Mathématiques et sciences humaines
Mathematics and social sciences

187 | Automne 2009
Journée 2007 de la Société Francophone de Classification

Recherche de classes empiétantes dans un graphe : application aux réseaux d'interactions entre protéines

Computation of overlapping classes in a graph: application to protein-protein interactions networks

Lucile Denoeud-Belgacem

Édition électronique
URL : http://journals.openedition.org/msh/11106
DOI : 10.4000/msh.11106
ISSN : 1950-6821

Éditeur
Centre d'analyse et de mathématique sociales de l'EHESS

Édition imprimée
Date de publication : 30 décembre 2009
Pagination : 7-42
ISSN : 0987-6936

Référence électronique
Lucile Denoeud-Belgacem, « Recherche de classes empiétantes dans un graphe : application aux réseaux d'interactions entre protéines », *Mathématiques et sciences humaines* [En ligne], 187 | Automne 2009, mis en ligne le 15 décembre 2009, consulté le 20 avril 2019. URL : http://journals.openedition.org/msh/11106 ; DOI : 10.4000/msh.11106

Math. & Sci. hum. / Mathematics and Social Sciences (47e année, n° 187, *2009*(3), p. 7–42)

RECHERCHE DE CLASSES EMPIÉTANTES DANS UN GRAPHE : APPLICATION AUX RÉSEAUX D'INTERACTIONS ENTRE PROTÉINES

Lucile DENŒUD-BELGACEM[1]

RÉSUMÉ – *Cet article présente une méthode de classification empiétante permettant de mettre en évidence des zones denses en arêtes dans un graphe. On cherche plus précisément à extraire du graphe des sous-graphes dont la densité en arêtes soit élevée par rapport à la densité du graphe entier, ces sous-graphes pouvant avoir des sommets en commun. Cette méthode est appliquée à un problème issu de la biologique : l'annotation des protéines. Les graphes considérés traduisent alors des interactions observées entre les protéines. Partant du principe biologique que des protéines impliquées dans une même fonction cellulaire interagissent, les sous-graphes obtenus par l'application de la méthode de classification empiétante aux réseaux d'interactions donnent des indications sur les fonctions des protéines constituant ces sous-graphes, ce qui permet de fournir une aide informatique à la prédiction de fonctions inconnues de certaines protéines. Le caractère empiétant autorisé par la méthode présentée ici permet en particulier de prendre en compte le fait que les protéines peuvent être impliquées chacune dans plusieurs fonctions cellulaires.*

MOTS-CLÉS – Classification empiétante, Classification par densité, Graphe, Nuées dynamiques, Partitionnement, Réseau d'interactions entre protéines

SUMMARY – Computation of overlapping classes in a graph : application to protein-protein interactions networks

This article describes a method of overlapping classification, in order to compute zones which are dense in edges in a graph. More precisely, the aim is to compute subgraphs in which the density of edges is large compared to the edge-density of the whole graph. These subgraphs may share common vertices. This method is applied to a problem arising in biology : the annotation of proteins. The graphs then represent the observed interactions between proteins. Thanks to the biological principle that proteins involved in the same cellular function interact, the subgraphs provided when the method is applied to the protein-protein interactions networks provide information about the functions of proteins belonging to these subgraphs. This provides a computer-aided tool for the prediction of unknown functions of some proteins. The overlapping allowed by the method depicted here makes it possible to take into account the fact that each protein may be involved into several cellular functions.

KEYWORDS – Classification by density, Graph, k-means, Overlapping classification, Partitioning, Protein-protein interactions networks

[1] FuturMaster, 696 rue Yves Kermen, 92100 Boulogne-Billancourt, lucile.belgacem@futurmaster.com

1. INTRODUCTION

Dans cet article, nous proposons une méthode permettant de mettre en évidence des zones denses en arêtes dans un graphe. Il s'agit en fait de déterminer une classification empiétante des sommets du graphe de façon que les sous-graphes induits par ces classes soient relativement denses en arêtes par rapport au graphe global. Cette étude est motivée par une problématique biologique : l'annotation des protéines.

Dans cette introduction, nous commençerons donc par situer ce travail dans le contexte biologique, en présentant brièvement le problème de l'annotation des protéines, puis les travaux antérieurs s'inscrivant dans une démarche similaire à la nôtre. Enfin nous décrirons plus précisément l'objectif et les enjeux de la méthode proposée.

La partie 2 sera consacrée à la description de la méthode de classification empiétante. Celle-ci est constituée de trois étapes : création des noyaux initiaux des classes (partie 2.1), améliorations de ces noyaux à l'aide d'une méthode de nuées dynamiques (partie 2.2), extension des noyaux permettant l'empiétement des classes ainsi construites (partie 2.3). Ces étapes nécessitent la définition de plusieurs outils mathématiques : fonctions de densité locale (partie 2.4), distances dans les graphes (partie 2.5), critères d'extension (partie 2.6).

Les composants de cette méthode sont ensuite testés dans la partie suivante à l'aide de graphes aléatoires. On décrit d'abord la façon dont sont engendrés les graphes (partie 3.1) ; ceux-ci permettent d'analyser le comportement des différentes étapes de la méthode (parties 3.2 à 3.4). Cette étude permet d'éliminer certaines variantes et de ne retenir que celles qui seront appliquées à des données réelles.

Cette application à des données réelles fait l'objet de la partie 4. On y analyse les propriétés des classes obtenues d'abord d'un point de vue combinatoire (partie 4.1) puis d'un point de vue biologique (partie 4.2).

Une courte conclusion (partie 5) résume les principaux éléments de l'article.

1.1. CONTEXTE BIOLOGIQUE[2]

Après le séquençage de la majorité des génomes des organismes utilisés en laboratoire, les biologistes sont face à un nouveau défi : comprendre, à grande échelle, la fonction des composants codés par le génome et notamment des protéines. Sachant qu'aucune protéine n'assure seule sa fonction et que la vie de chaque organisme dépend de dizaines de milliers d'interactions entre protéines spécifiques, il est nécessaire de décrire en premier lieu les interactions moléculaires entre les protéines afin d'appréhender les mécanismes complexes survenant au sein de la cellule et de l'organisme.

La fonction d'une protéine (ou du gène qui la code) se définit à plusieurs niveaux, correspondant aux divers niveaux d'organisation biologique, de la macromolécule à la population d'organismes. À un niveau moléculaire, elle correspond à son action

[2]Cf. par exemple [Alberts *et al.*, 1995] ou [Etienne, Millot, 1998] pour des ouvrages génériques du domaine.

biochimique (catalyse d'une réaction chimique, interaction avec d'autres molécules, etc.). À un deuxième niveau, on parle de la fonction cellulaire d'une protéine, qui ne peut être prédite par la fonction biochimique. Elle résulte des interactions entre entités moléculaires et correspond à un processus biologique dans lequel la protéine est impliquée (voies métaboliques, cascades de signalisation, etc.). La protéine peut aussi être vue comme le composant d'une entité plus large (membrane, ribosome, noyau, etc.). Trois autres niveaux peuvent ensuite être définis pour caractériser la fonction d'une protéine au sein d'un tissu ou d'un organe, d'un organisme entier et d'une population d'organismes. Notons qu'une protéine peut avoir plusieurs fonctions moléculaires, qui participent à plusieurs processus biologiques et peuvent correspondre à différents composants cellulaires.

Dans la hiérarchie des fonctions, le passage de la fonction moléculaire à un niveau cellulaire est assuré par les interactions physiques directes entre les protéines. Les protéines interagissent de façon dynamique pour former des complexes protéiques. Ces interactions peuvent avoir une durée particulière et être spécifiques à un compartiment cellulaire, à un type cellulaire, ou à un stade de développement. Parallèlement, les interactions assurées par une protéine donnée dépendent de caractéristiques intrinsèques à cette protéine : sa taille, sa structure, la spécificité de ses sites d'interactions ou sa localisation cellulaire.

Aujourd'hui, le décryptage des génomes rend disponible la séquence d'un très grand nombre de protéines pour lesquelles nous ne disposons que de peu d'informations fonctionnelles. En effet une moyenne de 30-35% de ces protéines est de fonction cellulaire inconnue.

Jusqu'à une époque très récente, les fonctions des protéines ont été déterminées par des analyses moléculaires (similarité de séquence du gène avec un gène déjà décrit, similitude de structure tridimensionnelle de la protéine avec une structure connue), par la localisation sub-cellulaire et les modifications post-traductionnelles de la protéine, par l'étude de l'expression du gène dans l'espace et dans le temps ou encore par l'observation des effets de l'altération ou de la délétion du gène. Les nouvelles technologies comme le crible double-hybride à haut débit permettent désormais de déterminer systématiquement les interactions moléculaires sur des protéomes entiers d'organismes simples. Pour ces organismes, les données de la littérature et les résultats de ces expériences permettent la construction d'une carte détaillée des interactions protéine-protéine. Ces données offrent un nouvel outil pour la détermination des fonctions des protéines (ou *annotation*) car les données d'interactions disponibles pour une protéine renseignent sur sa fonction, indépendamment de toute information de séquence ou de structure. Il est en effet vraisemblable que des protéines impliquées dans une même fonction cellulaire interagissent et, par conséquent, plus les protéines ont d'interacteurs communs, plus il est probable que leurs fonctions soient reliées.

Les réseaux d'interaction considérés ici correspondent à des graphes non orientés et non pondérés (*graphes d'interactions entre protéines*) dans lesquels les sommets sont les protéines et dans lesquels il existe une arête entre deux sommets si les protéines correspondantes interagissent. Ces graphes, impliquant quelques milliers de protéines, ne sont pas aisés à manipuler intuitivement et les outils nécessaires à

l'analyse de ces données complexes sont encore rares. C'est dans ce cadre que nous avons développé une nouvelle méthode de classification fonctionnelle des protéines fondée sur la recherche de zones denses en arêtes dans ces graphes.

1.2. TRAVAUX ANTÉRIEURS

Les travaux visant à prédire les fonctions cellulaires des protéines à partir du graphe d'interactions sont assez peu nombreux et tous récents puisque les données d'interactions ne sont disponibles que depuis quelques années. Nous résumons dans cette partie certains d'entre eux.

La première méthode, proposée par B. Schwikowski *et al.* [2000], consiste simplement à déduire les fonctions inconnues d'une protéine des fonctions de ses protéines voisines (si elles sont connues) : on choisit d'assigner à une protéine les fonctions les plus représentées dans son voisinage. Ce procédé est dénommé *règle de la majorité* (ou *majority rule*). Cette méthode est strictement locale, puisqu'elle se limite au voisinage immédiat des protéines et ne prend pas en compte le graphe dans son ensemble. De plus, cette règle n'inclut pas les connections entre protéines inconnues. Dans leur article, B. Schwikowski *et al.* soulignent que, dans le graphe d'interactions entre protéines chez la levure, seules 364 protéines de fonctions inconnues sur 554 possèdent un voisin de fonction connue, et seulement 69 possèdent plus de deux voisins de fonction connue, ce qui limite considérablement le nombre de prédictions faites par ce procédé et leur valeur.

C. Brun *et al.* [2002] décrivent une nouvelle méthode de classification fonctionnelle des protéines consistant en quatre étapes. Tout d'abord, on commence par sélectionner un certain nombre de protéines qui seront classées : on choisit les protéines de degré supérieur ou égal à trois. Ensuite on définit une relation d'équivalence entre les protéines à classer : deux protéines sont en relation si elles interagissent directement l'une sur l'autre ou si elles ont au moins un interacteur commun ; on considère alors le graphe de cette relation et on en sélectionne les composantes connexes. Puis on calcule une distance entre toute paire $\{P_1, P_2\}$ de protéines à classer. Deux distances sont proposées ; elles sont définies par :

$$D1(P_1, P_2) = \frac{|Int(P_1)\Delta Int(P_2)|}{|Int(P_1) \cup Int(P_2)|}$$

et

$$D2(P_1, P_2) = \frac{|Int(P_1)\Delta Int(P_2)|}{|Int(P_1) \cup Int(P_2)| + |Int(P_1) \cap Int(P_2)|}$$

où $Int(P)$ est l'ensemble des interacteurs de la protéine P augmenté de P elle-même et où Δ représente la différence symétrique. Cette seconde distance est appelée distance de Czekanovski-Dice. On construit enfin, à partir de cette distance, un arbre de classification.

M. P. Samanta et S. Liang [2003] proposent une méthode statistique pour annoter des protéines de fonction inconnue reposant sur le principe selon lequel si deux protéines partagent un nombre d'interacteurs significativement grand, alors elles ont des fonctions proches. On calcule la probabilité pour que deux protéines

ayant chacune n_1 et n_2 interacteurs (c'est-à-dire étant de degré n_1 et n_2 dans le graphe d'interactions) aient m interacteurs communs. Cette probabilité est calculée pour toute paire de sommets du graphe, puis on choisit la paire correspondant à la probabilité la plus élevée, et on fusionne ces deux sommets qui appartiendront à une même classe. La probabilité de ce nouveau groupe est la moyenne géométrique des probabilités individuelles. On répète ce processus, créant ainsi de plus en plus de classes ou étendant les classes existantes, jusqu'à ce qu'un seuil maximal pour la probabilité soit atteint. Cette méthode a été appliquée au graphe d'interactions chez la levure et a permis de prédire de façon hautement probable les fonctions de 81 protéines.

Toujours en 2003, G. Bader et C. Hogue [2003] décrivent une méthode de classification pour l'annotation des protéines à partir du graphe d'interactions. Cette méthode se déroule en trois étapes. On rappelle qu'un *k-core* est un sous-graphe de degré minimal k. Tout d'abord, on pondère les sommets du graphe par le degré le plus grand d'un *k-core* dans le voisinage du sommet considéré, qu'on multiplie par $k_{\max}$, le degré maximum dans le voisinage immédiat du sommet. La deuxième étape consiste à choisir itérativement parmi tous les sommets celui ayant le poids le plus élevé pour créer une nouvelle classe, puis lui ajouter ses sommets adjacents tant que leur poids est supérieur à un certain seuil, les sommets ne pouvant pas être choisis plus d'une fois. Lors de la troisième étape, les classes sont filtrées (les classes ne contenant pas au moins un 2-*core* sont éliminées), et on étend éventuellement les classes restantes de manière empiétante. On attribue enfin aux complexes obtenus un score correspondant au produit de la densité du sous-graphe induit par le complexe et du cardinal de celui-ci.

Colombo *et al.* [2003] développent une méthode composée de trois étapes : tout d'abord on forme les noyaux des classes en sélectionnant les sommets maximisant une densité locale en arête, puis on étend ces noyaux par leurs éléments adjacents si ceux-ci y sont suffisamment connectés. Enfin on ordonne les sommets restants dans l'ordre décroissant de la densité locale, et on les place itérativement dans la classe la plus proche (correspondant au nombre de connections avec le sommet le plus grand). Dans cet article, cette méthode est appliquée à la détection de gènes orthologues.

Dans l'article [Vasquez *et al.*, 2003], le problème d'annotation est envisagé comme problème d'optimisation globale. Il s'agit d'affecter une fonction aux protéines non caractérisées de façon à minimiser le nombre de paires de protéines interagissant et ne possédant pas la même fonction. Ce problème est résolu par l'utilisation de la méthode du recuit simulé.

Enfin, Y. Chen et D. Xu [2005] proposent une méthode d'annotation prenant en compte non seulement le graphe d'interactions entre protéines mais aussi d'autres données disponibles concernant les complexes de protéines et les profils *gene-express* de données de puces (calcul d'un coefficient de corrélation entre chaque paire de gènes). Les auteurs proposent d'utiliser une machine de Boltzmann ainsi que la méthode de recuit simulé pour analyser ces données et en déduire, pour chaque protéine, une liste de fonctions possibles avec un score pour chaque fonction.

1.3. OBJECTIFS ET ENJEUX DE LA MÉTHODE PROPOSÉE

On considère un graphe simple non pondéré $G = (X, E)$, représentant un réseau d'interactions entre protéines. On note n le nombre de sommets de G et m son nombre d'arêtes : $n = |X|$, $m = |E|$. Le but de la méthode est de former des classes (sous-ensembles de X) correspondant à des sous-graphes induits denses en arêtes, tout en étant pertinentes d'un point de vue biologique (cardinal des classes, nombre de classes, ...), les classes devant être assimilées aux fonctions cellulaires des protéines.

La méthode de classification (cf. par exemple [Arabie *et al.*, 1999 ; Brucker, 1978 ; Brucker, Barthélemy, 2007 ; Hansen, Jaumard, 1997 ; Johnson, 1967] pour une présentation de ce domaine) proposée est une méthode de classification empiétante et non complète (cf. [Charon *et al.*, 2007(b) ; Denœud, 2006, 2009 ; Denœud *et al.*, 2005]). En effet, le but étant avant tout de mettre en évidence des « classes naturelles », les contraintes liées à un partitionnement seraient trop restrictives. De plus, il est très souhaitable de permettre le chevauchement des classes puisque les protéines peuvent intervenir dans plusieurs fonctions cellulaires.

La première difficulté relative à notre objectif réside dans la nature de la classification (empiétante et pas forcément complète), qui rend impossible l'application de méthodes classiques de partitionnement de graphes (cf. [Alpert, Kahng, 1995 ; Elsner, 1997]). La deuxième difficulté réside en la détermination du nombre de classes, celui-ci ne devant pas être fixé arbitrairement mais devant dépendre du graphe étudié. Cette détermination est délicate mais évidemment capitale pour le résultat de la méthode.

Enfin l'objectif d'annotation des protéines ajoute des contraintes difficiles à modéliser et à prendre en compte. Il est difficile de mettre au point une fonction objectif adaptée au problème : le nombre de classes doit être pertinent, les classes doivent être denses en arêtes, ni trop petites ni trop grosses, l'empiétement doit être relativement limité pour permettre l'interprétation des classes... De plus, même en élaborant une fonction objectif adaptée, les deux premières difficultés envisagées ici (détermination du nombre de classes et empiètement) rendent l'espace de solution de notre problème de taille gigantesque et donc la résolution du problème plus ardue.

Ne disposant pas d'une fonction objectif bien définie, on se pose alors le problème de la validation de la méthode : comment évaluer la « qualité » des classes obtenues ? Comment comparer entre elles deux solutions ? Pour cela, nous utiliserons les différents critères évoqués plus haut. On appliquera la méthode à des graphes aléatoires puis à des graphes réels d'interactions entre protéines.

2. DESCRIPTION DE LA MÉTHODE

La méthode proposée est composée de trois étapes. On crée initialement les noyaux des classes par le biais d'une fonction de densité locale définie sur les sommets du graphe. Ensuite, on modifie ces noyaux (et éventuellement leur nombre) en utilisant une adaptation de la méthode des nuées dynamiques [Diday, 1971]. Ces deux premières étapes fournissent des classes non chevauchantes. On étend finalement les

noyaux de manière empiétante dans une troisième étape, en se fondant sur des critères caractérisant la qualité des classes obtenues. Les paragraphes suivants donnent le détail de chacune de ces étapes.

2.1. ÉTAPE 1 : CRÉATION DES NOYAUX INITIAUX DES CLASSES

On considère une fonction de densité locale De évaluant la densité en arêtes au voisinage d'un sommet quelconque (cf. la partie 2.4. pour l'expression de De). On cherche tous les sommets s réalisant des maxima locaux de cette fonction De et de densité supérieure à la moyenne, c'est-à-dire vérifiant :

$$\forall s' \in \Gamma(s), De(s) \geq De(s') \text{ et } De(s) \geq \overline{De}$$

où $\Gamma(s)$ désigne l'ensemble des sommets adjacents à s et $\overline{De}$ la moyenne des densités sur tous les sommets du graphe : $\overline{De} = \frac{1}{|X|} \sum_{s \in X} De(s)$. Ces maxima locaux formeront les noyaux initiaux des classes ; si plusieurs de ces sommets sont adjacents, on les place dans le même noyau.

Ensuite on ajoute à chaque noyau tout sommet s qui lui est adjacent et dont la densité $De(s)$ est supérieure ou égale à $\overline{De}$. Si un tel sommet est adjacent à plusieurs noyaux, on ne le classe pas lors de cette étape : les noyaux créés sont disjoints. L'algorithme est décrit dans le Tableau 1.

pour tout sommet $s \in X$ *:*
- *calculer sa densité locale* $De(s)$

calculer la densité moyenne $\overline{De}$

pour tout sommet $s \in X$ *:*
- *marquer* s *si :* $\forall s' \in \Gamma(s)$, $De(s) \geq De(s')$ *et* $De(s) \geq \overline{De}$

considérer le sous-graphe G_{opt} *induit par les sommets marqués*
initialiser les noyaux par les composantes connexes de G_{opt}*, lesquelles sont déterminées par un parcours de graphe* (cf. [Cormen *et al.*, 1994])

pour tout sommet s *de* X *non classé avec* $De(s) \geq \overline{De}$ *:*
- *si* s *possède au moins un voisin classé et si tous ses voisins classés sont dans un même noyau, classer* s *dans ce noyau.*

TABLEAU 1. Algorithme d'initialisation

Cette étape correspond à la première étape proposée dans [Colombo *et al.*, 2003]. Avec une structure de données appropriée (listes d'adjacence), elle est de complexité $O(m + n \times c_{De})$, où c_{De} désigne la complexité du calcul de la densité en un sommet du graphe (celle-ci est en $O(1)$ pour certaines densités étudiées plus loin).

2.2. ÉTAPE 2 : AMÉLIORATION DES NOYAUX

Une fois les noyaux initiaux déterminés, on cherche à améliorer ces noyaux en utilisant une méthode de nuées dynamiques [Diday, 1971]. Elle consiste à alterner une étape d'affectation (création d'une partition obtenue en affectant chaque sommet au centre le plus proche) avec une étape de recentrage (calcul des centres des classes) et à

répéter ce processus jusqu'à ce que la partition obtenue soit stable. Nous avons modifié légèrement cette méthode afin de permettre la modification du nombre de classes au cours du déroulement de l'algorithme. Nous avons aussi relâché la contrainte relative à l'obtention d'un partitionnement de X, certains sommets pouvant n'être affectés à aucune classe en cas d'ambiguïté.

On considère une distance d définie sur les sommets du graphe (cf. la partie 2.5. pour l'expression de cette distance). De façon classique, on définit les centres des classes comme les centres de gravité des classes par rapport à cette distance : le sommet c est centre de la classe C s'il vérifie $\sum_{s' \in C} d(c, s') = \min_{s \in C}(\sum_{s' \in C} d(s, s'))$. Plusieurs sommets, pas forcément adjacents, peuvent être centres d'une même classe.

Si on considère maintenant que les classes sont réduites à leurs centres, l'affectation des sommets se fait en calculant, pour chaque sommet s et pour chaque classe C, la moyenne des distances entre s et les sommets de C : $\frac{1}{|C|} \sum_{c \in C} d(s, c)$; on sélectionnera ensuite la classe qui réalise le minimum de cette valeur comme nouvelle classe de s. Si plusieurs classes vérifient ce minimum, on ne classe pas le sommet s afin d'éviter de faire un choix arbitraire (il sera classé ultérieurement).

La modification du nombre de classes se fait au moyen de trois processus, qui sont lancés après stabilisation de la solution dans la méthode des nuées :

- Si deux classes ont une distance moyenne entre leurs centres inférieure ou égale à la distance moyenne intra-classes (c'est-à-dire la distance moyenne entre toute paire de sommets appartenant à une même classe), ces deux classes sont fusionnées. Autrement dit, si deux classes différentes ont leurs centres respectifs plus proches que deux sommets d'une même classe en moyenne, alors les deux classes sont trop proches pour qu'on les considère comme distinctes, et donc on les fusionne.
- Si un sommet est à distance maximum de tout centre (il n'a donc pas été classé lors de l'étape d'affectation), ce sommet forme un nouveau centre. En effet, si un tel sommet se trouve dans une région du graphe non représentée dans les classes, il semble intéressant de former une classe à cet endroit afin de classer un maximum de sommets et d'éviter que des zones du graphe ne soient pas étudiées. On s'assure tout de même que les nouveaux centres sont à distance maximum les uns des autres (en particulier deux sommets voisins ne doivent pas constituer deux nouveaux centres).
- On vérifie la connexité des classes obtenues ; si une classe n'est pas connexe, chacune de ses composantes connexes devient une classe. Cette étape nous assure la connexité des classes, ce qui semble une contrainte minimale puisqu'on souhaite obtenir des classes denses en arêtes.

L'étape 2 de la méthode consiste alors à répéter l'algorithme des nuées dynamiques, suivi de ces trois processus, et ce jusqu'à ce qu'on observe une stabilisation des classes. N'ayant pas de résultat théorique sur la convergence de cet algorithme, on ajoute de plus un nombre d'itérations maximum N pour arrêter l'exécution du programme au bout d'un certain temps si on n'a pas observé de stabilisation. L'algorithme correspondant est donné dans le Tableau 2.

Chaque étape de l'algorithme des nuées dynamiques se fait en $O(n^3)$, l'étape de modification du nombre de classes aussi. Il est toutefois difficile d'évaluer la

complexité globale de cet algorithme puisqu'on ne connaît pas le nombre d'itérations des différentes étapes.

** Répéter :*

Algorithme des nuées dynamiques :

– *Répéter :*
  - *Recentrage : pour chaque classe C, calculer les centres (calculer $\sum_{s' \in C} d(s, s')$ et sélectionner les sommets s qui minimisent cette somme).*
  - *Affectation :*
    - ◇ *Pour chaque sommet s non centre, calculer la distance entre s et chaque classe (moyenne des distances entre le sommet et les centres de la classe).*
    - ◇ *Si une seule classe vérifie le minimum de cette distance : s est affecté à cette classe, sinon s n'est pas classé.*
    - ◇ *Si la distance minimum est maximum, marquer s.*

– *jusqu'à ce que les classes ne soient plus modifiées.*

Modification du nombre de classes :

– *Calculer la distance moyenne intra-classes $\bar{d}_{int}$ (moyenne des distances entre toute paire de sommets d'une même classe).*
– *Pour chaque paire de classes $\{C, C'\}$:*
  - *calculer la distance moyenne entre les centres $\bar{d}_c$ (moyenne des distances entre tous les centres de C et tous les centres de C').*
  - *Si $\bar{d}_c < \bar{d}_{int}$, les sommets de C' sont ajoutés à C, la classe C' est vidée.*
– *Soit Ω l'ensemble des sommets marqués. Répéter :*
  - *choisir un élément de Ω au hasard et créer un nouveau centre constitué de ce sommet.*
  - *mettre à jour Ω par les sommets à distance maximum de tout centre.*
– *tant que Ω est non vide.*
– *On vérifie la connexité des classes : si une classe n'est pas connexe, chacune de ses composantes connexes forme une nouvelle classe.*

** jusqu'à ce que les classes ne soient plus modifiées ou jusqu'à N itérations.*

TABLEAU 2. Algorithme d'amélioration

2.3. ÉTAPE 3 : EXTENSION EMPIÉTANTE DES NOYAUX

On considère une fonction évaluant la « qualité » d'un sous-ensemble de X (densité en arêtes, cardinal (cf. partie 2.6.)). Le principe de l'extension est d'ajouter itérativement à chaque classe les sommets qui y sont le plus connectés s'ils permettent d'augmenter la qualité de la classe (qu'ils soient déjà classés ou non). L'algorithme correspondant est donné dans le Tableau 3.

Les ensembles de candidats traités ne sont pas nécessairement disjoints puisqu'un sommet peut être adjacent à plusieurs classes, les classes obtenues peuvent donc être empiétantes. L'ordre dans lequel les classes sont examinées n'intervient pas puisque le chevauchement des classes est autorisé.

Pour chaque classe :
- *calculer la valeur de la fonction de qualité*
- *répéter :*
  - *pour chaque sommet adjacent à la classe courante, calculer le nombre d'arêtes le reliant à cette classe*
  - *sélectionner tous les sommets réalisant le maximum de ce nombre ; ils forment l'ensemble des candidats*
  - *calculer la qualité de la classe formée par la classe courante et l'ensemble des sommets candidats*
  - *si la valeur calculée est supérieure ou égale à celle de la classe courante :*
    - ◇ *placer les candidats dans la classe*
    - ◇ *mettre à jour la qualité de la classe*
- *tant que des sommets ont été ajoutés à cette itération.*

S'il existe des classes identiques, n'en garder qu'un exemplaire.

TABLEAU 3. Algorithme d'extension

2.4. FONCTIONS DE DENSITÉ LOCALE

Pour la construction des noyaux des classes, on utilise une fonction de densité locale De évaluant la densité en arêtes au voisinage d'un sommet s du graphe. On propose d'étudier les cinq fonctions de densité De_1, De_2, De_3, De_4, De_5 suivantes :

Le degré relatif de s : c'est le degré $deg(s)$ de s divisé par le plus grand degré Δ dans le graphe :

$$De_1(s) = \frac{deg(s)}{\Delta}.$$

Le degré moyen dans le voisinage de s : il s'agit de la somme des degrés du sommet s et de ses sommets adjacents divisée par le nombre de sommets considérés :

$$De_2(s) = \frac{deg(s) + \sum_{s' \in \Gamma(s)} deg(s')}{1 + deg(s)}.$$

Le taux de triangles dans le voisinage de s : on considère le nombre $N_t(s)$ de triangles ayant s pour sommet (c'est-à-dire le nombre d'arêtes reliant entre eux les voisins de s) et on le divise par le nombre maximum de triangles réalisables autour d'un sommet du degré de s :
• si $deg(s) > 1$,

$$De_3(s) = \frac{N_t(s)}{\frac{1}{2}\ deg(s)\ (deg(s) - 1)},$$

• sinon,

$$De_3(s) = 0.$$

Le pourcentage d'arêtes dans le voisinage de s : on divise le nombre d'arêtes autour de s (d'extrémité s ou reliant deux sommets adjacents à s) et on le divise par le nombre maximum d'arêtes sur l'ensemble formé par s et ses voisins :

$$De_4(s) = \frac{deg(s) + N_t(s)}{\frac{1}{2}\ deg(s)\ (deg(s)+1)}.$$

Cette fonction permet de prendre en compte non seulement la densité d'arêtes dans le voisinage de s, mais aussi son degré. C'est cette fonction qui est préconisée dans [Colombo *et al.*, 2003] , où on impose de plus qu'un sommet de degré 1 prenne 0 comme valeur de densité locale afin de ne pas former de noyaux à partir de tels sommets, qui correspondent, suivant la formule donnée, à une valeur maximale de De_4.

Le taux de triangles pondéré par le degré relatif : cette fonction est simplement le produit des fonctions De_1 et De_3 :
• si $deg(s) > 1$,

$$De_5(s) = \frac{N_t(s)}{\frac{1}{2}\ \Delta\ (deg(s)-1)}$$

• sinon,

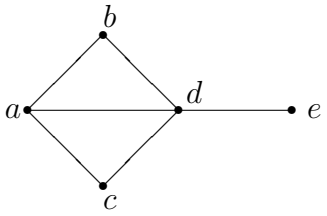
$$De_5(s) = 0.$$

Elle permet elle aussi de prendre en compte à la fois la densité en arêtes dans le voisinage et le degré du sommet. Elle donne toutefois un peu plus de poids au degré du sommet que De_4.

Remarque 1. Les fonctions De_1, De_3, De_4 et De_5 sont à valeurs dans $[0, 1]$.

Afin de mieux comprendre le comportement de ces différentes fonctions, étudions l'exemple suivant :

EXEMPLE 1. On considère le graphe à cinq sommets suivant :



Calculons les différentes valeurs de densité en chaque sommet de ce graphe :

s	a	b et c	d	e
$deg(s)$	3	2	4	1
$N_t(s)$	2	1	2	0
$De_1(s)$	0,75	0,5	**1**	0,25
$De_2(s)$	2,75	**3**	2,4	**2,5**
$De_3(s)$	0,66	**1**	0,33	0
$De_4(s)$	0,83	**1**	0,6	**1**
$De_5(s)$	**0,5**	**0,5**	0,33	0

En gras sont indiquées les valeurs correspondant à des optima locaux de chaque fonction de densité ; il s'agit des sommets dont la densité est supérieure ou égale à la densité de tous leurs voisins. D'après $De1$, le seul sommet optimum local est le sommet d, qui est tout simplement celui de degré maximum. Pour les autres fonctions, b et c sont des sommets optima locaux. Pour De_2 et De_4, le sommet e est lui aussi un optimum local car ce sommet est de degré 1 et donc de densité 1. D'après De_5, le sommet a fait lui aussi partie des optima.

On peut voir sur ce petit exemple que l'utilisation de l'une ou l'autre de ces fonctions peut modifier du tout au tout les noyaux initiaux des classes.

2.5. DISTANCES DANS LE GRAPHE

Nous avons besoin, pour l'étape d'amélioration des noyaux, de définir une distance entre sommets du graphe. On propose d'utiliser les deux indices de distance suivants : l'indice du nombre de plus courtes chaînes et la distance de Dice (cf. par exemple [Denœud *et al.*, 2005] pour une étude comparative de plusieurs indices de distance).

2.5.1. Indice du nombre de plus courtes chaînes

Cet indice est fondé sur la distance la plus classique dans un graphe : la longueur de la plus courte chaîne. Dans un graphe $G = (X, E)$ non pondéré et non orienté, la distance entre deux sommets distincts s et s' correspond au nombre d'arêtes dans une plus courte chaîne entre s et s'.

Nous allons plus précisément utiliser une généralisation des chaînes. Pour un entier k quelconque, définissons une *pseudo-chaîne de longueur k* comme une suite de k arêtes a_1, a_2, ..., a_k telle que, pour tout indice i compris entre 2 et $k-1$, a_i ait une extrémité commune avec a_{i-1} et l'autre extrémité avec a_{i+1} (cela n'exclut pas que deux arêtes consécutives soient identiques, contrairement à une chaîne dans le sens classique de la théorie des graphes). Le nombre de pseudo-chaînes de longueur donnée peut facilement être calculé à l'aide des puissances de la matrice d'adjacence du graphe à traiter. En effet, soit $M = (M_{ss'})_{(s,s')\in X^2}$ cette matrice d'adjacence et soit $M^k = (M^k_{ss'})_{(s,s')\in X^2}$ sa k^e puissance. Le nombre $ch_k(s, s')$ de pseudo-chaînes de longueur k entre deux sommets s et s' est alors donné par $M^k_{ss'}$.

Pour définir l'indice du nombre de plus courtes chaînes entre deux sommets s et s', nous allons nous intéresser au nombre de pseudo-chaînes de faible longueur entre s et s'. En effet, il est souhaitable que deux sommets reliés par exemple par une seule pseudo-chaîne de longueur 2 soient moins proches que deux sommets reliés par

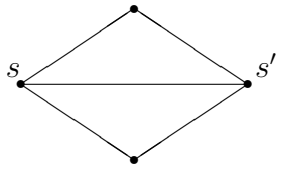
deux telles pseudo-chaînes. D'autre part, il n'est pas gênant de ne pas tenir compte des pseudo-chaînes de longueur élevée qui n'apportent que peu d'informations.

Puisque la méthode est destinée à être appliquée à des graphes d'interactions entre protéines, dont le diamètre n'excède pas 7, on considérera qu'une pseudo-chaîne est de longueur « faible » si elle possède au plus trois arêtes. On définit alors, pour tout s et tout s' appartenant à X, l'indice $Pcc(s, s')$ du nombre de plus courtes chaînes entre s et s' par :

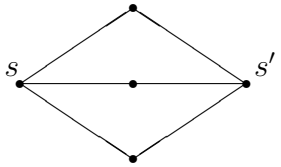
$$Pcc(s, s') = ch_1(s, s') + \frac{ch_2(s, s')}{2} + \frac{ch_3(s, s')}{4}.$$

De cette manière, on prend en compte toutes les pseudo-chaînes de longueur inférieure ou égale à trois, tout en pondérant ces valeurs par une puissance de deux suivant la longueur considérée. En effet une chaîne de longueur 1 doit compter davantage qu'une pseudo-chaîne de longueur 2, qui elle-même doit avoir plus de poids qu'une pseudo-chaîne de longueur 3. La pondération par une puissance de 2 contribue à donner davantage de poids aux chaînes de longueur 1, qui seront aussi comptabilisées comme pseudo-chaînes de longueur 3, et aussi à prendre en compte le degré des sommets considérés (s'il existe une arête entre s et s', et que s est de degré d, on comptera alors au moins d pseudo-chaînes distinctes de longueur 3 entre s et s').

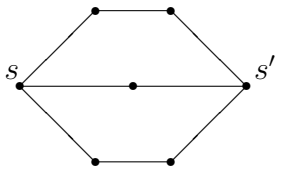
EXEMPLE 2. Illustrons cet indice par les cas de figure suivants :



$$Pcc(s, s') = 1 + \tfrac{2}{2} + \tfrac{5}{4} = 3,25$$



$$Pcc(s, s') = 0 + \tfrac{3}{2} + \tfrac{0}{4} = 1,5$$



$$Pcc(s, s') = 0 + \tfrac{1}{2} + \tfrac{2}{4} = 1$$

2.5.2. Distance de Dice

Soient s et s' deux sommets. On note S (respectivement S') l'ensemble constitué des sommets voisins de s (respectivement de s') augmenté de s (respectivement de s') lui-même. La distance de Dice $Dice(s, s')$ [Dice, 1945] entre deux sommets s et s' de G est donnée par :

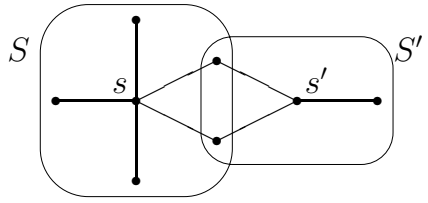
$$Dice(s, s') = \frac{|S \Delta S'|}{|S| + |S'|}$$

où $S \Delta S'$ désigne la différence symétrique entre S et S' : $S \Delta S' = (S \cup S') \setminus (S \cap S')$. Cette distance correspond donc au cardinal de la différence symétrique entre S et S' rapporté à la somme des cardinaux de ces deux ensembles. Cette distance vaut 0 lorsque $S = S'$ et 1 lorsque $S \cap S' = \emptyset$.

Remarque 2. Cette mesure est bien une distance si on considère les ensembles S et S'. Ça n'en est plus tout à fait une lorsqu'elle est définie comme ci-dessus entre deux sommets d'un graphe : $Dice(s, s') = 0$ n'implique pas l'égalité $s = s'$ mais simplement $S = S'$. On l'appellera tout de même *distance de Dice*.

EXEMPLE 3. On considère la configuration dessinée ci-dessous. Pour ce graphe, on a $|S| = 6$, $|S'| = 4$ et $|S \Delta S'| = 6$. Ainsi :

$$Dice(s, s') = \frac{6}{6 + 4} = \frac{3}{5}.$$



2.6. CRITÈRES D'EXTENSION

Il s'agit de définir une notion de qualité d'un sous-ensemble de sommets. On autorisera alors l'ajout d'un sommet dans une classe si cela augmente cette qualité. La fonction de qualité doit à la fois dépendre :

- du pourcentage d'arêtes du sous-graphe, le but étant de mettre en évidence des zones denses en arêtes ;
- du cardinal du sous-ensemble considéré. La fonction ne doit pas être nécessairement maximale pour des cliques ; à densité égale, une classe d'ordre élevé doit avoir une qualité plus grande qu'une classe d'ordre inférieur.

Nous proposons dans cette section deux critères vérifiant ces propriétés.

On notera H un sous-graphe de G, p son ordre (nombre de sommets) et q sa taille (nombre d'arêtes). On notera S l'ensemble des sommets candidats à entrer dans la classe H ; ce sont les sommets les plus connectés à la classe, c'est-à-dire reliés chacun par c arêtes aux sommets de la classe, c étant le nombre maximum d'arêtes entre un sommet extérieur à la classe et les sommets de la classe.

2.6.1. Critère du degré moyen

Étant données les remarques précédentes, il semble immédiat de considérer le degré moyen d_m comme fonction de qualité :

$$d_m(H) = \frac{\sum_{s \in H} d(s)}{p} = \frac{2.q}{p}.$$

En effet, plus un sous-graphe est dense en arêtes, plus son degré moyen est élevé et un petit sous-graphe aura en moyenne un degré moyen moins élevé qu'un gros sous-graphe, même si c'est une clique ($d_m(H) \leq p - 1$).

Si on considère une classe H qu'on cherche à étendre et S l'ensemble des candidats, la règle d'extension est alors :

$$H \text{ devient } H \cup S \text{ si et seulement si } d_m(H \cup S) \geq d_m(H).$$

Lorsque l'ensemble S est réduit à un singleton, cette extension revient à ajouter le candidat à la classe si $c \geq \frac{q}{p}$ (on rappelle que c est le nombre d'arêtes entre chaque sommet de S et les sommets de la classe), ce qui s'applique en particulier aux cliques qu'on pourra donc étendre suivant ce critère.

On peut, plus généralement, s'intéresser aux règles d'extension suivantes :

$$H \text{ devient } H \cup S \text{ si et seulement si } c \geq \frac{\alpha.q}{p}$$

où α est un nombre réel strictement positif. Plus le coefficient α est élevé et plus le critère d'extension est strict : si α est petit ($\frac{\alpha.q}{p} < 1$), la classe sera étendue par tous ses sommets adjacents ; s'il est grand ($\frac{\alpha.q}{p} > \Delta$, où Δ représente le degré maximum du graphe), aucune extension n'aura lieu.

On obtient alors une famille de règles d'extension (critère du degré moyen généralisé) plus ou moins strictes suivant la valeur de α. On pourra déterminer cette dernière en fonction du graphe traité et des exigences sur les classes (densité, cardinal).

2.6.2. Critère probabiliste

On constate que les zones denses en arêtes sont rares (surtout dans un graphe ayant une densité en arêtes relativement faible, ce qui est le cas dans l'application envisagée plus loin). On est donc amené à chercher des zones rares, c'est-à-dire de faible probabilité, car on espère qu'elles reflètent le mécanisme biologique sous-jacent. On propose alors comme critère d'extension d'étendre une classe si la probabilité d'existence de la classe étendue est plus faible que celle de la classe de départ.

Pour chaque classe H (sous-graphe de G) ayant p sommets, considérons la probabilité $P(H)$ que H ait q arêtes sachant que le graphe initial G possède m arêtes pour n sommets. On étendra H si les sommets ajoutés diminuent la valeur de cette probabilité.

Soit $M = \frac{n(n-1)}{2}$ le nombre maximum de paires qu'on peut former avec les sommets de G et $Q = \frac{p(p-1)}{2}$ le nombre maximum de paires qu'on peut former avec ceux de H.

PROPOSITION 1. *La probabilité $P(H)$ vaut (loi hypergéométrique) :*

$$P(H) = \frac{C_m^q C_{M-m}^{Q-q}}{C_M^Q}.$$

DÉMONSTRATION 1. Sur les M paires possibles, il y en a m qui représentent une arête du graphe (sommets adjacents). On cherche la probabilité que H contienne exactement q paires de sommets adjacents de G. Cette probabilité est égale à celle de tirer exactement q éléments possédant une certaine propriété (ici, de définir une arête) lors de Q tirages sans remise dans un ensemble de M éléments dont m possèdent cette propriété. D'où le résultat. □

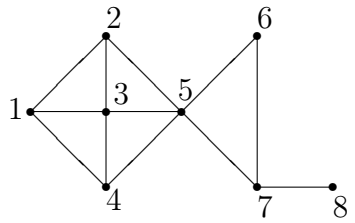
Si on considère une classe H qu'on cherche à étendre et l'ensemble S des candidats, la règle d'extension est :

$$H \text{ devient } H \cup S \text{ si et seulement si } P(H \cup S) \leq P(H).$$

2.6.3. Exemple

On illustre l'étape 3 de la méthode (extension des classes) par l'exemple suivant.

EXEMPLE 4. Soit le graphe G suivant.



On suppose avoir obtenu sur ce graphe les noyaux suivants :
– Noyau 1 : $\{1, 2, 3, 4\}$
– Noyau 2 : $\{6\}$

Nous allons utiliser le critère du degré moyen généralisé avec $\alpha = 2$. Commençons par étendre le noyau 1 :

- classe 1 : $\{1, 2, 3, 4\}$, $\frac{2q}{p} = \frac{10}{4} = 2,5$.

On détermine l'ensemble S des sommets candidats à entrer dans la classe ; ici le sommet 5 est le seul adjacent au noyau : $S = \{5\}$. Il est relié par 3 arêtes au noyau : $c = 3$. On vérifie si $c \geq \frac{2q}{p}$. Ici c'est le cas, on ajoute donc le sommet 5 à la classe :

- classe 1 : $\{1, 2, 3, 4, 5\}$, $\frac{2q}{p} = \frac{16}{5} = 3,2$.

On détermine à nouveau l'ensemble des candidats : $S = \{6, 7\}$, $c = 1$. L'inégalité $c \geq \frac{2q}{p}$ n'est pas vérifiée, la classe n'est donc pas étendue et on s'arrête.

Étendons maintenant le noyau 2 :

- classe 2 : $\{6\}$, $\frac{2q}{p} = \frac{0}{1} = 0$.

Les candidats sont $S = \{5, 7\}$, avec $c = 1$. Puisque $c \geq \frac{2q}{p}$, on étend la classe en y ajoutant ces sommets :

- classe 2 : $\{5, 6, 7\}$, $\frac{2q}{p} = \frac{6}{3} = 2$.

Les sommets adjacents à la classe sont 2, 3, 4 et 8. Ils sont tous candidats car tous reliés par une arête à la classe : $S = \{2, 3, 4, 8\}$, $c = 1$. Le critère d'extension n'est pas vérifié donc on n'étend pas la classe et on s'arrête.

L'algorithme a construit deux classes : $\{1, 2, 3, 4, 5\}$ et $\{5, 6, 7\}$, résultat qui semble intuitivement satisfaisant. Il y a un sommet non classé (8) et un sommet classé dans deux classes (5). L'utilisation du critère d'extension probabiliste donne pour ce graphe les mêmes classes.

3. VALIDATION DE LA MÉTHODE SUR DES GRAPHES ENGENDRÉS ALÉATOIREMENT

Dans cette partie, on teste la méthode proposée sur des graphes engendrés aléatoirement et qui contiennent des classes. Nous présentons tout d'abord la méthode de génération aléatoire de graphe que nous allons employer, puis on testera sur ces graphes les différentes étapes de la méthode de classification étudiée. On comparera ainsi les différentes fonctions de densité locale, les deux indices de distance proposés ainsi que les deux critères d'extension.

3.1. GÉNÉRATION ALÉATOIRE DE GRAPHES

Nous proposons d'utiliser le modèle de Watts et Strogatz [1998] pour engendrer de façon aléatoire des graphes ayant une structure relativement proche des graphes d'interactions entre protéines (graphes dits de *petit monde* ; cf. [Denœud, 2006] pour plus de détails). Il nous faut en outre créer des graphes dans lesquels il existe des classes de sommets denses en arêtes. Afin de simplifier l'interprétation des résultats, on ne constitue pas de classes empiétantes mais simplement un partitionnement des sommets. On propose donc, étant donnés quatre paramètres n (nombre de sommets), nbc (nombre de classes), p_i (probabilité d'arêtes initiales) et p_r (probabilité de recâblage d'une arête), la procédure décrite dans le Tableau 4. On remarquera que le graphe obtenu ressemble d'autant plus à un graphe représentant une partition (c'est-à-dire à un graphe dont les composantes connexes sont des cliques) que p_i est proche de 1 et p_r de 0.

Si on néglige les arêtes éventuellement ajoutées pour assurer la connexité du graphe, la valeur donnée à p_i détermine le nombre d'arêtes, et donc la densité, dans le graphe. Les graphes obtenus sont de densité très faible (tout comme les graphes d'interactions entre protéines) étant donné qu'on place au maximum toutes les arêtes des classes, ce qui correspond à un nombre restreint, surtout quand le nombre de classes est élevé, plus les quelques éventuelles arêtes ajoutées pour la connexité du graphe. La valeur p_r a à la fois un effet de diminution de la densité des classes et d'augmentation de la densité interclasse. Plus la valeur de p_r est grande et moins les classes sont apparentes dans le graphe.

– *On répartit de manière équilibrée les n sommets dans les nbc classes.* – *Entre chaque paire de sommets d'une même classe, on place une arête avec une probabilité p_i.* – *On recâble chaque arête $\{x, y\}$ avec une probabilité p_r (c'est-à-dire qu'on échange une de ses extrémités, par exemple y, avec n'importe quel sommet du graphe non déjà voisin de l'autre extrémité, x ; ce nouveau voisin de x est choisi avec un tirage aléatoire uniforme sur l'ensemble des sommets non voisins de x).* – *S'il existe des sommets de degré zéro, on ajoute une arête entre ce sommet et un sommet choisi au hasard dans sa classe.* – *On vérifie la connexité du graphe : tant que le graphe n'est pas connexe, on effectue un parcours en profondeur à partir d'un sommet choisi au hasard, puis on ajoute une arête aléatoirement entre un sommet de la composante connexe trouvée et un sommet appartenant au reste du graphe.*

TABLEAU 4. Algorithme de génération aléatoire de graphes

3.2. TEST DE L'ÉTAPE 1 : CRÉATION DES NOYAUX INITIAUX DES CLASSES

Commençons par tester l'étape 1 de création des noyaux initiaux des classes. On utilise la procédure de génération aléatoire de graphes pour $n = 1000$ avec, dans un premier temps, une probabilité d'arêtes initiales p_i égale à 1 et une probabilité de recâblage p_r égale à 0, puis avec $p_i = 0,55$ et $p_r = 0,35$. Le premier cas correspond à des cliques connectées par quelques arêtes (arêtes qu'il a fallu ajouter pour rendre le graphe connexe), le second correspond à des caractéristiques proches des graphes réels d'interactions entre protéines (cf. [Denœud, 2006]). Le nombre de classes dans le graphe varie de 10 (classes composées de 100 éléments) à 100 (classes de cardinal 10). Pour chaque jeu de paramètres, on engendre 100 graphes auxquels on applique l'étape 1 (cf. partie 2.1.) de la méthode de recherche de zones denses dans un graphe, pour chacune des fonctions de densité locale présentées.

Afin de comparer les différents résultats et d'évaluer la qualité des noyaux obtenus, on calcule deux critères :

- la distance de transfert normalisée (comprise entre 0 et 1) entre les noyaux et la partition initiale. La distance de transfert entre deux partitions est définie comme le nombre minimum d'éléments qu'il est nécessaire de transférer d'une classe dans une autre pour passer d'une partition à l'autre (cf. [Charon *et al.*, 2006, 2007(a) ; Denœud, 2006, 2008, 2009 ; Denœud, Guénoche, 2006]). Soit $Y \subseteq X$ l'ensemble des sommets classés dans les noyaux. Les noyaux forment une partition de l'ensemble Y. Soit P la partition initiale des sommets de X. On considère alors la partition P réduite à l'ensemble Y, et on calcule la distance de transfert normalisée entre ces deux partitions.
- le pourcentage de paires conservées. Il s'agit du pourcentage de paires de sommets classés ensemble dans la partition initiale qui sont également classés ensemble dans les noyaux, en ne prenant en compte que les sommets de Y.

Les tableaux suivants (Tableau 5) donnent les résultats obtenus ; dans chaque case la valeur en haut à gauche correspond à la moyenne du nombre de noyaux obtenus, celle en haut à droite au pourcentage moyen d'éléments classés dans ces

noyaux, celle en bas à gauche à la distance de transfert normalisée moyenne (définie précédemment) et celle en bas à droite au pourcentage moyen de paires conservées.

$p_i = 1$ $p_r = 0$	De_1	De_2	De_3	De_4	De_5
$nbc = 10$	1 | 1,8 0,88 | 5,8	3 | 70,1 0,57 | 36,3	10 | 98,2 0 | 100	10 | 98,2 0 | 100	10 | 98,2 0 | 100
$nbc = 50$	3 | 9,5 0,93 | 1	15 | 26 0,29 | 61	50 | 90,3 0 | 100	50 | 90,3 0 | 100	50 | 90,3 0 | 100
$nbc = 100$	30 | 15,6 0,68 | 6,3	31 | 28 0,24 | 58	100 | 81,6 0 | 100	100 | 81,6 0 | 100	100 | 81,6 0 | 100
$p_i = 0,55$ $p_r = 0,35$	De_1	De_2	De_3	De_4	De_5
$nbc = 10$	13,3 | 22,1 0,4 | 55,1	16,1 | 19,5 0,54 | 60,8	17,3 | 22,8 0,35 | 83,4	18,3 | 22,3 0,38 | 82,7	12,6 | 23,8 0,24 | 81,6
$nbc = 50$	56,4 | 25,7 0,33 | 52,7	80 | 23,5 0,44 | 63	89,7 | 28,5 0,29 | 94,3	124,2 | 28,7 0,44 | 90,1	55,5 | 31,5 0,09 | 94,5
$nbc = 100$	99,8 | 32,1 0,29 | 33,5	126,5 | 26,9 0,37 | 40,5	141,8 | 32,2 0,23 | 61,7	250,5 | 40,1 0,47 | 54,7	105,8 | 35 0,09 | 62,7

TABLEAU 5. Étude comparative des noyaux initiaux pour les différentes fonctions de densité locale

Les premières lignes, qui correspondent à p_i=1 et $p_r = 0$, permettent déjà d'évaluer les différentes fonctions de densité locale proposées ; les classes initiales des graphes testés sont des cliques, et elles sont reliées entre elles par un nombre presque minimum d'arêtes, on aimerait donc que dans ce cas trivial la méthode retrouve les classes dès cette première étape. C'est le cas pour les fonctions De_3, De_4 et De_5, qui produisent comme noyaux initiaux exactement les classes initiales. En revanche les fonctions De_1 et De_2 se comportent mal ; elles ne retrouvent qu'un petit nombre de classes par rapport au nombre de classes initiales.

Les dernières lignes correspondent aux paramètres $p_i = 0,55$ et $p_r = 0,35$. Dans ce cas la méthode ne parvient pas à retrouver les classes de façon exacte. Commençons par étudier le nombre de noyaux donnés par les différentes fonctions. On remarque que toutes les fonctions ont tendance à créer plus de noyaux que de classes initiales dans tous les cas sauf pour $nbc = 100$ avec De_1. Les fonctions de densité De_1 et De_5 donnent un nombre de noyaux le plus proche de nbc, puis viennent De_2, De_3 et en dernier De_4, laquelle donne à chaque fois beaucoup plus de noyaux que de classes initiales dans le graphe. Le pourcentage d'éléments classés dans les noyaux est à peu près stable quels que soient la fonction ou le nombre initial de classes (en général entre 20 % et 30 %). On peut tout de même noter que le nombre d'éléments classés avec De_4 et De_5 est plus élevé qu'avec les autres fonctions, surtout lorsque nbc est élevé. Cette différence correspond à la différence entre les nombres de noyaux créés. La distance de transfert et le pourcentage de paires conservées évaluent la distance entre les noyaux et les classes initiales. Ces deux indices permettent de conclure que les fonctions De_3, De_4 et De_5 semblent produire des noyaux plus proches des classes initiales et, parmi celles-ci, De_5 est la meilleure. En effet, les valeurs de la distance de transfert pour cette fonction sont

très faibles ; les noyaux construits sont donc très proches au sens des transferts de la partition initiale.

Au vu de ces observations, nous retiendrons donc pour la suite la fonction De_5 qui forme les noyaux les plus proches des classes initiales et en un nombre très proche de nbc.

3.3. Test de l'étape 2 : amélioration des noyaux des classes

On étudie maintenant le comportement de l'étape 2 proposée dans la partie 2.2. On veut notamment comparer les deux indices de distance proposés pour la méthode des nuées dynamiques. Pour cela, nous allons appliquer la méthode sur des graphes engendrés aléatoirement suivant la procédure décrite Tableau 4 pour $n = 100$. Il est impossible de considérer ici des graphes à 1000 sommets car il est alors trop long de faire les expérimentations pour tous les paramètres.

Dans un premier temps, on fixe p_r à 0, et on fait varier p_i de 1 à $0,25$ avec un pas de $0,25$, les classes des partitions initiales étant donc de moins en moins denses. On fait de plus varier le nombre de classes initiales nbc qui prendra les valeurs 2, 5, 10 et 20. Pour chaque jeu de paramètres, on construit 100 graphes sur lesquels on applique les étapes 1 (avec De_5) et 2 de la méthode. Les résultats sont présentés dans le Tableau 6. Comme dans la partie précédente, nous donnons les moyennes observées du nombre de classes, du pourcentage d'éléments classés, de la distance de transfert normalisée et du pourcentage de paires conservées entre les noyaux obtenus et les classes initiales pour chaque méthode et pour chaque valeur de (p_i, p_r).

D'une façon générale, on observe que moins la densité initiale des classes est élevée et plus il y a de classes dans la partition initiale, et plus les différentes méthodes ont du mal à retrouver les classes. Pour $p_i = 1$ (les classes initiales sont des cliques), toutes les méthodes retrouvent exactement les classes initiales (quasiment exactement pour $nbc = 20$) ; en fait même l'étape 1 permet de les retrouver. Pour les autres valeurs de p_i, l'étape 1 ne produit pas toujours le bon nombre de noyaux. Les classes sont cependant presque exactement retrouvées après l'étape 2 dans de nombreux cas.

Les autres cas correspondent à des classes moins distinctes dans le graphe, et il est donc normal que la méthode ne parvienne pas à les retrouver exactement. En fait, dans la plupart de ces cas, les noyaux obtenus ont une densité plus élevée que les classes initiales. On peut remarquer aussi que l'étape 2 classe une très grande majorité de sommets, le pourcentage d'éléments classés variant entre 90 % et 100 % quelle que soit la distance utilisée. Ces résultats montrent l'intérêt et l'efficacité de la méthode des nuées dynamiques pour améliorer les noyaux.

Étudions maintenant le comportement de l'étape 2 suivant la distance utilisée ($Dice$ ou Pcc). Au niveau du nombre de noyaux, l'indice Pcc a tendance à fournir un peu moins de noyaux que la distance de $Dice$. Aucune des deux méthodes ne semble pourtant se distinguer si on regarde l'écart au nombre de noyaux initial ; parfois l'une est meilleure (cas $p_i = 0,25$, $nbc = 10$) et parfois c'est l'autre (cas $p_i = 0,75$, $nbc = 20$). En ce qui concerne la distance de transfert et le pourcentage de paires conservées, les résultats sont aussi relativement similaires pour les deux distances

$\boldsymbol{p_i = 1}$	étape 1	étape 2 (Pcc)	étape 2 ($Dice$)
$nbc = 2$	2 | 98 0 | 100	2 | 100 0 | 100	2 | 100 0 | 100
$nbc = 5$	5 | 92 0 | 100	5 | 100 0 | 100	5 | 100 0 | 100
$nbc = 10$	10 | 82,8 0 | 100	10 | 100 0 | 100	10 | 100 0 | 100
$nbc = 20$	19,9 | 67 0 | 100	19,9 | 100 0,002 | 99,7	19,9 | 99,9 0,002 | 99,8
$\boldsymbol{p_i = 0,75}$	étape 1	étape 2 (Pcc)	étape 2 ($Dice$)
$nbc = 2$	2,04 | 47 0,002 | 100	2 | 100 0 | 100	2 | 100 0 | 100
$nbc = 5$	4,99 | 49,9 0,001 | 100	5,01 | 100 0 | 99,99	5,01 | 99,8 0 | 99,99
$nbc = 10$	8,8 | 51,9 0 | 99,8	9,45 | 99,1 0,05 | 99	9,99 | 99,9 0,003 | 99,9
$nbc = 20$	15,4 | 51,2 0,007 | 69,5	15,8 | 97,4 0,2 | 60,9	19,5 | 99,5 0,04 | 64,9
$\boldsymbol{p_i = 0,5}$	étape 1	étape 2 (Pcc)	étape 2 ($Dice$)
$nbc = 2$	2,25 | 36,8 0,04 | 100	2 | 100 0 | 100	2,02 | 100 0,004 | 99,5
$nbc = 5$	5,1 | 44 0,008 | 99,8	5 | 100 0 | 99,99	5,1 | 100 0,007 | 99,1
$nbc = 10$	10,2 | 46,6 0,04 | 85,2	10,11 | 99,4 0,03 | 85,8	11,2 | 99,7 0,04 | 80
$nbc = 20$	12,5 | 45 0,02 | 65,1	14,7 | 96,5 0,26 | 48,4	18,8 | 98,5 0,14 | 46,3
$\boldsymbol{p_i = 0,25}$	étape 1	étape 2 (Pcc)	étape 2 ($Dice$)
$nbc = 2$	2,9 | 28,3 0,17 | 77	2 | 100 0 | 100	2,8 | 99,5 0,15 | 82,1
$nbc = 5$	9,3 | 35,9 0,25 | 57,8	6,6 | 99,3 0,08 | 79	10,3 | 98,6 0,33 | 52,4
$nbc = 10$	9,3 | 35,5 0,15 | 51,6	11,2 | 97 0,15 | 46,8	15,4 | 97,9 0,27 | 35,6
$nbc = 20$	12,6 | 11,8 0,008 | 67,3	13,8 | 95,2 0,39 | 30,2	18,6 | 97,1 0,3 | 27

TABLEAU 6. Étude comparative des noyaux obtenus par l'étape 2 ($p_r = 0$)

avec De_5. Pour les cas $nbc = 20$, la distance de *Dice* semble préférable, pour les autres cas c'est plutôt *Pcc* qui construit les meilleures classes. Il semble donc pour l'instant assez difficile de trancher entre ces deux distances. Étudions maintenant le comportement de la méthode sur d'autres instances, engendrées en fixant cette fois-ci p_i à 1 et avec différentes valeurs de p_r. Pour chaque jeu de paramètres, on construit à nouveau quatre ensembles de 100 graphes contenant respectivement 2, 5, 10 et 20 classes initiales. Les résultats sont présentés dans le Tableau 7.

$\boldsymbol{p_r = 0,1}$	étape 1	étape 2 (*Pcc*)	étape 2 (*Dice*)
$nbc = 2$	2 | 50 0 | 100	2 | 100 0 | 100	2 | 100 0 | 100
$nbc = 5$	5 | 52 0 | 100	5 | 100 0 | 100	5 | 100 0 | 100
$nbc = 10$	9,8 | 50 0 | 100	9,8 | 99,8 0,02 | 99	9,99 | 99,98 0 | 99,96
$nbc = 20$	16,7 | 57,6 0 | 99,8	16,7 | 96,7 0,14 | 94,4	19,2 | 99,6 0,03 | 97,4
$\boldsymbol{p_r = 0,3}$	étape 1	étape 2 (*Pcc*)	étape 2 (*Dice*)
$nbc = 2$	1,5 | 33,9 0,131 | 97,5	1,2 | 100 0,41 | 100	1,5 | 100 0,26 | 100
$nbc = 5$	4,6 | 40,5 0,015 | 98,9	4,5 | 99,8 0,1 | 95,4	4,5 | 99,4 0,09 | 95,3
$nbc = 10$	9 | 44,6 0,01 | 99,9	8,9 | 99,2 0,11 | 92,3	9,2 | 98,8 0,07 | 95,2
$nbc = 20$	15,1 | 44,8 0,003 | 93,3	14,8 | 95,1 0,27 | 73,4	17,4 | 97,1 0,14 | 76,6
$\boldsymbol{p_r = 0,5}$	étape 1	étape 2 (*Pcc*)	étape 2 (*Dice*)
$nbc = 2$	1,25 | 32 0,37 | 89,9	1 | 100 0,5 | 100	1,1 | 100 0,48 | 97,8
$nbc = 5$	3,5 | 29,9 0,15 | 90,2	1,4 | 99,9 0,74 | 95,7	3,4 | 99,3 0,34 | 79,5
$nbc = 10$	7,6 | 34,7 0,07 | 95	5,4 | 96,8 0,5 | 70,2	8,5 | 97,9 0,24 | 74,6
$nbc = 20$	13,1 | 38,4 0,05 | 76,2	12,6 | 91,4 0,44 | 47,7	16,4 | 95,6 0,32 | 45,8

TABLEAU 7. Étude comparative des noyaux obtenus par l'étape 2 ($p_i = 1$)

Plus la valeur de p_r est grande, et moins il y a d'arêtes intraclasses, et plus il y a d'arêtes interclasses. Les classes initiales sont donc de moins en moins nettes, et en effet l'algorithme a de plus en plus de mal à les retrouver. Ici le nombre de classes

n'a pas tout à fait le même effet que précédemment puisque de lui dépend le nombre d'arêtes dans le graphe. C'est pourquoi il est parfois plus difficile de retrouver la partition initiale lorsque celle-ci possède deux classes que lorsqu'elle en a cinq : le nombre d'arêtes interclasses est beaucoup plus grand pour $nbc = 2$ que pour $nbc = 5$ (cas $p_r \geq 0,3$). Globalement la méthode fonctionne tout de même mieux quand le nombre de classes n'est pas trop grand.

L'étape 1 a tendance à fournir un nombre de noyaux légèrement insuffisant, et l'étape 2 ne parvient pas aussi bien que précédemment à créer de nouvelles classes. En fait, l'indice de distance *Pcc* produit même un nombre de noyaux inférieur ou égal au nombre de noyaux initiaux. La distance de *Dice* réagit mieux car elle parvient à augmenter le nombre de noyaux pour se rapprocher du nombre de classes initial *nbc*. Pour $p_r = 0,5$, les méthodes, et plus particulièrement *Pcc*, ont tendance à produire des classes qui contiennent les classes initiales (la distance de transfert est élevée mais le pourcentage de paires conservées l'est aussi). On peut remarquer de façon générale que les noyaux obtenus avec l'utilisation de la distance de *Dice* sont nettement plus proches des classes de la partition initiale que ceux construits à partir de *Pcc*. D'après ces résultats, il semble donc préférable d'utiliser la distance de *Dice* plutôt que l'indice *Pcc*.

3.4. TEST DE L'ÉTAPE 3 : EXTENSION DES CLASSES

Jusqu'à l'étape 2, les classes obtenues ne sont pas empiétantes. Pour tester la méthode, il nous a donc suffi de comparer les classes obtenues à celles de la partition initiale. Pour tester l'étape 3, il ne semble pas intéressant de partir d'un graphe possédant des classes chevauchantes car il devient alors difficile d'interpréter et de comparer les résultats. Afin de tester les différentes méthodes d'extension, nous partirons d'un graphe contenant une partition initiale des sommets, et nous nous concentrerons sur les caractéristiques des classes obtenues (cardinal et densité des classes, degré moyen dans les classes, degré d'empiètement). Ces critères vont nous permettre de comparer entre elles différentes solutions, et de valider la méthode.

Commençons par tester l'extension sur deux groupes de 100 graphes possédant 100 sommets et 10 classes initiales, engendrés aléatoirement comme décrit plus haut. Le premier groupe (groupe A) est construit avec $p_i = 1$ et $p_r = 0,5$, le second (groupe B) avec $p_i = 0,5$ et $p_r = 0$. Ces deux jeux de paramètres correspondent à des classes de densité environ 0,5 ; dans le premier cas la densité globale des graphes est en moyenne de 0,09, dans le second elle vaut 0,04. Il y a donc environ deux fois plus d'arêtes dans les graphes du groupe A que dans ceux du groupe B, mais autant dans les classes, les classes y sont donc moins bien définies. Le tableau suivant donne les caractéristiques moyennes des noyaux produits à partir de ces graphes par l'étape 2 avec De_5 et la distance de *Dice* : le nombre de classes *nbc*, le pourcentage d'éléments classés, le cardinal moyen des classes, la densité moyenne des classes et le degré moyen des sommets dans les sous-graphes induits par les classes.

groupe	*nbc*	% d'élts classés	card. moyen	dens. moyenne	deg. moyen
$A : p_i = 1,\ p_r = 0,5$	8,5	97,9	11,9	0,43	4,5
$B : p_i = 0,5,\ p_r = 0$	11,2	99,7	8,9	0,56	4,2

On applique donc maintenant l'étape 3 sur ces noyaux avec les deux critères d'extension : critère du degré moyen et critère probabiliste (définis dans la partie 2.6.). Pour le critère du degré moyen, on fait varier le paramètre α de 0,1 à 2,5 avec un pas de 0,1. Le cardinal moyen, le nombre moyen de classes par sommet, la densité moyenne des classes et le degré moyen des sommets dans les classes sont représentés dans le Tableau 8 pour les deux groupes de graphes et les différentes méthodes d'extension.

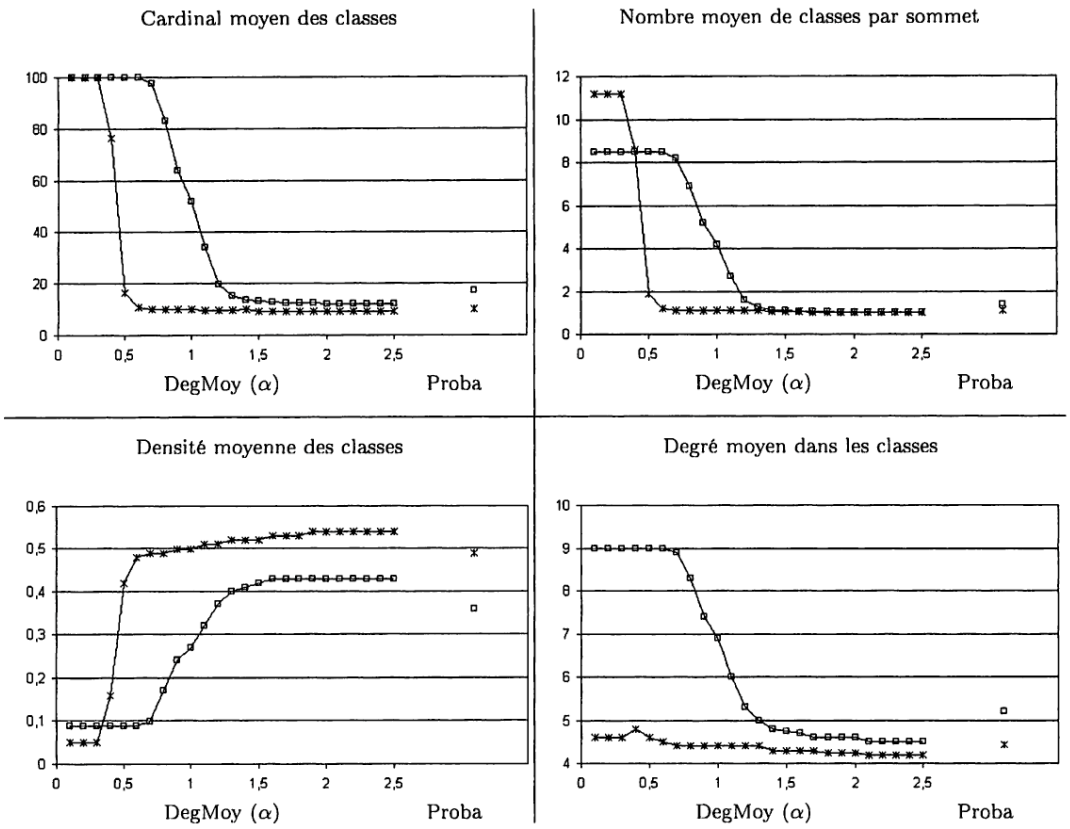


TABLEAU 8. Caractéristiques des classes après l'extension ($n = 100, nbc = 10$)

En ce qui concerne le critère du degré moyen, les deux courbes correspondant aux deux groupes de graphes considérés se comportent globalement de la même façon. On remarque tout d'abord que quand α augmente (c'est-à-dire quand le critère d'extension devient plus strict), les classes deviennent de plus en plus petites, de moins en moins empiétantes, mais de plus en plus denses en arêtes. Il est en effet intuitif que plus une classe est petite, plus il est facile qu'elle soit dense (une classe à deux sommets sera automatiquement de densité 1). Si α est proche de zéro, le critère est totalement relâché et les classes sont étendues par tous les sommets du graphe. Si, au contraire, α est grand, les classes sont alors réduites aux noyaux donnés par l'étape 2.

On note toutefois, en raison de la différence de densité dans les graphes, que pour le groupe A ($p_i = 1\,;\ p_r = 0,5$), les classes construites sont plus grosses, et l'empiètement est plus important que pour les graphes du groupe B ($p_i = 0,5\,;\ p_r = 0$).

On observe que les valeurs de α pertinentes pour tous les critères simultanément (cardinal compris entre dix et vingt, empiètement entre un et deux, densité élevée, degré moyen encore assez important) sont dans $[1,2\ ;\ +\infty[$ pour le groupe A et dans $[0,5\ ;\ +\infty[$ pour le groupe B, $\alpha = +\infty$ correspondant à ne pas étendre les classes, les classes finales étant alors égales aux noyaux construits par l'étape 2. On choisit ensuite α suivant qu'on veut plutôt privilégier la densité des classes, leur degré moyen ou leur taille.

Le paramètre α présente l'avantage de permettre une plus grande souplesse de la méthode, qui peut être adaptée à l'application ou aux exigences de l'utilisateur. Il permet aussi de construire une hiérarchie des classes, les classes étant à chaque fois incluses les unes dans les autres quand α varie. Cette méthode présente toutefois l'inconvénient d'exiger un travail d'étalonnage à effectuer pour chaque graphe étudié afin de choisir la valeur de α qui convient.

En ce qui concerne le critère probabiliste, les résultats sont assez conformes à ceux obtenus avec le critère du degré moyen. Pour le groupe A, il correspond à peu près à $\alpha = 1,25$; pour le groupe B, il correspond à $\alpha = 0,7$. Les caractéristiques des classes obtenues sont donc satisfaisantes ; l'empiètement et le cardinal sont modérés, la densité et le degré moyen dans les classes sont corrects. L'avantage de ce critère est qu'il s'adapte automatiquement aux caractéristiques du graphe initial. On peut aussi utiliser ce critère pour guider la recherche d'une bonne valeur de α pour le critère du degré moyen.

4. APPLICATION À UN GRAPHE D'INTERACTIONS ENTRE PROTÉINES

Dans cette partie, on s'intéresse à l'application de la méthode proposée à un graphe réel d'interactions entre protéines. Le graphe étudié, appelé Toth868 et représenté[3] Figure 1, est extrait du réseau d'interactions des protéines chez la levure. Il possède 868 sommets et 2000 arêtes, ce qui correspond à une densité de 0,005 et à un degré moyen de 4,61.

Tout d'abord, nous présenterons les résultats donnés par la méthode de classification empiétante sur ces graphes. Nous appliquerons les trois étapes de la méthode avec De_5 comme densité locale et *Dice* comme indice de distance. La méthode d'extension sera appliquée avec le critère du degré moyen pour différentes valeurs de α et le critère probabiliste. Enfin nous validerons les classes d'un point de vue biologique en montrant qu'elles correspondent à des processus biologiques.

4.1. ANALYSE COMBINATOIRE DES CLASSES

Commençons par étudier les noyaux fournis par les deux premières étapes de la méthode appliquée à ce graphe. Le tableau suivant présente les caractéristiques des

[3]Cette représentation a été obtenue grâce au logiciel *pajek* ([Batagelj, Mrvar, 2003, 2008]).

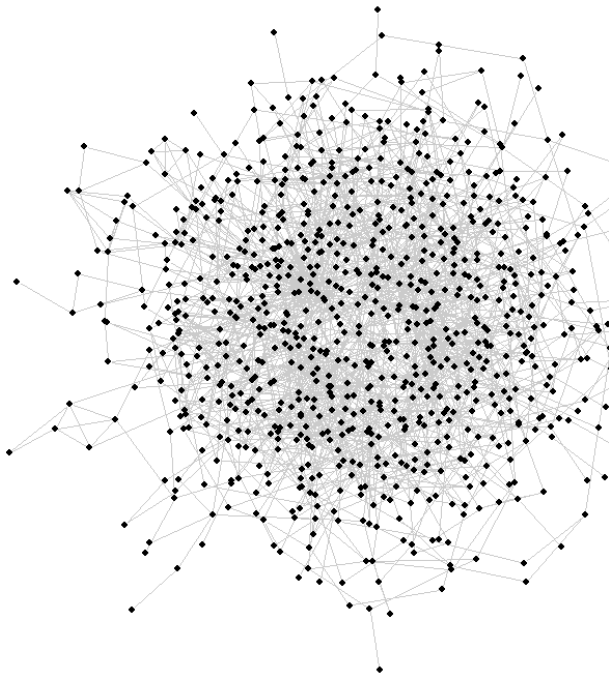


FIGURE 1. Toth868

classes au cours de ces deux étapes (nombre de classes *nbc*, pourcentage d'éléments classés, cardinal moyen, densité moyenne, degré moyen des classes).

	nbc	% d'élts classés	card. moyen	dens. moyenne	deg. moyen
étape 1	71	27,65	3,38	0,88	1,77
étape 2	232	96,66	3,62	0,75	1,41

On constate que l'étape 2 forme énormément plus de noyaux que l'étape 1 et classe une très grande majorité des sommets. Le cardinal des noyaux n'augmente pas de façon significative entre les deux étapes, ce qui s'explique par la faible densité du graphe. Les noyaux fournis par la méthode des nuées dynamiques sont ici nombreux et de cardinal très faible.

Appliquons maintenant l'étape d'extension à ces noyaux. On présente Tableau 9 quatre graphiques donnant les caractéristiques moyennes des classes formées avec le critère du degré moyen pour α variant dans $[0,9\ ;\ 2,1]$ avec un pas de 0,1 ainsi qu'avec le critère probabiliste. Le critère du degré moyen produit des classes de cardinal plus ou moins élevé suivant la valeur de α : plus α augmente, moins les classes sont étendues, moins l'empiètement est important et plus la densité moyenne des classes est élevée. Le critère probabiliste donne des classes de cardinal relativement élevé (28 sommets en moyenne) et donc un empiètement des classes relativement important (7,6 classes par sommet en moyenne).

On remarque qu'à cardinal moyen égal (27,5 obtenu pour $\alpha = 1,11$), les classes fournies par l'extension probabiliste sont moins denses et ont un degré moyen moins élevé que celles fournies par l'extension du degré moyen. Observons la répartition des classes par cardinaux pour ces deux solutions (Figure 2). On voit nettement sur cette

figure qu'avec la méthode d'extension probabiliste, il y a beaucoup plus de classes de cardinal moyen, alors qu'avec le degré moyen, il y a beaucoup de classes très petites et très grandes. Ceci explique les meilleurs résultats en moyenne pour ce critère, qui donne en fait moins de classes exploitables pour l'interprétation biologique.

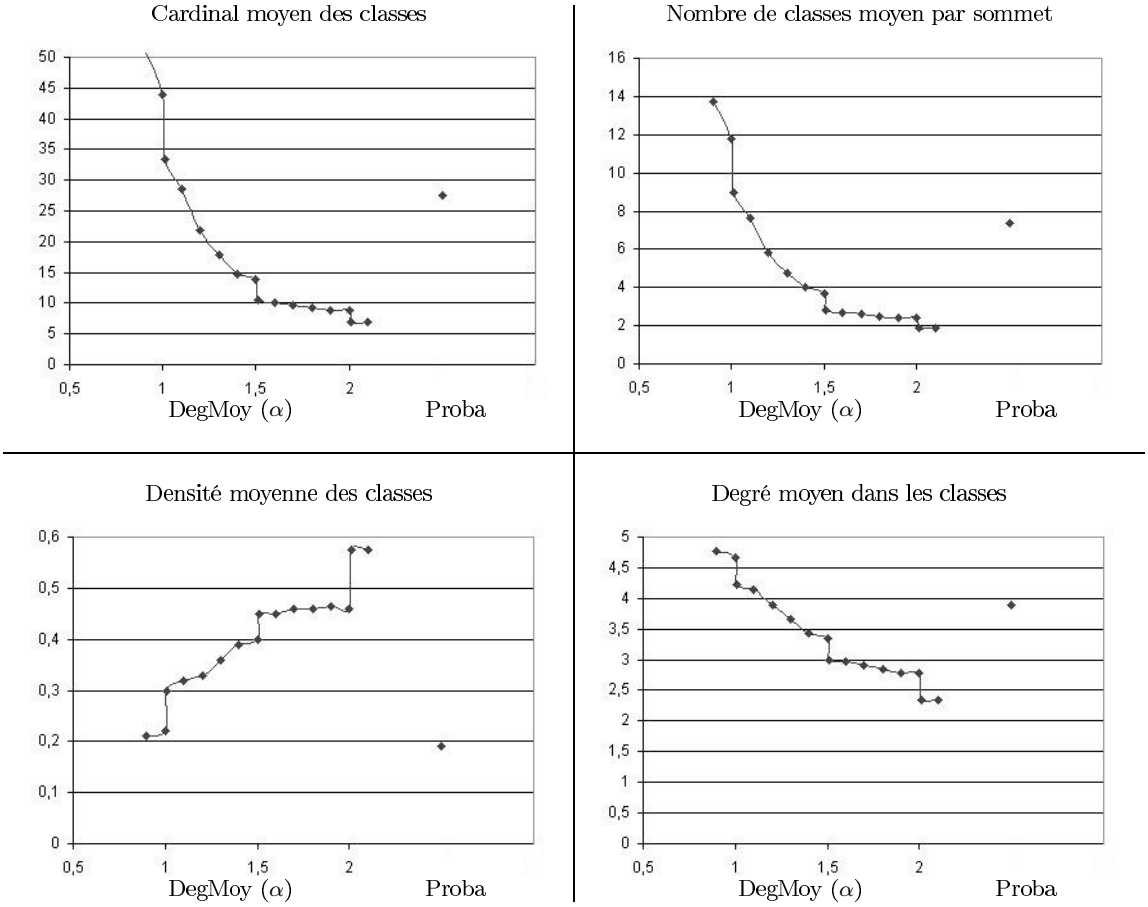


TABLEAU 9. Toth868 : caractéristiques moyennes des classes après l'extention

On donne dans le Tableau 10 les représentations des différents critères d'extension sur une classe : le critère du degré moyen avec $\alpha = 2$, $\alpha = 1,5$ et $\alpha = 1,11$ et le critère d'extension probabiliste. On remarque que le cardinal de la classe augmente avec chaque diminution de α, et la classe obtenue par le critère probabiliste est la plus étendue. On observe un comportement similaire de la méthode pour l'extension des autres classes du graphe.

La méthode parvient à détecter les zones denses en arêtes de ce graphe ; les classes formées sont de cardinal modéré et semblent correspondre pour la plupart à des configurations remarquables. Les différents critères d'extension permettent d'obtenir une hiérarchie des classes, et d'apporter ainsi un aspect dynamique aux classes, qui rendra leur interprétation plus facile.

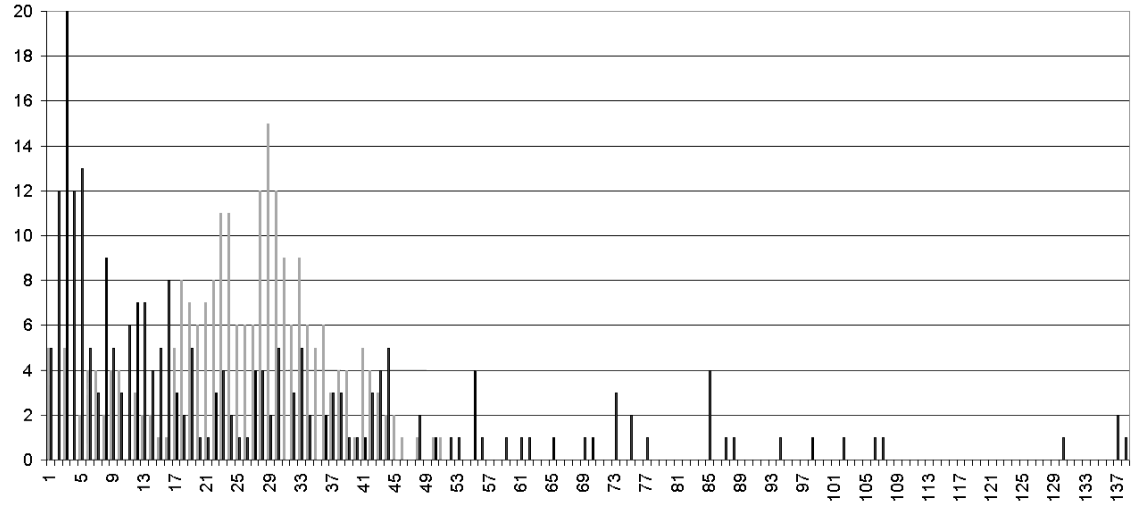


FIGURE 2. Répartition des classes par cardinaux
(en gris : extension probabiliste - en noir : extension du degré moyen $\alpha = 1,11$)

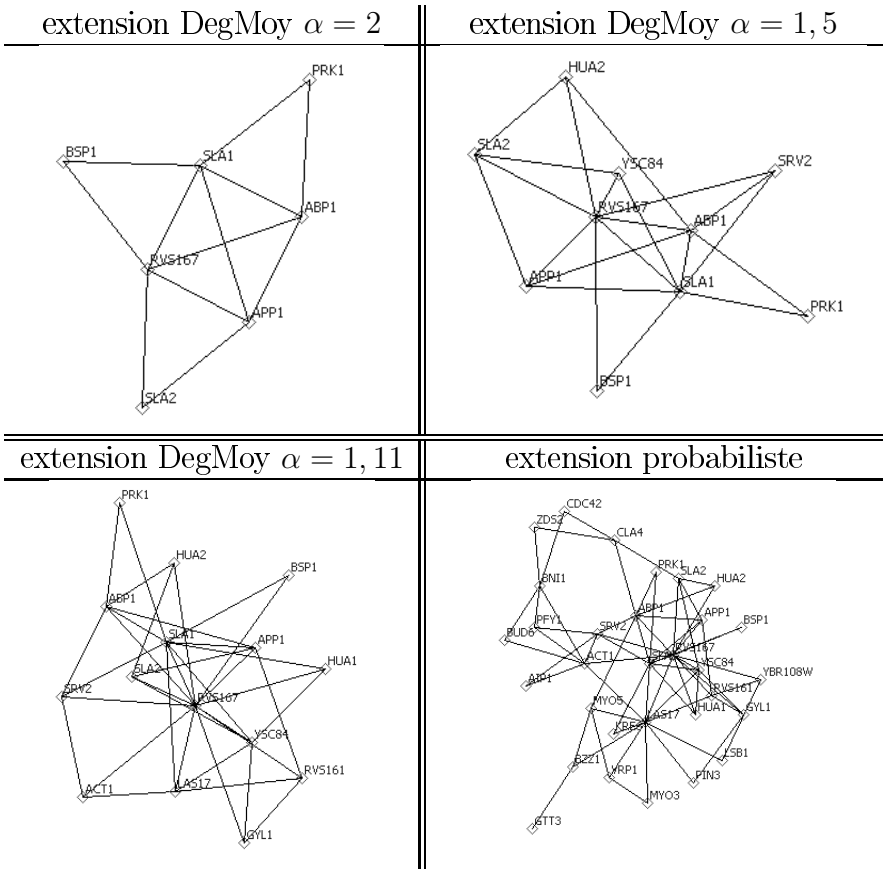


TABLEAU 10. Toth868 : caractéristiques moyennes des classes après l'extention

4.2. VALIDATION BIOLOGIQUE

4.2.1. Méthodologie

Afin de valider la méthode dans le cadre d'une application aux graphes d'interactions entre protéines, on doit s'assurer que les classes produites correspondent à des groupes de protéines partageant la (les) même(s) fonction(s) cellulaire(s). Pour cela, on utilise le serveur *GOToolBox* (cf. [Martin *et al.*, 2004]), qui fournit, à partir d'une liste de protéines, la liste des fonctions surreprésentées dans le groupe suivant différents tests statistiques.

Ce serveur est fondé sur les données fonctionnelles du projet *Gene Ontology* (GO) [2004] qui vise à fournir un vocabulaire spécifique à chaque domaine biologique (fonctions moléculaires, processus biologiques, composants cellulaires) et une classification de ces termes pour décrire les produits des gènes dans tous les organismes. Les différentes fonctions (ou processus) sont organisées en un arbre. La racine *Gene Ontology*, correspondant au niveau 0, possède trois fils : *cellular component*, *biological process* et *molecular function*, correspondant au niveau 1 de l'arbre. On ne s'intéresse ici qu'au sous-arbre issu de *biological process*. Plus on s'éloigne de la racine de l'arbre (niveaux de plus en plus élevés), et plus les fonctions sont spécialisées. Cet arbre n'en est en fait pas tout à fait un car il existe quelques arêtes transversales, de sorte qu'il arrive qu'un même nœud corresponde à plusieurs niveaux de l'arbre. Pour de tels nœuds, nous ne prendrons en compte que le niveau le plus élevé.

On utilise donc *GoToolBox* sur nos classes de protéines, avec la loi hypergéométrique et la correction de Bonferroni. On sélectionne, parmi les fonctions surreprésentées dans chaque classe C, celles qui sont possédées par au moins 20 % des protéines de la classe. On appelle F_C l'ensemble de ces fonctions. Soit $f \in F_C$. On note $\Gamma(C)$ l'ensemble composé des éléments de la classe C et des éléments adjacents à C. On définit alors les valeurs suivantes :

nombre de protéines parmi $\Gamma(C)$ qui :	possèdent la fonction f	ne possèdent pas la fonction f
appartiennent à C	n_{fC}	$n_{\bar{f}C}$
n'appartiennent pas à C	$n_{f\bar{C}}$	$n_{\bar{f}\bar{C}}$

Ensuite on calcule, pour chacune de ces fonctions :

- la proportion de protéines de la classe participant à cette fonction (*valeur prédictive positive*) :
$$VP^+(f,C) = \frac{n_{fC}}{n_{fC} + n_{\bar{f}C}} \ ;$$
- la proportion de protéines parmi les voisins de la classe ne participant pas à cette fonction (*valeur prédictive négative*) :
$$VP^-(f,C) = \frac{n_{\bar{f}\bar{C}}}{n_{f\bar{C}} + n_{\bar{f}\bar{C}}} \ ;$$
- la proportion de protéines appartenant à la classe parmi les protéines de $\Gamma(C)$ possédant la fonction f (*sensibilité*) :
$$SE(f,C) = \frac{n_{fC}}{n_{fC} + n_{f\bar{C}}};$$

– la proportion de protéines n'appartenant pas à C parmi les protéines de $\Gamma(C)$ qui ne possèdent pas la fonction f (*spécificité*) :

$$SP(f,C) = \frac{n_{\bar{f}\bar{C}}}{n_{\bar{f}C} + n_{\bar{f}\bar{C}}}.$$

Ces quatre indices standards permettent d'évaluer l'intensité de la relation entre les deux variables : « être dans la classe C » et « posséder la fonction f ». Pour qu'une classe soit en adéquation avec une fonction, il faut que ces indices soient élevés (proches de 1).

Remarque 3. Pour le calcul de ces indices, nous nous contentons de prendre en compte le voisinage de la classe. En effet, des valeurs élevées dans cet ensemble suffisent à convaincre de l'intérêt de la classe. De plus, si dans le graphe il existe d'autres zones de protéines possédant la même fonction, mais que ces zones ne sont pas adjacentes à la classe, il n'est alors pas pertinent d'en tenir compte lorsqu'on évalue l'adéquation de la classe à la fonction. En effet la méthode n'a pas de raison de regrouper ces différentes zones dans une même classe, qui ne pourra alors pas être dense en arêtes, voire même pas connexe.

En général, on se contente de la sensibilité et de la spécificité pour évaluer la qualité d'une prédiction. L'indice de Youden (cf. [Schwartz, 1984]), compris entre -1 et 1, est défini par :

$$Y(f,C) = SE(f,C) + SP(f,C) - 1.$$

Il permet de prendre en compte directement ces deux indices, qui doivent être élevés tous les deux pour prouver le lien entre C et f. Si l'indice de Youden est négatif, alors ce lien est faible, plus il est proche de 1, et plus il est fort.

Nous proposons de définir de manière similaire un indice noté Y', lui aussi compris entre -1 et 1, regroupant les valeurs prédictives positive et négative :

$$Y'(f,C) = VP^{+}(f,C) + VP^{-}(f,C) - 1.$$

Nous définissons ensuite le score d'une fonction f par rapport à la classe C par :

$$sc(f,C) = \frac{Y(f,C) + Y'(f,C)}{2},$$

et le score de la classe C est donné par :

$$SC(C) = max_{f \in F_C} sc(f,C).$$

Ces deux nouveaux indices sont eux aussi compris entre -1 et 1.

Nous allons maintenant utiliser *GOToolBox* sur les classes construites par la méthode à partir du graphe Toth868. Nous évaluerons la qualité de ces classes en terme de score.

4.2.2. Résultats

Nous allons utiliser les données fonctionnelles des protéines pour valider les classes et comparer les différentes méthodes d'extension. Pour l'étape 3, on utilise le critère d'extension du degré moyen pour $\alpha = 2$, $\alpha = 1,5$ et $\alpha = 1,11$ ainsi que le critère d'extension probabiliste. Dans ce graphe de 868 protéines, 10 (soit environ 1%) sont de fonction inconnue. Parmi ces 10 protéines non annotées, 9 sont classées par l'étape 2. La dernière est classée par toutes les méthodes sauf avec le degré moyen pour $\alpha = 2$. La méthode pourra donc permettre la prédiction de fonctions pour toutes les protéines non annotées du graphe.

On détermine ensuite, grâce a *GOToolBox*, pour chaque jeu de classes, la (ou les) fonction(s) correspondant au score maximum pour chaque classe. On donne dans le Tableau 11, pour chaque méthode, le cardinal moyen des classes, le nombre moyen de protéines annotées dans les classes, le niveau moyen des meilleures fonctions, la moyenne des valeurs prédictives positives des meilleures fonctions et la moyenne des scores. Nous avons choisi de privilégier la valeur prédictive positive, le but étant de prédire l'appartenance d'une protéine à une fonction. Lorsque aucune fonction n'a été trouvée pour une classe, celle-ci n'est pas prise en compte dans le calcul des moyennes du niveau, de VP^+ et de SC.

	cardinal moyen	nb moyen de prot. ann.	niveau moyen	VP^+ moyenne	SC moyen
$\alpha = 2$	8,66	8,58	7,62	0,52	0,44
$\alpha = 1,5$	13,95	13,84	7,56	0,51	0,44
$\alpha = 1,11$	27,72	27,50	7,10	0,53	0,45
proba	27,23	27,1	6,7	0,45	0,38

TABLEAU 11. Toth868 : scores et valeurs prédictives positives des classes pour différentes extensions

On constate que plus les classes sont étendues par l'étape 3, et plus le niveau moyen des meilleures fonctions est petit, mais il reste tout de même relativement élevé (entre 6,7 et 7,62). Les moyennes des valeurs prédictives positives et des scores sont relativement stables pour toutes les méthodes d'extension ; on constate tout de même que l'extension probabiliste donne des résultats un peu moins satisfaisants suivant ces critères que le degré moyen, et que celui-ci est d'autant meilleur que α est petit.

L'histogramme représenté Figure 3 donne la répartition des classes par score pour les différents critères d'extension. On constate que moins le critère d'extension est strict, plus les classes ont des scores centrés sur la moyenne. Il apparaît pour ce graphe que le critère probabiliste est moins satisfaisant puisqu'il fournit tout de même beaucoup de classes ne correspondant à aucune fonction, et peu de classes de scores très élevés. Le critère du degré moyen avec $\alpha = 1,11$ donne un score moyen un peu plus élevé que pour des valeurs supérieures de α et moins de classes sans fonction associée, mais produit aussi une moindre proportion de classes de scores très élevés. Il semble assez délicat de déterminer la meilleure valeur du paramètre α.

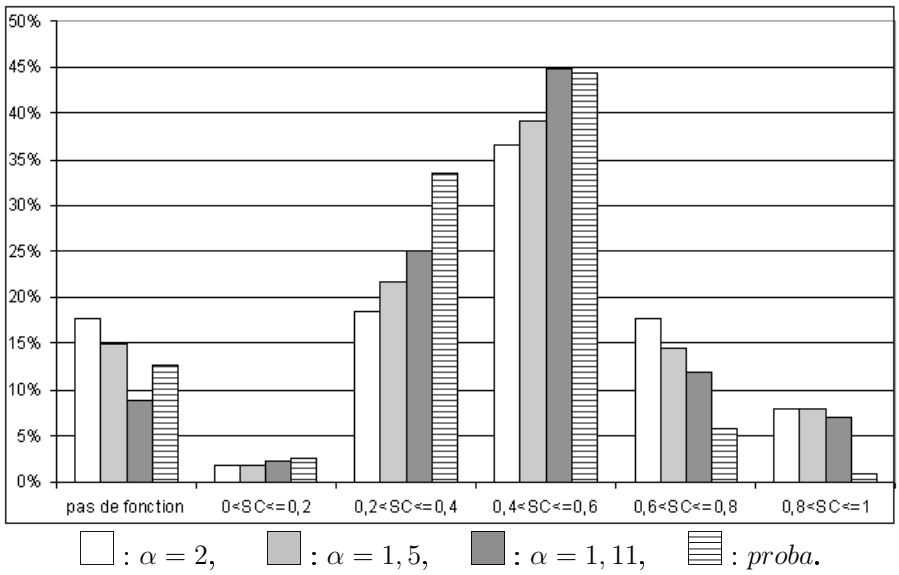


FIGURE 3. Toth868 : répartition des classes par score pour différentes extensions

La Figure 4 représente la répartition des classes par valeur prédictive positive. On s'aperçoit sur cet histogramme que, bien que les valeurs prédictives positives moyennes soient autour de 0,5, il y a un grand nombre de classes pour lesquelles elles sont élevées. On observe à nouveau que le critère probabiliste produit moins de bonnes classes que le critère du degré moyen. Il n'y a aucune classe de valeur prédictive positive inférieure à 0,2 car nous n'avons sélectionné que des fonctions contenant au moins 20 % des protéines de la classe.

Les valeurs étudiées jusqu'ici ne correspondent qu'aux meilleures fonctions de chaque classe, c'est-à-dire correspondant aux scores maximaux. Pour chaque classe, il y a en fait plusieurs fonctions sélectionnées et possédant un score satisfaisant. Afin d'interpréter les classes d'un point de vue biologique, il est donc intéressant de prendre en compte toutes ces fonctions. Nous avons représenté sur la Figure 5 une classe et son voisinage, les étiquettes des sommets étant les noms des protéines associées. Les sommets de la classe sont représentés par un losange ; le losange est grisé lorsque la protéine correspondante n'est pas annotée. Les sommets du voisinage de la classe sont représentés par une croix. Nous avons matérialisé les fonctions possédant les meilleurs scores pour ces classes par des zones délimitées en pointillés : les protéines situées à l'intérieur de la zone possèdent la fonction correspondante, les protéines à l'extérieur ne la possèdent pas. Le tableau suivant donne les caractéristiques des fonctions représentées dans la Figure 5 : identifiant dans *Gene Ontology* (*GOID*), score pour la classe, niveau, nom.

GOID	score	niveau GO	nom
GO : 0006810	1	5	*transport*
GO : 0006886	0,77	8	*intracellular protein transport*
GO : 0007018	0,6	9	*microtubule-based movement*

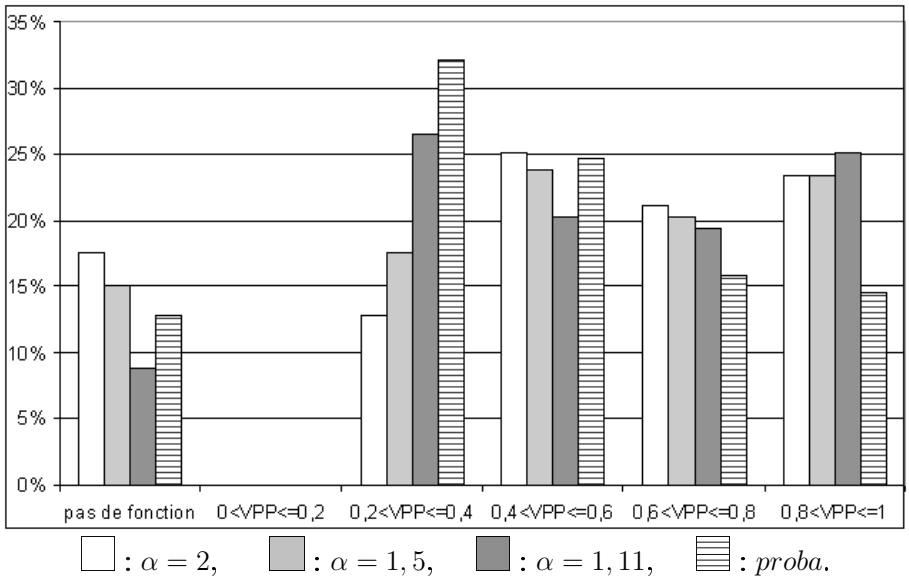


FIGURE 4. Toth868 : répartition des classes par valeur prédictive positive pour différentes extensions

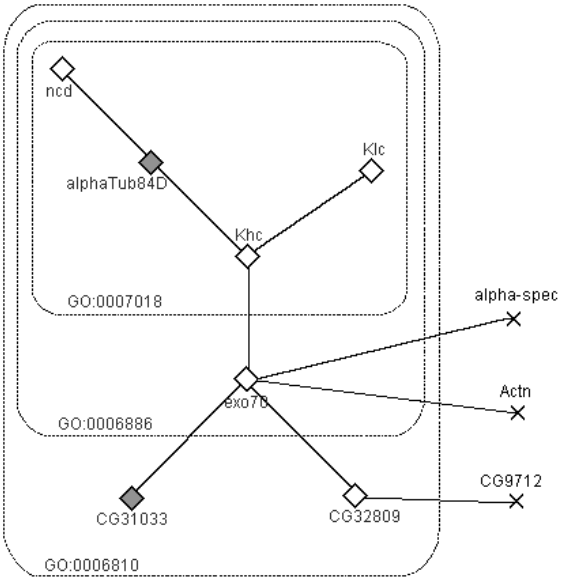


FIGURE 5. Une des classes obtenues, associée au processus de transport

Pour cette classe, trois fonctions possèdent des scores élevés. Toutes les protéines de la classe participent à des processus de transport, mais aucune des voisines de la classe n'y participe. Les fonctions *intracellular protein transport* et *microtubule-based movement* sont toutes deux des descendantes de la fonction *intracellular transport*, elle-même fille de la fonction moins spécialisée *transport*. Deux protéines ne sont pas annotées, et on pourrait donc se servir de ces résultats pour prédire l'appartenance de CG31033 au processus de transport, et faire une prédiction plus spécifique pour alphaTub84D que l'on peut supposer participer aux deux fonctions *intracellular protein transport* et *microtubule-based movement*.

Pour ce graphe, la méthode fournit un grand nombre de classes, dont une majorité sont en adéquation avec des processus biologiques. L'extension probabiliste semble se comporter moins bien que l'extension du degré moyen. Le choix du paramètre α pour cette dernière reste délicat. Peut-être serait-il intéressant, en vue d'une utilisation de la méthode pour la prédiction de fonction, de prendre en compte les différents niveaux d'extension possibles, et de choisir le plus adapté à chaque classe. Ceci pourrait même être fait de façon systématique en intégrant les données biologiques dans le programme.

5. CONCLUSION

La méthode de classification présentée ici est issue d'une problématique biologique : étant donné un réseau d'interactions entre protéines, former des classes de protéines interagissant fortement entre elles, et correspondant donc à des complexes de protéines associés à des fonctions cellulaires, l'objectif final étant de prédire les fonctions inconnues de certaines protéines. Les classes doivent pouvoir être chevauchantes car une protéine est susceptible d'intervenir dans plusieurs fonctions cellulaires. La méthode proposée ne repose pas sur l'optimisation d'une fonction objectif, trop délicate à mettre au point, et le nombre de classes n'est pas connu à l'avance. On souhaite simplement que la méthode détecte des classes intrinsèques au graphe de départ, quitte à ne pas classer tous les sommets, et que les classes obtenues aient des caractéristiques pertinentes d'un point de vue biologique (cardinal pas trop important, empiètement modéré) afin de faciliter leur interprétation.

La méthode proposée se déroule en trois étapes. Tout d'abord on crée des noyaux initiaux des classes en sélectionnant les optima locaux d'une fonction de densité locale définie sur les sommets du graphe. On améliore ensuite ces noyaux par une adaptation de la méthode des nuées dynamiques qui modifie les noyaux ainsi que leur nombre. Ces deux premières étapes forment des classes disjointes. Dans la troisième et dernière étape, les noyaux sont étendus de manière empiétante suivant un critère sur la qualité des classes obtenues.

La méthode a été validée d'un point de vue combinatoire et algorithmique en l'appliquant à des graphes engendrés aléatoirement. Nous avons montré que la méthode proposée parvient efficacement à retrouver des classes de sommets denses en arêtes initialement placées dans les données. Ces tests ont aussi permis d'étudier et de régler différents paramètres et variantes de la méthode.

L'application à un graphe réel d'interactions entre protéines donne des résultats encourageants en terme de caractéristiques des classes qui semblent, pour la plupart, être en adéquation avec des processus biologiques. Reste maintenant à utiliser ces résultats pour prédire les fonctions de certaines protéines non annotées, et à vérifier le bien-fondé de ces prédictions d'un point de vue biologique. Pour cela on pourrait par exemple choisir un certain nombre de protéines annotées, dont on oublierait les fonctions temporairement, afin de vérifier si les prédictions établies par la méthode permettent de les retrouver.

BIBLIOGRAPHIE

ALBERTS B., BRAY D., LEWIS J., CARLIER N., BUTOR C., KAHN A., *Biologie moléculaire de la cellule*, Paris, Flammarion, 1995.

ALPERT C. J., KAHNG A. B., "Recent direction in netlist partitioning : a survey", *Integration : the VLSI Journal* 19, 1-2, 1995, p. 1-81.

ARABIE P., HUBERT L. J., DE SOETE G., *Clustering and classification*, River Edge (NJ), World Scientific Publishing, 1996.

BADER G., HOGUE C., "An automated method for finding molecular complexes in large protein interaction networks", *MC bioinformatics* 4(2), 2003.

BATAGELJ V., MRVAR A., "Pajek - Analysis and Visualization of Large Networks", in Jünger M., Mutzel P. (eds.), *Graph Drawing Software*, Berlin, Springer, 2003, p. 77-103.

BATAGELJ V., MRVAR A., Networks/Pajek, Program for Large Network Analysis, 2008/ http ://pajek.imfm.si/doku.php

BRUCKER P., "On the complexity of clustering problems", *Optimization and Operations Research, Lecture Notes in Economics and Mathematical Systems* 157, M. Beckmann, H. Künzi (eds), Heidelberg, Springer-Verlag, 1978, p. 45-54.

BRUCKER P., BARTHÉLEMY J.-P., *Éléments de classification*, Paris, Hermès, 2007.

BRUN C., WOJCIK J., GUÉNOCHE A., JACQ B., « Étude bioinformatique des réseaux d'interactions : PRODISTIN, une nouvelle méthode de classification fonctionnelle des protéines », *Actes des Journées ouvertes en Biologie, Informatique et Mathématiques JOBIM'2002*, Saint-Malo, 2002, p. 171-182.

CHARON I., DENŒUD L., GUÉNOCHE A., HUDRY O., "Maximum transfer distance between partitions", *Journal of Classification* 23(1), 2006, p. 103-121.

CHARON I., DENŒUD L., HUDRY O., « Maximum de la distance de transfert à une partition donnée », *Mathématiques et Sciences humaines* 179, 2007(a), p. 45-83.

CHARON I., DENŒUD L., HUDRY O., "Overlapping clustering in a graph using k-means and application to protein interactions networks", *Selected contributions in classification and data analysis*, sous la direction de P. Brito, P. Bertrand, G. Cucumel et F. De Carvalho, Heidelberg, Springer, Coll. Studies in classification, data analysis and knowledge organization, 2007(b), p. 173-182.

CHEN Y., XU D., "Genome-scale protein function prediction in yeast saccharomyces cerevisiae through integrating multiple sources of high-throughput data", *Proceedings of the Pacific Symposium on Biocomputing* 10, 2005, p. 471-482.

COLOMBO T., QUENTIN Y., GUÉNOCHE A., "Looking for high density areas in a graph ; application to orthologous genes", *Colloque Knowledge Discovery and Discrete Mathematics, Actes des Journées Informatiques de Metz*, INRIA, 2003, p. 203-212.

CORMEN T., LEISERSON C., RIVEST R., *Introduction à l'algorithmique*, Paris, Dunod, 1994.

DENŒUD L., *Étude de la distance de transfert entre partitions et recherche de zones denses dans un graphe*, thèse de doctorat, Université Paris 1, 2006.

DENŒUD L., "Transfer distance between partitions", *Advances in Data Analysis and Classification* 2, 2008, p. 279-294.

DENŒUD-BELGACEM L., "Transfer distance between partitions and search of dense zones in graphs", *4OR*, 2009, [à paraître].

DENŒUD L., CHARON I., GUÉNOCHE A., HUDRY O., « Classes empiétantes dans un graphe et application aux interactions entre protéines », *ROADEF'05*, sous la direction de J.-C. Billaut et C. Esswein, Presses universitaires François Rabelais, 2005, 393-408.

DENŒUD L., GARRETA H., GUÉNOCHE A., "Comparison of distance indices between partitions", *Proceedings of Applied Stochastic Models and Data Analysis*, Ph. Lenca *et al.* (eds.), on CD-Rom, Brest, 2008.

DENŒUD L., GUÉNOCHE, "Comparison of distance indices between partitions", *Proceedings of IFCS'2006, Data Science and Classification*, V. Batagelj *et al.* (eds.), Springer, 2006, p. 21-28.

DICE L. R., "Measures of the amount of ecologic association between species", *Ecology* 26, 1945, p. 297-302.

DIDAY E., « Une nouvelle méthode en classification automatique et reconnaissance des formes : la méthode des nuées dynamiques », *Revue de Statistique appliquée*, vol. XIX (2), 1971, p. 19-33.

ELSNER U., *Graph Partitioning : a survey*, Technische Universität Chemnitz, Allemagne, SFB 393, 1997.

ETIENNE J., MILLOT F., *Biochimie génétique, biologie moléculaire*, abrégé de médecine, Paris, Masson, 1998.

THE GENE ONTOLOGY CONSORTIUM, "The Gene Ontology (GO) database and informatics resource", *Nucleic Acids Res.*, vol. 32, 2004, D258-D261.

HANSEN P., JAUMARD B., "Cluster analysis and mathematical programming", *Journal of Mathematical Programming*, vol. 79, 1997, p. 191-215.

JOHNSON S. C., "Hierarchical clustering schemes", *Psychometrika* 32(3), 1967, p. 241-254.

MARTIN D., BRUN C., REMY E., MOUREN P., THIEFFRY D., JACQ B., "GOToolBox : functional analysis of gene datasets based on Gene Ontology", *Genome Biology*, 5(12), R101, 2004.

SAMANTA M. P., LIANG S., "Predicting protein functions from redundancies in large-scale protein interaction networks", *Proceedings of the National Academy of Sciences of the United States of America*, 100, 2003, p. 12579-12583.

SCHWARTZ D., *Méthodes statistiques à l'usage des médecins et des biologistes*, Paris, Flammarion, coll. Médecine-Sciences, 1984.

SCHWIKOWSKI B., UETZ P., FIELDS S., "A network of protein-protein interactions in yeast", *Nature Biotechnology*, vol. 18, 2000, p. 1257-1261.

VASQUEZ A., FLAMMINI A., MARITAN A., VESPIGNANI A., "Global protein function prediction from protein-protein interaction networks", *Nature Biotechnology* 21, 2003, p. 697-700.

WATTS D. J., STROGATZ S.H., "Collective dynamics of small-world networks", *Nature* 393, 1998, p. 440-442.