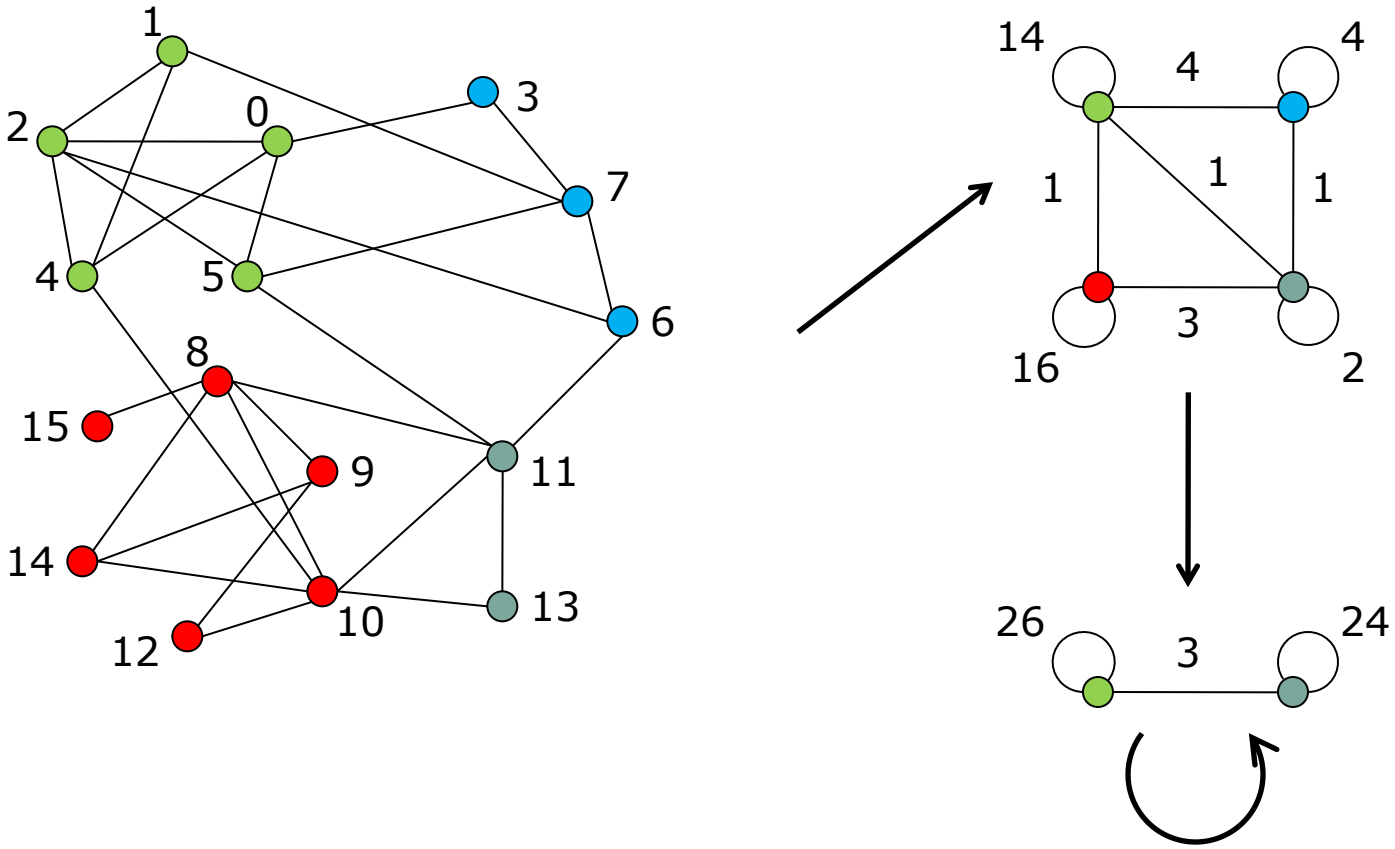


Après 4 itérations,
plus de changement

Un exemple

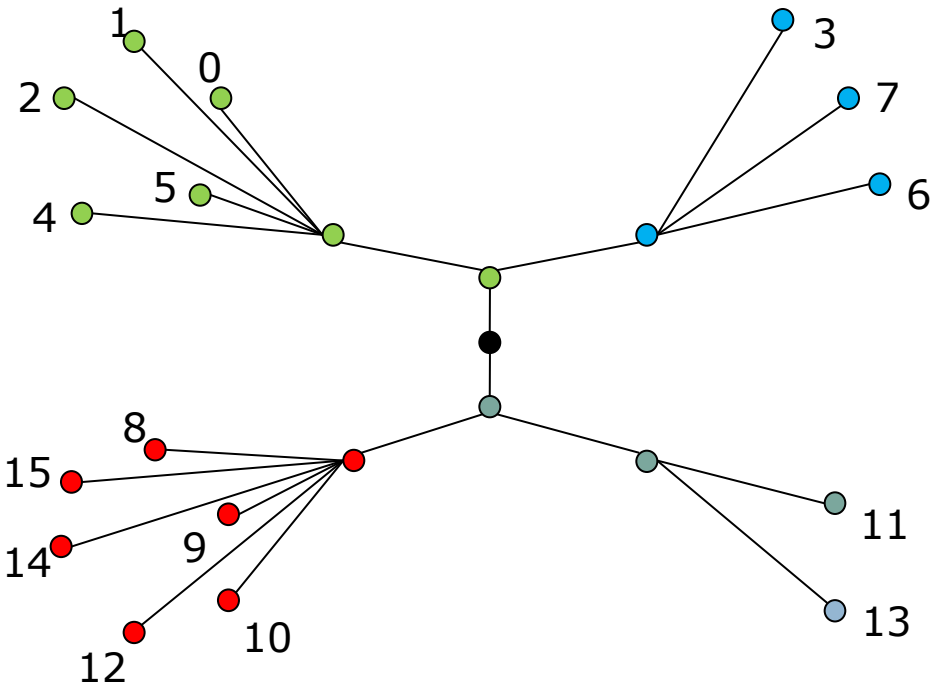


Un exemple



Un exemple

- Donne un dendrogramme non binaire.
  - Niveaux intermédiaires moins sujets aux problèmes de résolution limite.



L'algorithme formellement

- Suite de passes :
  - Chacune calcule un niveau de la hiérarchie.
  - Entrée : graphe (pondéré).
  - Sortie : graphe pondéré dont les sommets sont les communautés du graphe en entrée.

- Les passes sont appliquées récursivement.
- Arrête quand la modularité ne peut plus êtr améliorée.

L'algorithme formellement

- Chaque passe :
  - Au début chaque sommet forme une communauté.
  - Répéter de manière itérative :
    - Supprimer i de sa communauté.
    - Insérer i dans la communauté voisine qui maximise la modularité (approche locale gloutonne).
  - Arrêter quand un maximum local est atteint.

Résultats expérimentaux

	Karate	Arxiv	Internet	Web nd.edu	Belgian Phone Calls	Web UK-2005	Web Webbase01
	n=34/m=77	9k/24k	70k/351k	325k/1M	2.5M/6.3M	39M / 783M	118M/1B
Newman Clauset Moore	0s	3.6s	799s	5034s	-	-	-
Pons Latapy	0s	3.3s	575s	6666s	-	-	-
Wakita Tsurumi (estimé)	0s	0s	8s	52s	1279s	(3 jours)	-
Notre approche	**0s**	**0s**	**1s**	**2s**	**89s**	**515s**	**22mn**
	3 passes	5 passes	5 passes	5 passes	5 passes	4 passes	5 passes

Résultats expérimentaux

	Karate	Arxiv	Internet	Web nd.edu	Belgian Phone Calls	Web UK-2005	Web Webbase01
	n=34/m=77	9k/24k	70k/351k	325k/1M	2.5M/6.3M	39M / 783M	118M/1B
Newman Clauset Moore	0s 0.38	3.6s 0.772	799s 0.692	5034s 0.927	-	-	-
Pons Latapy	0s 0.42	3.3s 0.757	575s 0.729	6666s 0.895	-	-	-
Wakita Tsurumi (estimé)	0s	0s	8s	52s	1279s	(3 jours)	-
Notre approche	0s **0.42**	0s **0.813**	1s **0.781**	2s **0.935**	89s **0.774**	515s **0.979**	22mn **0.984**
	3 passes	5 passes	5 passes	5 passes	5 passes	4 passes	5 passes

Résultats expérimentaux

- Les graphes aux niveaux supérieurs sont petits :
  - Les premières passes sont les plus couteuses.
  - En général : première passe > 90% du temps de calcul.
- Il y a peu d'itérations par passe :
  - Seules les itérations de la première passe sont couteuses.
  - <33 pour tous les réseaux testés.
- Traiter un sommet est simple.

Modularité

$$Q = \frac{1}{2m} \sum_C \left[e_C - \frac{a_C^2}{2m} \right]$$

Liens avec une extrémité dans C

Liens dans C

- La contribution d'un sommet isolé est donc :

$$Q(i) = -\left(\frac{k_i}{2m} \right)^2$$

Degré de i

Déplacer un sommet

- Un sommet isolé ‘i’ peut être déplacé dans ‘C’ avec un gain :

$$\Delta Q(C,i) = \left[\frac{e_C + k_{i,C}}{2m} - \left(\frac{a_C + k_i}{2m} \right)^2 \right] - \left[\frac{e_C}{2m} - \left(\frac{a_C}{2m} \right)^2 - \left(\frac{k_i}{2m} \right)^2 \right]$$

Liens de i à C

- Ne dépend que de ‘i’ et ‘C’
- Complexité linéaire avec k_i

Structures de données

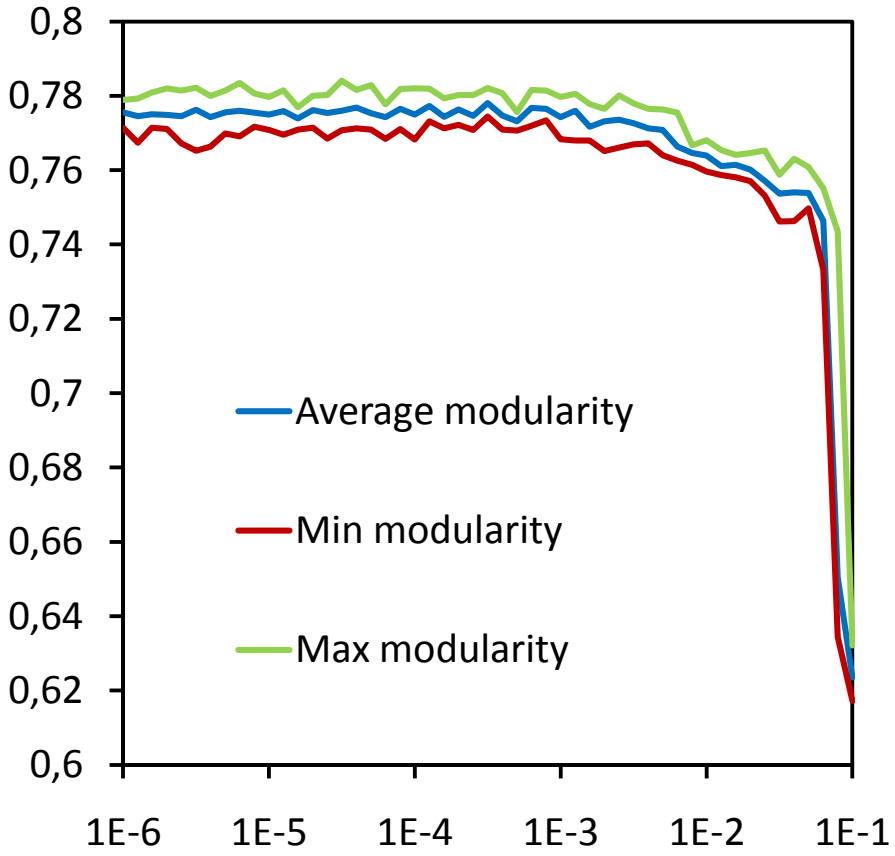
- À garder en mémoire :
  - Les listes d'adjacence (2m+n).
  - Les vecteurs e, a, node2comm (n chacun).
  - Total = 2m+4n : 100M liens, 1G liens :
    - 8.4 Go pour le graphe.
    - 1.2 Go pour les vecteurs.

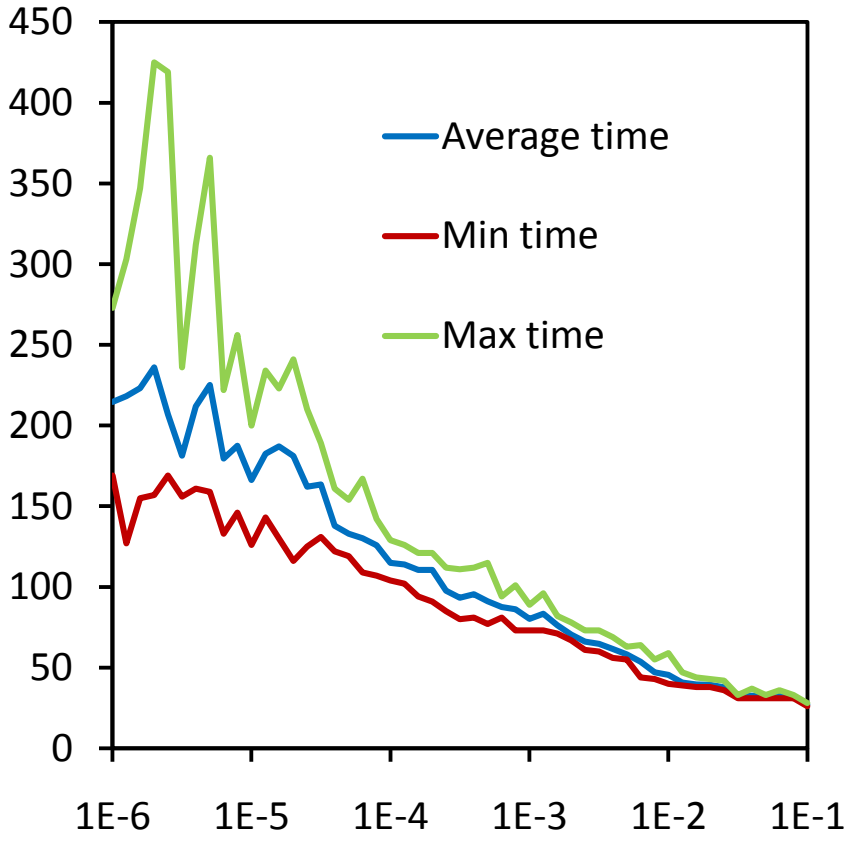
- L'algorithme est itératif :
  - Les listes d'adjacences peuvent être lues sur disque de manière séquentielle.
  - Permet de gérer de très grands graphes même si des machines de base (portable).

Heuristiques

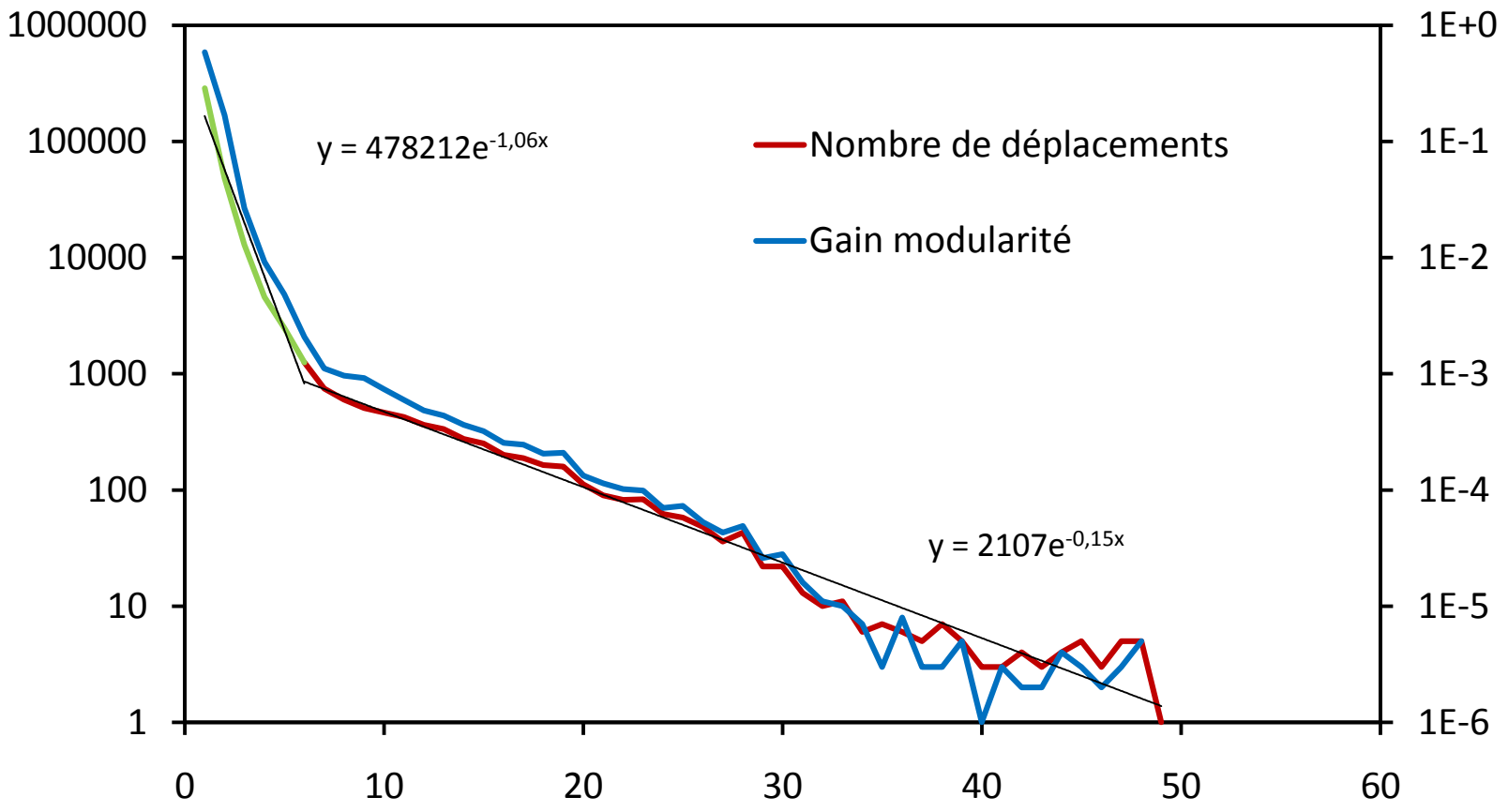
- L'algorithme peut être accéléré :
  - Les dernières itérations/passes apportent un gain marginal :
    - Arrêter quand le gain est inférieur à un seuil fixé.
  - Les sommets de degré 1 peuvent être supprimés avant le calcul :
    - Utile seulement sur les très grands graphes (millions de sommets).
  - Seuls quelques sommets (<10%) sont déplacés à chaque itération :
    - Un sommet statique n'est pas considéré à l'itération suivante.

Epsilon (mobistar)

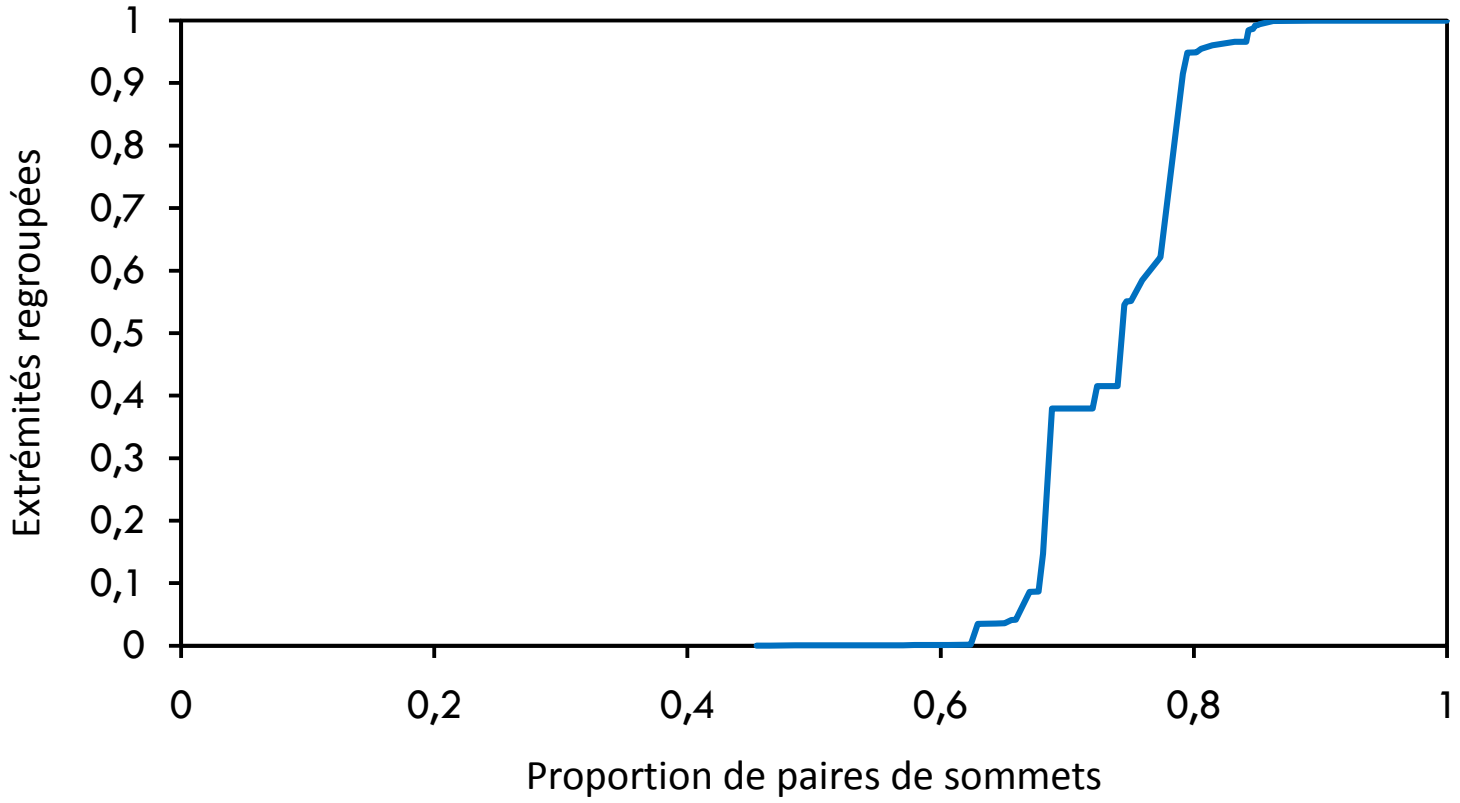




Nombre de déplacements



Stabilité des paires (karate)



Conclusions

- Techniques efficaces pour le calcul sur des graphes de très grande taille (millions/milliards de sommets/liens)
- Très bons résultats en terme de modularité

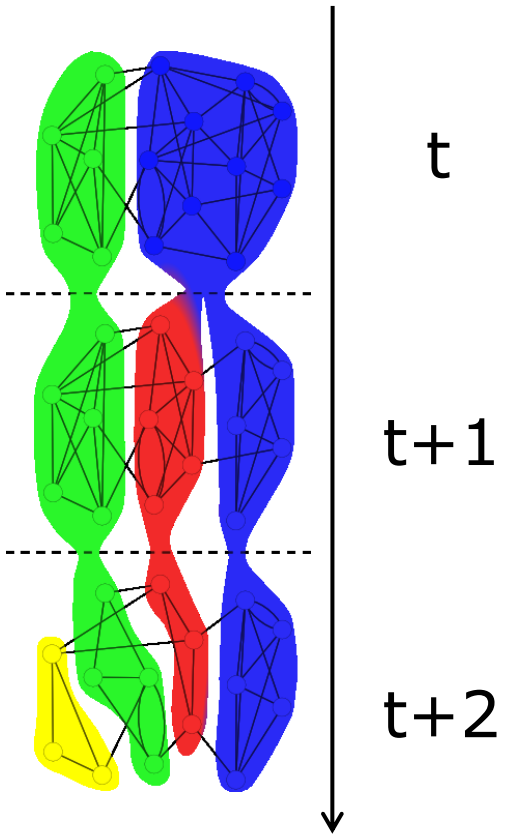
COURS SYRRES

COMMUNAUTÉS DYNAMIQUES

Jean-Loup Guillaume

Communautés dynamiques

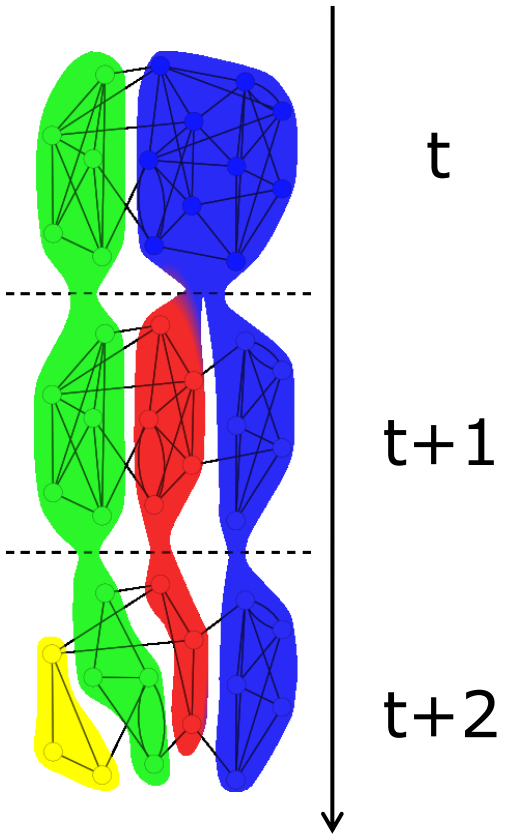
- Approche naturelle :
  - Calcul des communautés à chaque instant de manière indépendante.
  - Suivi des communautés d'un instantané à l'autre.



Communautés dynamiques

- Approche naturelle :
  - Calcul des communautés à chaque instant de manière indépendante.
  - Suivi des communautés d'un instantané à l'autre.

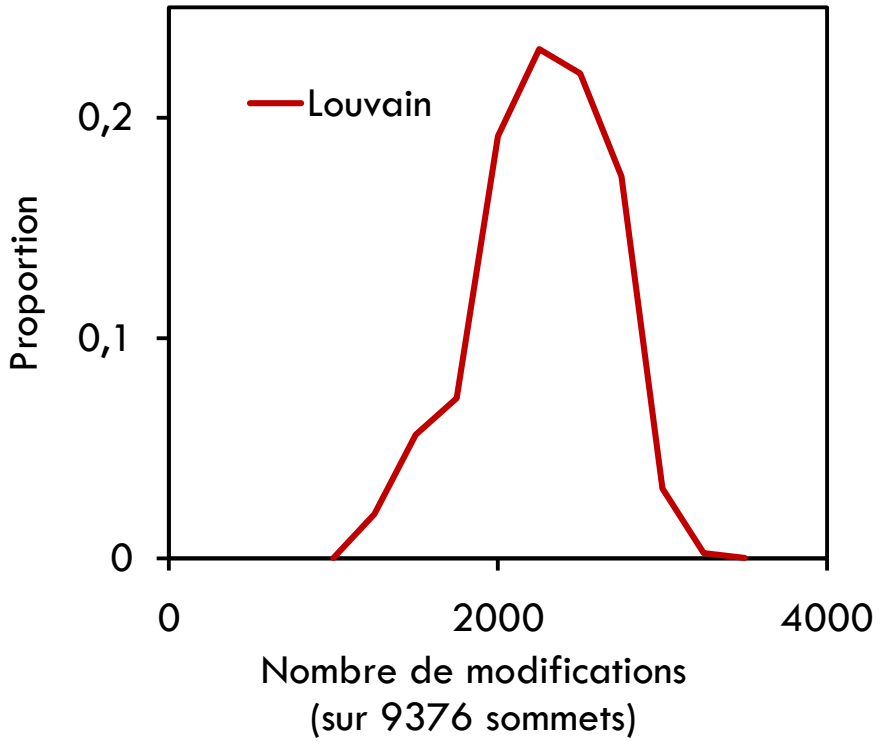
- Problèmes :
  - Complexité :
    - Taille du graphe.
    - Nombre d'instantanés.



Communautés dynamiques

- Approche naturelle :
  - Calcul des communautés à chaque instant de manière indépendante.
  - Suivi des communautés d'un instantané à l'autre.

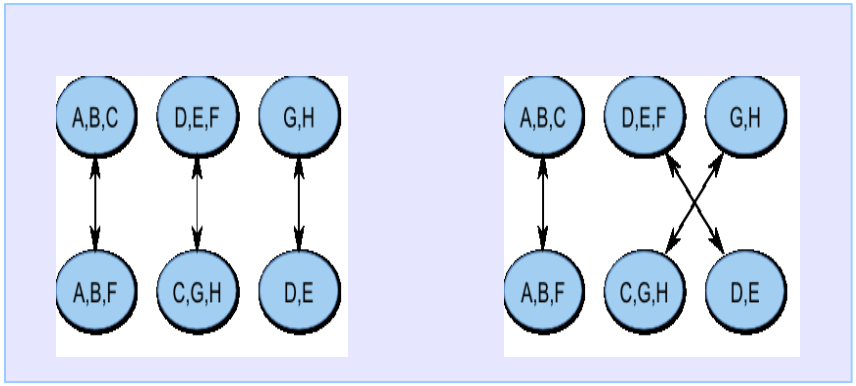
- Problèmes :
  - Complexité.
  - Stabilité :
    - Nombreuses partitions de même qualité.
    - Nombreuses modifications de la partition pour une petite modification de topologie.



Communautés dynamiques

- Approche naturelle :
  - Calcul des communautés à chaque instant de manière indépendante.
  - Suivi des communautés d'un instantané à l'autre.

- Problèmes :
  - Complexité.
  - Stabilité.
  - Suivi :
    - Associer les groupes à t et à t+1.



Précédentes approches

- Modifier la définition de communauté pour simplifier le suivi
  - Cliques avec recouvrement.

Palla, Barabasi and Vicsek, Nature 2007

Précédentes approches

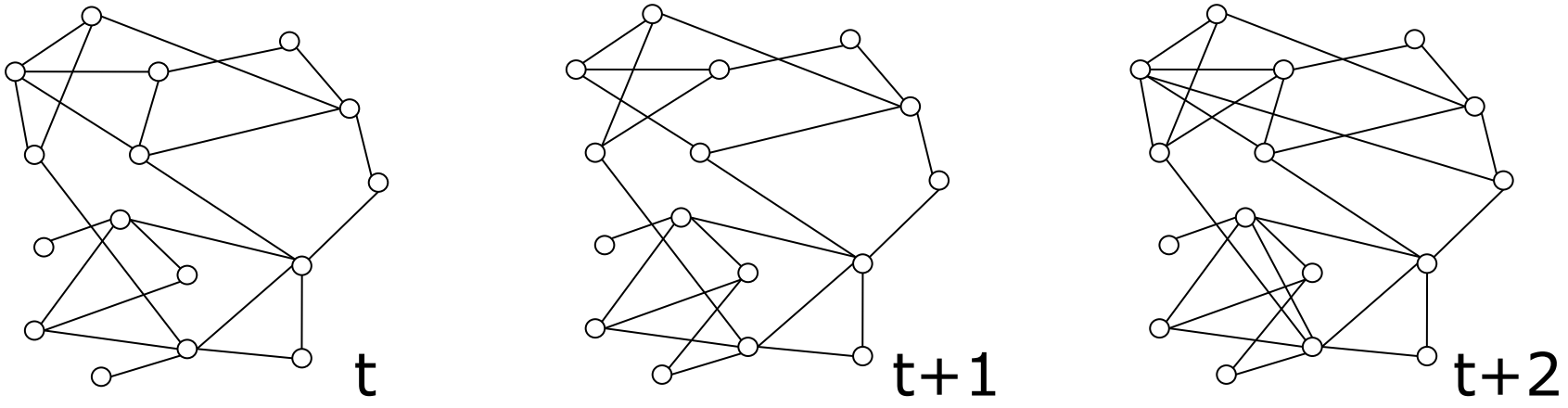
- Modifier la définition de communauté pour simplifier le suivi
- Se restreindre aux zones très stables
  - Communautés non affectées par la dynamique.

Hopcroft, Khan, Kulis and Selman, PNAS 2004

Précédentes approches

- Modifier la définition de communauté pour simplifier le suivi
- Se restreindre aux zones très stables
- Construire un graphe temporel
  - Ajouter des liens temporels entre les instantanés.
  - Utiliser n'importe quel algorithme pour les graphes statiques.

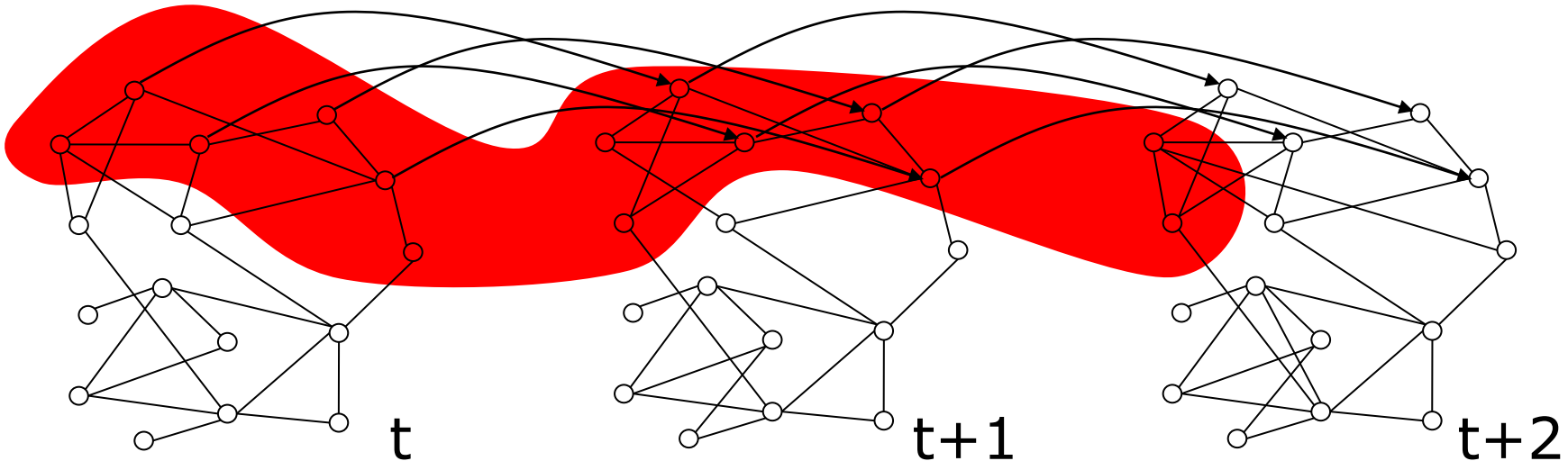
Jdidia, Robardet and Fleury, ICDIM 2007



Précédentes approches

- Modifier la définition de communauté pour simplifier le suivi
- Se restreindre aux zones très stables
- Construire un graphe temporel
  - Ajouter des liens temporels entre les instantanés.
  - Utiliser n'importe quel algorithme pour les graphes statiques.

Jdidia, Robardet and Fleury, ICDIM 2007



Précédentes approches

- Modifier la définition de communauté pour simplifier le suivi
- Se restreindre aux zones très stables
- Construire un graphe temporel
- Modifier la fonction de qualité
  - Prise en compte de la qualité instantanée et de la stabilité.

D. Chakrabarti, R. Kumar, and A. Tomkins, SIGKDD 2006.

Y. Lin et al, Transactions on Knowledge Discovery from Data

Précédentes approches

- Modifier la définition de communauté pour simplifier le suivi
- Se restreindre aux zones très stables
- Construire un graphe temporel
- Modifier la fonction de qualité
- Approche fouille de données
  - Recherche de groupes de liens denses et fréquents.
  - Suivi d'individus dans les groupes.

Borgnat, Fleury, Guillaume, Robardet and Scherrer, Computer Networks 2008

Précédentes approches - limitations

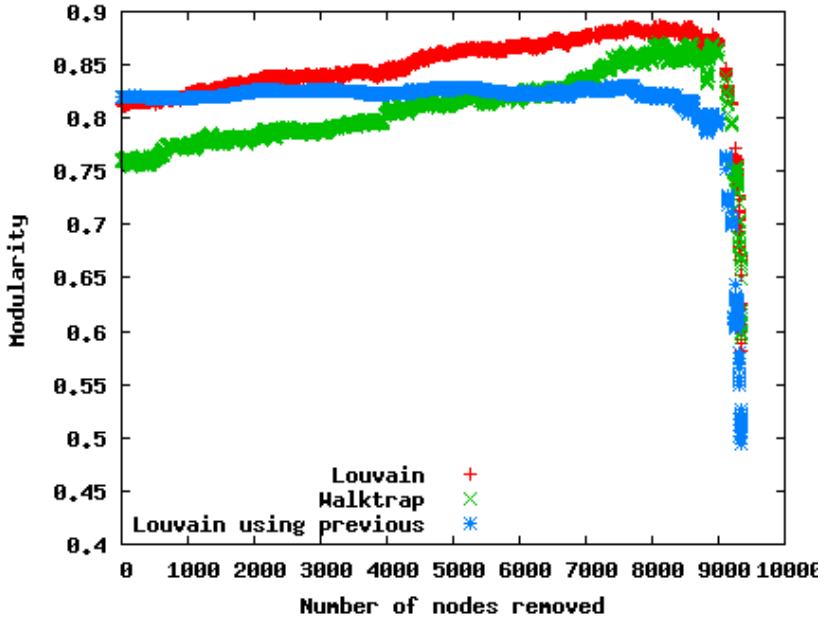
- Modifier la définition de communauté pour simplifier le suivi
  - Une seule proposition très restrictive.
- Se restreindre aux zones très stables
  - Proportion de zone stable et quid du reste du graphe ?
- Construire un graphe temporel
  - Plusieurs façons de construire le graphe et aucune vraiment satisfaisante.
- Modifier la fonction de qualité
  - Définition de qualité temporelle non triviale.
- Approche fouille de données
  - Problème de passage à l'échelle.

Précédentes approches

- Modifier la définition de communauté -> cœurs.
  - Une seule proposition très restrictive.
- Se restreindre aux zones très stables -> stabilité/cœurs.
  - Proportion de zone stable et quid du reste du graphe ?
- Construire un graphe temporel :
  - Plusieurs façons de construire le graphe et aucune vraiment satisfaisante.
- Modifier la fonction de qualité -> intégrale de modularité.
  - Définition de qualité temporelle non triviale.
- Approche fouille de données :
  - Problème de passage à l'échelle.

Approches étudiées - stabilisation

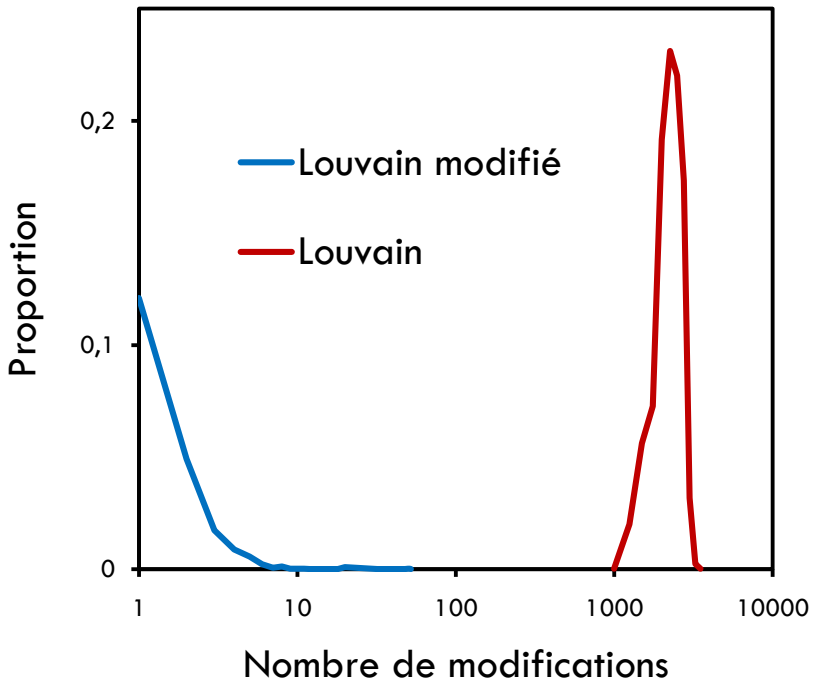
- Stabilité imposée dans l'algorithme :
  - Le calcul à l'instant 't' prend en compte celui de l'instant 't-1'
  - Utilisation de Louvain avec initialisation
- Résultats préliminaires :
  - Perte de qualité minimale.



Approches étudiées - stabilisation

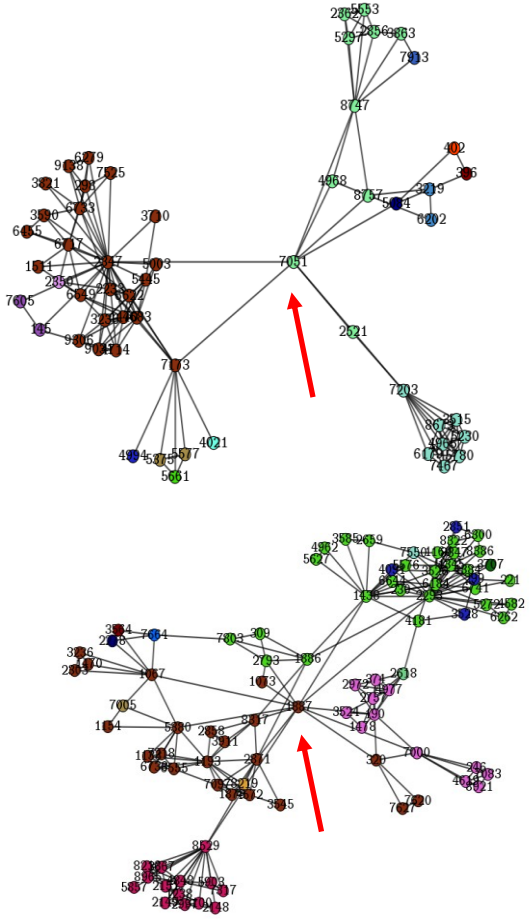
- Stabilité imposée dans l'algorithme :
  - Le calcul à l'instant 't' prend en compte celui de l'instant 't-1'
  - Utilisation de Louvain avec initialisation

- Résultats préliminaires :
  - Perte de qualité minimale.
  - Très fort gain de stabilité.



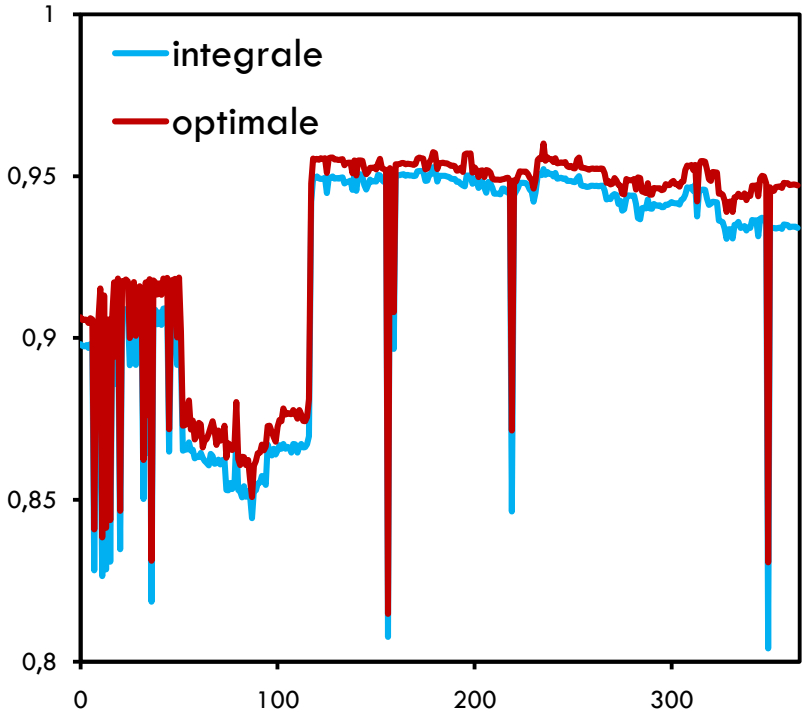
Approches étudiées - stabilisation

- Stabilité imposée dans l'algorithme :
  - Le calcul à l'instant 't' prend en compte celui de l'instant 't-1'
  - Utilisation de Louvain avec initialisation
- Résultats préliminaires :
  - Perte de qualité minimale.
  - Gain de stabilité.
- Pour aller plus loin :
  - Compromis stabilité/qualité.
  - Détection d'événements :
    - Liens entre les modifications de topologie et les modifications de la partition.



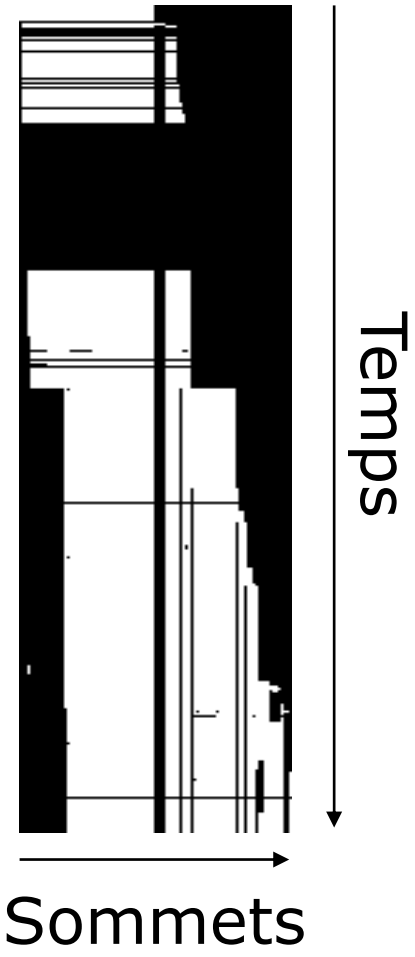
Approches étudiées - intégrale

- Optimisation globale :
  - Une seule partition sur toute la durée.
  - Complètement stable.
- Résultats préliminaires :
  - Définition + algorithme de calcul.
  - Qualité proche de l'optimale.



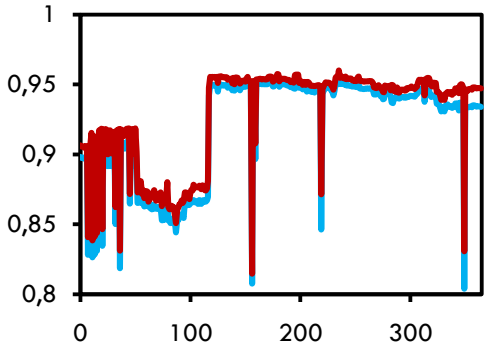
Approches étudiées - intégrale

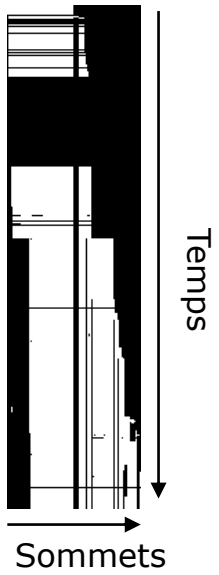
- Optimisation globale :
  - Une seule partition sur toute la durée.
  - Complètement stable.
- Résultats préliminaires :
  - Définition + algorithme de calcul.
  - Qualité proche de l'optimale.
  - Suivi de communautés.



Approches étudiées - intégrale

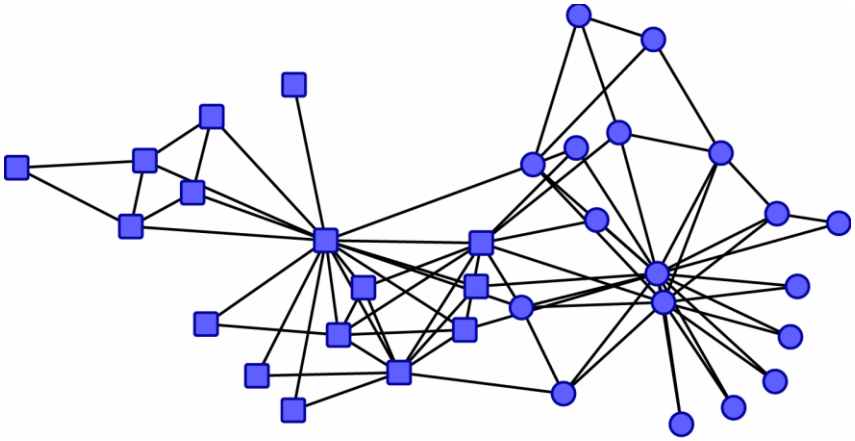
- Optimisation globale :
  - Une seule partition sur toute la durée.
  - Complètement stable.
- Résultats préliminaires :
  - Définition + algorithme de calcul.
  - Qualité proche de l'optimale.
  - Suivi de communautés.
- Pour aller plus loin :
  - Fenêtre de calcul / effets de bord.
  - Détection d'événements.





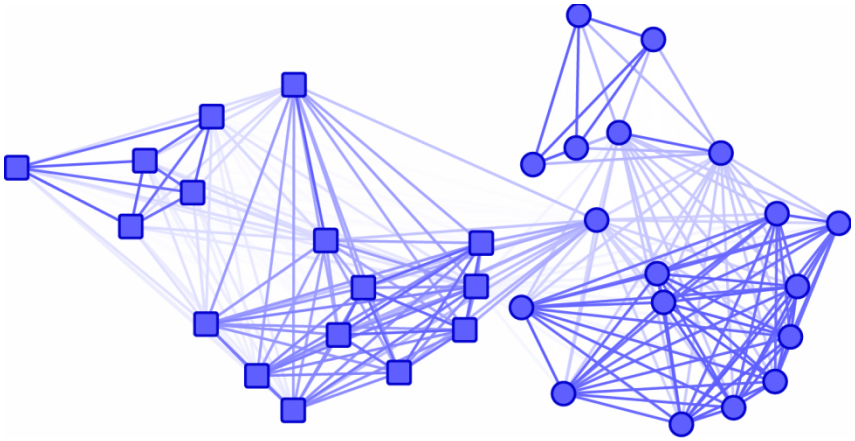
Approches étudiées – cœurs

- Pour un même graphe :
  - Plusieurs partitions de qualité similaire.
  - Recherche de similarités entre partitions.
- Résultats préliminaires :
  - Utilisation de l'algo de Louvain :
    - Non déterministe -> ≠ partitions.



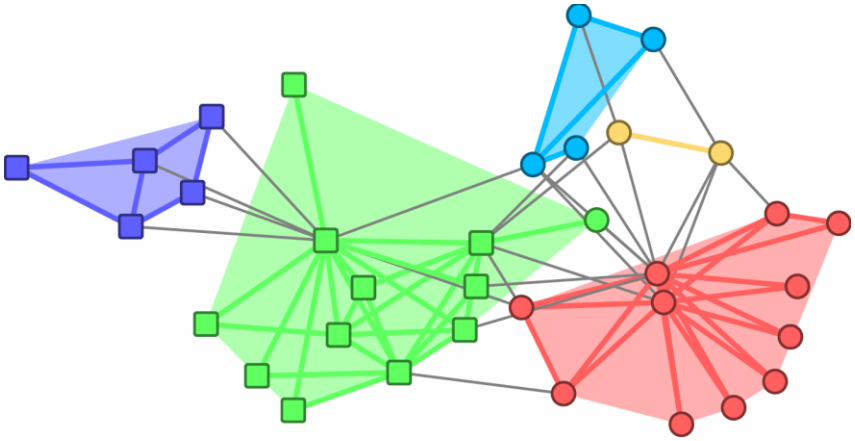
Approches étudiées – cœurs

- Pour un même graphe :
  - Plusieurs partitions de qualité similaire.
  - Recherche de similarités entre partitions.
- Résultats préliminaires :
  - Utilisation de l'algo de Louvain :
    - Non déterministe -> ≠ partitions.
  - Graphe de similarité :
    - Proximité = accord entre les partitions.



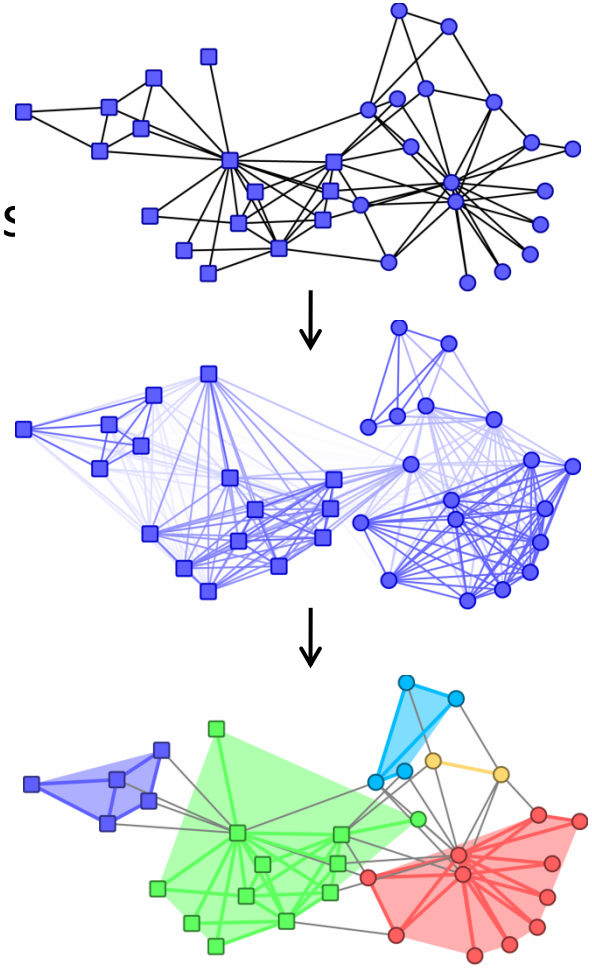
Approches étudiées – cœurs

- Pour un même graphe :
  - Plusieurs partitions de qualité similaire.
  - Recherche de similarités entre partitions.
- Résultats préliminaires :
  - Utilisation de l'algo de Louvain :
    - Non déterministe -> ≠ partitions.
  - Graphe de similarité :
    - Proximité = accord entre les partitions.
  - Seuil sur la proximité -> cœurs.



Approches étudiées – cœurs

- Pour un même graphe :
  - Plusieurs partitions de qualité similaire.
  - Recherche de similarités entre partitions
- Résultats préliminaires :
  - Utilisation de l'algo de Louvain :
    - Non déterministe -> ≠ partitions.
  - Graphe de similarité :
    - Proximité = accord entre les partitions.
  - Seuil sur la proximité -> cœurs.
- Pour aller plus loin :
  - Seuil de proximité pertinent.
  - Analyse / dynamique des cœurs.



Conclusions

- Les méthodes classiques ne sont pas utilisables directement :
  - Problème de stabilité.

- Trois approches permettant de stabiliser :
  - En prenant en compte le passé.
  - En calculant une seule partition.
  - En se concentrant sur des zones très stables (cœurs).

Conclusions

- Les méthodes classiques ne sont pas utilisables directement :
  - Problème de stabilité.
- Trois approches permettant de stabiliser :
  - En prenant en compte le passé.
  - En calculant une seule partition.
  - En se concentrant sur des zones très stables (cœurs).
- … mais pas uniquement de stabiliser.
  - De nouvelles approches pour la détection de communautés.
- Objectifs à terme :
  - Obtenir des méthodes de suivi de communautés dynamiques.
  - Comprendre/analyser la structure des communautés dynamiques.