Université de Bordeaux
Master 2 BioInformatique

Année 2018 - 2019

TP de Visualisation d'Information
- Algorithme de détection de communautés -

Objectif : Nous allons lors de ce TP mettre en place un algorithme de détection de communautés dans un graphe de réseau social (vous pourrez générer un graphe en utilisant l'une des méthodes proposées dans l'interface). Celui-ci sera programmé directement dans l'interface de Tulip à l'aide de l'IDE Python.

Fichiers fournis : Vous trouverez à l'adresse :
www.labri.fr/perso/bourqui/downloads/cours/Master/2018/TP3
un répertoire contenant les fichiers nécessaires pour démarrer le TP. Celui-ci contient le fichier python de départ.

Principe de l'algorithme de détection de communautés

Afin d'identifier les communautés d'un graphe, vous devrez dans ce TP mettre en place le pipeline de traitements suivant (chacun de ces points est détaillé plus tard) :

1. Calcul pour chaque arête d'une mesure (comprise entre 0 et 1) indiquant si elle appartient à une communauté ou si elle relie deux communautés
2. Pour une valeur de borne allant de 0 à 1 :
   a) Filtrage des arêtes dont la mesure est plus petite que la borne
   b) Calcul des composantes connexes (celles-ci représenteront les communautés du graphe initial)
   c) Évaluation de la "qualité" des communautés

Cela permettra de trouver la "meilleure" valeur pour la borne

3. Extraction des sous-graphes correspondant aux communautés identifiées

Calcul de la mesure sur les arêtes

L'idée de la mesure (métrique) que vous devrez implémenter dans ce TP est de comparer le nombre de cycles de taille k ($k>2$) auxquels une arête appartient au nombre maximum de tels cyles (étant donnés les degrés des sommets du graphe).

Afin de simplifier le problème, nous ne nous intéresserons ici qu'aux cycles de taille 3 (et en bonus, aux cycles de taille 4, 5, *etc*).

Question 1 : Écrire une fonction calculer_c3 qui calcule et retourne pour l'arête e donnée la valeur $|C_3(e)|/ C_3^{max}(e)$ où $|C_3(e)|$ est le nombre de cycles de taille 3 passant par l'arête e, et $C_3^{max}(e)$ est le nombre maximum possible de tels cycles (étant donnés les degrés de ses extrémités).

Pour calculer le nombre de cycles de taille 3 passant par une arête e=(u,v), il suffit de compter combien les sommets u et v ont de voisins communs (*i.e.* la taille de l'intersection des deux voisinages). Le nombre maximum possible de cycles quant à lui est simplement la taille de l'union des deux voisinages de u et de v moins 2 (car u et v appartiennent respectivement aux voisinages de v et de u).

Question 2 : Écrire une fonction calculer_mesure qui calcule la mesure de la question précédente pour chaque arête du graphe et la stocke dans une propriété passée en paramètre.

Calcul de communautés

Afin de calculer la "meilleure" borne, *i.e.* la borne permettant de former des communautés denses (en nombre d'arêtes) tout en ayant peu d'arêtes entre les communautés, vous devrez implémenter différentes fonctions.

Filtrage des arêtes
Question 3 : Écrire une fonction filtrer_arete qui supprime du graphe en paramètre toute arête dont la mesure est inférieure à la borne. En fonction de la borne choisie, supprimer des arêtes va "déconnecter" le graphe, on considérera alors chaque composante connexe comme une communauté.

Calcul des composantes connexes
Question 4 : Écrire une fonction calculer_composantes_connexes qui calcule les composantes connexes du graphe g et extrait un sous-graphe par composante connexe (penser à utiliser les plugins et l'API).

Calcul des communautés
Question 5 : Écrire la fonction trouver_groupes qui permet de calculer les communautés du graphe passé en paramètre en fonction de la borne borne et extrait ces communautés (sous la forme de sous-graphes).

Évaluation de la qualité de la décomposition en communautés

Afin d'évaluer la qualité d'une décomposition en communautés d'un graphe, un certain nombre de mesures de qualité ont été définies. Dans ce TP, nous nous intéresserons à la mesure MQ. Le principe de cette mesure est de comparer la densité d'arêtes interne aux communautés à la densité d'arêtes entre les communautés (plus la densité intra-communauté est forte et la densité inter-communautés est faible, plus la décomposition est considérée de bonne qualité).

Considérons que le graphe ait été décomposé en C = {c_1, c_2, …, c_k} communautés, MQ est définie comme suit :

$$MQ = MQ^+ - MQ^-$$

où

$$MQ^+ = \frac{1}{k} \sum_{i=0}^{i=k} densite(c_i)$$

et densite(c_i) est la densité d'une communauté, *i.e.*le ratio entre le nombre d'arêtes dans la communauté et le nombre d'arêtes d'un graphe complet de $|c_i|$ sommets.
Et,

$$MQ- = \frac{1}{k(k-1)/2} \sum_{i=0}^{i=k-1} \sum_{j=i+1}^{j=k} densite(c_i, c_j)$$

et densite(c_i, c_j) est le ratio entre le nombre d'arêtes reliant des sommets de c_i et de c_j et le nombre maximum possible de telles arêtes, *i.e.* $|c_i||c_j|$.

Par définition, cette mesure est comprise entre -1 et 1 (-1 lorsque chaque paire de communautés forme un graphe biparti complet et aucune arête n'est interne à une communauté; 1 lorsque chaque communauté forme une clique et aucune arête ne relie deux communautés).

Calcul de densité intra-communauté
Question 6 : Écrire une fonction densite_intra_c qui calcule et retourne la densité interne de la communauté passée en paramètre (sous forme d'un graphe).

Question 7 : Écrire une fonction densite_intra qui calcule et retourne la densité interne moyenne des communautés du graphe g (*i.e.* MQ^+). On considérera que les communautés de g sont les sous-graphes de g.

Calcul de densités inter-communautés
Question 8 : Écrire une fonction densite_inter_c1_c2 qui calcule et retourne la densité d'arêtes entre les communautés c_i et c_j

Question 9 : Écrire une fonction densite_inter qui calcule et retourne la densité d'arêtes moyenne entre les communautés de g (*i.e.* MQ^-). On considérera que les communautés de g sont les sous-graphes de g.

Calcul de la "meilleure" borne
Question 10 : Écrire une fonction evaluer_qualite qui calcule et retourne la qualité de la décomposition de g en communautés. On considérera que les communautés de g sont les sous-graphes de g.

Question 11 : Écrire une fonction trouver_meilleure_borne qui en faisant varier une borne (par pas de pas) entre 0 et 1 retourne la borne permettant de maximiser la qualité de la décomposition en communautés du graphe g (la mesure sur les arêtes ayant déjà été calculée).

Fonction principale

Question 12 : Écrire la fonction principale du programme permettant de calculer la métrique, de trouver la meilleure borne, de calculer les communautés correspondantes.