ENSAE

Projet Python

Détection de communautés en physique des hautes énergies

Auteurs :
Matteo Amestoy
Yann Lefeuvre

Superviseur :
Xavier Dupré

19 décembre 2014

1 Introduction

L'émergence des réseaux sociaux, que ce soit dans la sphère publique ou professionnelle, fait que l'étude des graphes les représentant est en expansion. Ce champ d'étude comporte de nombreuses thématiques et nous avons décidé de nous concentrer sur la détection de communautés. En effet cette problématique peut être généralisée pour tout graphe et ses applications pour les réseaux sociaux sont nombreuses.

Notre travail porte sur un réseau de collaborations académiques. En effet l'objectif est d'aider les chercheurs en physique des hautes énergies à trouver une communauté importante sur un sujet précis afin d'y rentrer et participer au réseau. Ou au contraire si le scientifique désire explorer un domaine étudié par peu de physiciens et développer ses travaux dessus, notre projet pourra lui être utile.

2 Données et méthode

2.1 Données : publications sur *arXiv*

Nos données étaient composées du résumé de tous les articles publiés sur la plate-forme de publications scientifiques *arXiv* en physique sur la théorie des hautes énergies. Ces données (téléchargeables ici) sont extraites de l'étude de J. Leskovec, J. Kleinberg and C. Faloutsos : *Graph Evolution: Densification and Shrinking Diameters* [1]. Nous n'avons pas utilisé le graphe des collaborations présent dans l'article car il était difficile de relier le label du chercheur à son identité (et donc ses publications) et nous avons alors recréé nos données directement à partir des résumés des articles.

Dans le but de repérer les communautés au sein des chercheurs nous nous sommes intéressés à l'étude du graphe de collaborations de ce domaine. Ce graphe est construit de la manière suivante : chaque chercheur est représenté par un nœud i et il y a une arête entre le nœud i et le nœud j si les chercheurs i et j ont déjà publié un article ensemble. Ainsi un article ayant été co-écrit par plusieurs chercheurs génère un sous-graphe complet c'est-à-dire que tout les nœuds sont reliés entre eux. De plus il est intéressant de noter que le graphe n'est pas orienté.

Notre graphe final est composé de 10134 nœuds et 82134 arêtes. Le fait que le degré moyen du graphe soit assez élevé, de l'ordre de 16 arêtes par nœud, peut facilement s'expliquer par le fait qu'un article est généralement co-publié par trois ou quatre chercheurs et que chacun d'eux a publié plusieurs articles.

Pour compléter les informations du graphe nous avons créé une table comprenant, pour chaque chercheur, tous les mots de plus de trois lettres présent dans les titres de ses articles. Ces mots servent à caractériser plus spécifiquement le champ d'étude de chaque chercheur et

permettra par la suite de donner une étiquette aux communautés de chercheurs créées.

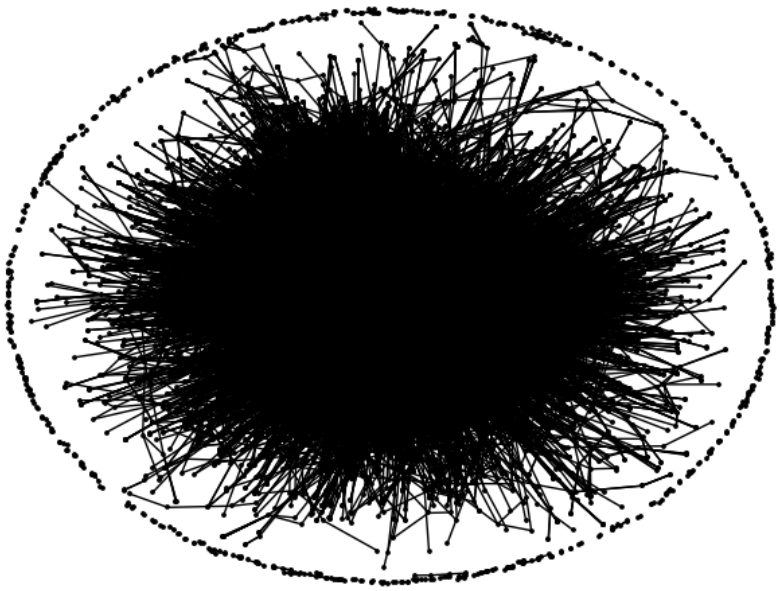

FIGURE 1 – Graphe initial

Le graphe 1 est assez peu lisible. Malgré ce fait, on s'aperçoit que le graphe n'est pas connexe : des chercheurs n'ont publié qu'un seul article.

Nous allons par la suite utiliser un algorithme permettant de détecter les communautés au sein de ce graphe.

2.2 Détection de communautés dans un graphe : méthode de Louvain

Détecter des communautés, dans un graphe dans notre cas, consiste à regrouper les nœuds les plus proches et ainsi à former des groupes de nœuds. Cependant on devine que l'on peut regrouper tout les nœuds en unique groupe et il faut alors déterminer à quel moment on peut arrêter l'agrégation pour obtenir des communautés stables. Un moyen de mesurer la bonne décomposition en groupes du graphes est la modularité, que l'on va chercher à maximiser

pour aboutir à la meilleure partition possible.

Optimisation de la modularité

La modularité peut être définie de plusieurs manières mais nous nous concentrons sur celle adoptée par la méthode de Louvain, université dont des chercheurs ont mis au point l'algorithme que nous utilisons. Le point important de la modularité est la considération de la répartition des poids dans un graphe non orienté par rapport à une distribution aléatoire. Elle ne consiste pas à simplement compter les arêtes mais étudie les arêtes et les degrés.
Elle est définie de la manière suivante :

$$M = \frac{1}{4m} \sum_{i,j} \big(A_{i,j} - \frac{k_i k_j}{2m}\big) \delta(c_i, c_j) \tag{1}$$

où $A_{i,j}$ est le poids de l'arête entre les nœuds i et j, $m = \frac{1}{2} \sum_{i,j} A_{i,j}$, $k_i = \sum_j A_{i,j}$ est la somme des poids des arêtes partant de i et c_i est la communauté à laquelle appartient i. Et $\delta(x, y) = \mathbf{1}_{\{x=y\}}$. Dans le cas d'un graphe sans poids (i.e. les poids sont tous égaux à un), k_i est le degré de i.

Si le graphe était généré aléatoirement, on devrait avoir $A_{i,j} = \frac{k_i k_j}{\sum_i k_i}$ (car $2m = \sum_{i,j} A_{i,j} = \sum_i k_i$) c'est-à-dire le poids moyen attendu entre i et j. La modularité prend ses valeurs entre -1 et 1.

La modularité sert à comparer deux partitions d'un graphe pour déterminer la meilleure et peut donc être utilisée pour déterminer le cluster idéal en maximisant la quantité. La figure 5 montre la partition d'un graphique qui maximise la modularité aboutissant à trois groupes (ou communautés).

Cependant cette maximisation n'est pas aisée, elle s'effectue sur de très nombreuses données et si l'on calcule la modularité sur toutes les partitions possibles, cela peut prendre un temps extrêmement long. Il existe donc des algorithmes qui aboutissent à un maximum local de la modularité en un temps acceptable, en l'augmentant progressivement d'une étape à l'autre. C'est donc un algorithme heuristique.

Méthode de Louvain

MM. Blondel, Guillaume, Lambiotte et Lefebvre [2] de l'université de Louvain ont développé à partir de 2007 un algorithme pour déterminer rapidement les communautés dans un graphe

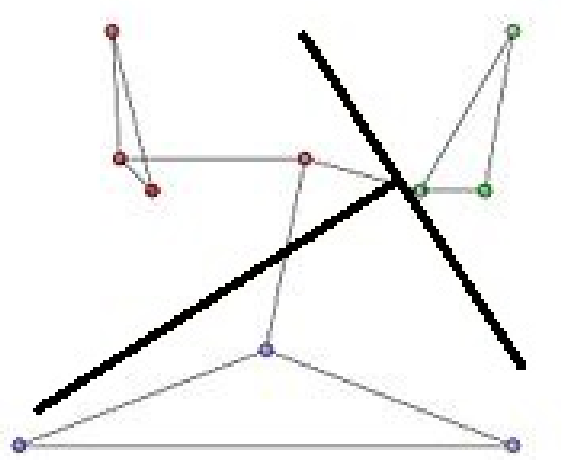

FIGURE 2 – Exemple de modularité maximale

en utilisant cette mesure de la structure d'un graphe.

Leur méthode comporte deux sections. La première assigne à chaque nœuds du graphe une communauté. Puis pour chaque nœuds i on étudie le gain de modularité à ajouter i à la communauté de ses nœuds voisins. On place ensuite i dans la communauté pour laquelle le gain est positif et le plus élevé. Si le gain est négatif, i reste dans sa communauté. On itère cette opération jusqu'à ce que la modularité n'augmente plus, quitte à considérer plusieurs fois le même nœud, on atteint alors un maximum local. Il est important de noter que l'ordre dont on traite les nœuds peut faire changer le maximum de la modularité. C'est d'ailleurs une caractéristique que l'on va étudier pour tester la stabilité des communautés.

Dans un second temps, on part de la partition obtenue à l'étape précédente et on considère qu'une communauté consiste en un nœud avec comme poids la somme des poids des nœuds initiaux agrégés. Face à ce nouveau graphe, on exécute à nouveau l'étape une. Et ainsi de suite on obtient de communautés de plus en plus grosses et de moins en moins nombreuses. À un moment la modularité n'augmente plus et on a finalement détecter la partition optimale. L'idée est de faire des communautés de communautés si l'agrégat est assez "fort" au sens de la mesure choisie.

3 Analyse des résultats

Une fois les communautés repérées grâce à l'algorithme décrit précédemment, il a fallu représenter et analyser la pertinence des résultats obtenus. Nous commencerons par étudier les statistiques qui nous permettent de caractériser un ensemble de communautés pour ensuite nous intéresser à la stabilité de l'algorithme mis en place.

Représentation graphique et statistiques

Lorsque l'on a appliqué l'algorithme à nos données on a obtenu une subdivision du graphe en 269 classes. La plupart de ces communautés ne regroupe que deux éléments. Pour caractériser ces classes nous avons donné à chacune d'entre elles une étiquette correspondant aux trois mots les plus fréquents dans les titres des auteurs qui composent la communauté. Cette étiquette correspond en quelque sorte au domaine de recherche dominant du groupe.

On peut alors résumer cette partition du graphe sous forme d'un graphe pondéré où chaque nœud est une communauté auquel on attribue un poids correspondant au nombre d'auteurs la composant. Les nœeuds i et j sont reliés s'il existe un auteur de la communauté i qui a publié un article avec un auteur de la communauté j. Le poids affecté à l'arête est alors le nombre de d'articles co-publiés par des chercheurs des communautés i et j. Ainsi plus le poids d'une arête est élevé plus le lien entre les deux communautés est fort. On note également que l'on a créé aussi des boucles sur chaque nœud c'est-à-dire une arête qui relie le sommet i à lui-même dont le poids représente le nombre d'arêtes dans le sous-graphe de la communauté.

Nous avons ensuite voulu représenter le graphe obtenu, cependant du aux grands nombres de nœuds (269) et d'arêtes, nous avons supprimé tous les noeuds de moins de 100 individus et les arêtes représentant moins de 60 collaborations. La suppression des noeuds n'a au final retiré que les communautés composées de deux ou trois auteurs ayant publié ensemble un seul article.

Le graphe obtenu en ne gardant que les 29 classes significatives se trouve sur lapage suivante

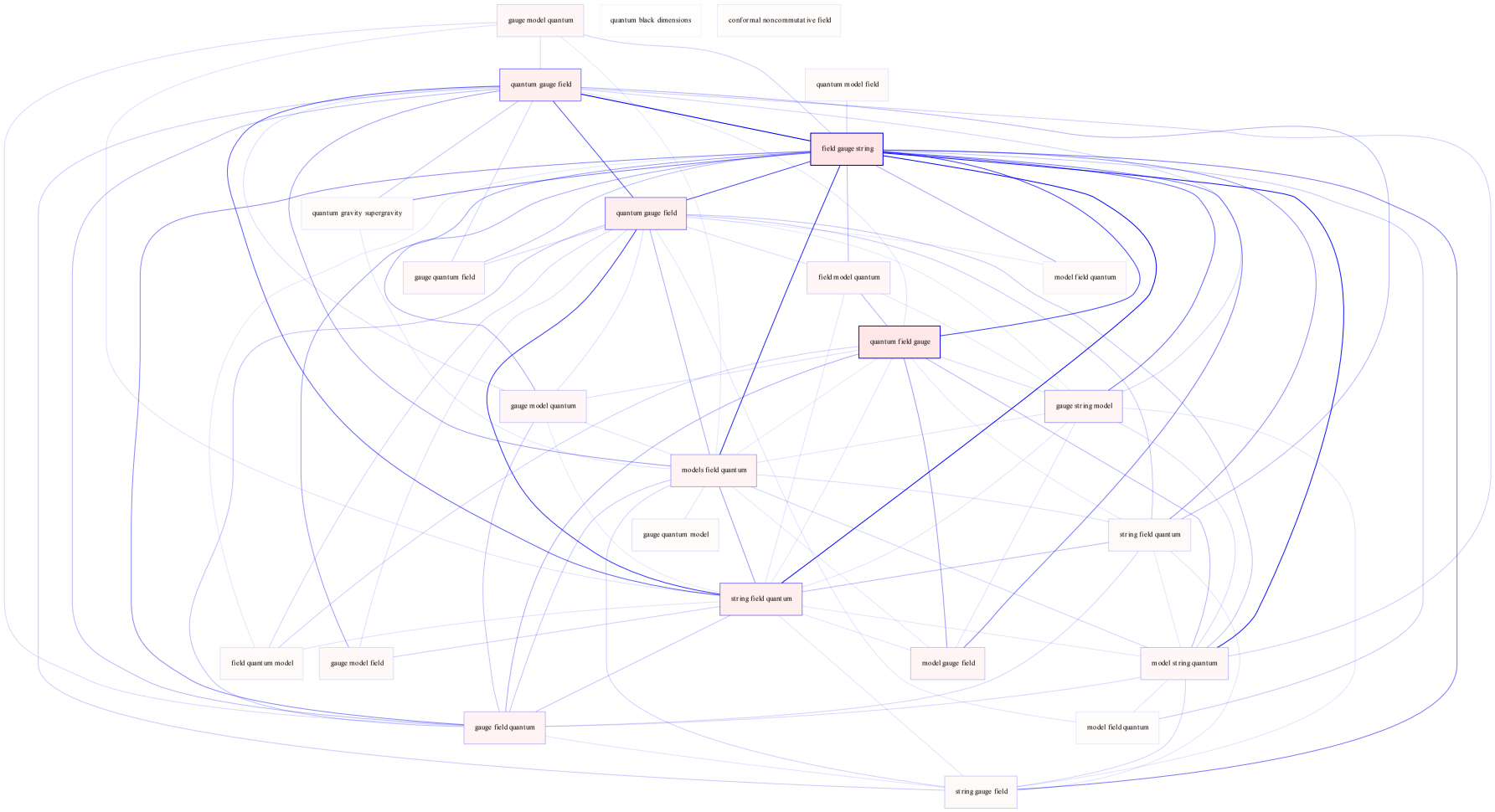

FIGURE 3 – Graphe des communautés

L'opacité est proportionnelle au poids, pour un nœud cela correspond au nombre d'auteurs qui le composent et pour une arête au nombre de co-publications. La bordure d'un nœud représente le nombre de co-publications au sein de la classe.

La représentation précédente permet de voir rapidement les pôles importants grâce au jeu de couleurs. Les teintes rouges plus foncées marquent des sujets sur lesquels de nombreux chercheurs ont travaillé et les arêtes plus opaques les liens importants entre certain champ de la physique des hautes énergies. On peut cependant regretter que de nombreuses étiquettes soient trop semblables pour différencier réellement les champs d'études.

Pour l'étude de la qualité de la détection de communautés, il est intéressant de regarder la répartition des nœuds et autres –de base. Nous avons considéré ainsi chaque groupe étiqueté par trois mots-clefs qui est en fait un sous-graphe et calculé le nombre de nœuds, de degrés etc. Puis à cette base de statistiques, une description fournissant des moyennes, écart-types et autres a été effectuée pour une meilleure analyse du graphe final. Le tableau 1 résume cela.

Pour préciser, le graphe initial étant non-orienté et sans poids, le nombre de degrés par cluster est le double de celui des arêtes. L'écart-type de toutes les mesures est assez important par rapport aux moyennes ce qui laisse à penser que les sous-graphes sont plutôt hétérogènes. Les quartiles et le graphique 4 en log-log confirment l'inégale répartition des nœuds dans chaque communauté : il existe de nombreux petits groupes et peu de larges classes.

	Number of nodes	Number of edges	Sum of degrees	Mean of degrees
mean	369.880	1135.560	2271.120	5.519
std	238.155	1066.508	2133.015	1.211
min	101	241	482	4.231
25%	208	459	918	4.784
50%	301	714	1428	5.111
75%	449	1219	2438	5.938
max	1000	4262	8524	8.907

TABLE 1 – Statistiques descriptives

Le code qui génère ce tableau est basé sur la fonction suivante qui renvoie les informations par cluster.

```
def stat(g,ind=None,plot=False):
    df_stat=pd.DataFrame(columns=["Number of nodes","Number of edges",
    "Sum of degrees","Mean of degrees"])
    df_stat["Number of nodes"]=[g.number_of_nodes()]
    df_stat["Number of edges"]=[g.number_of_edges()]
```

```
df_stat["Sum of degrees"]=[np.sum(g.degree().values())]
df_stat["Mean of degrees"]=[np.mean(g.degree().values())]

if ind!=None:
    df_stat.index=[ind]
return df_stat
```

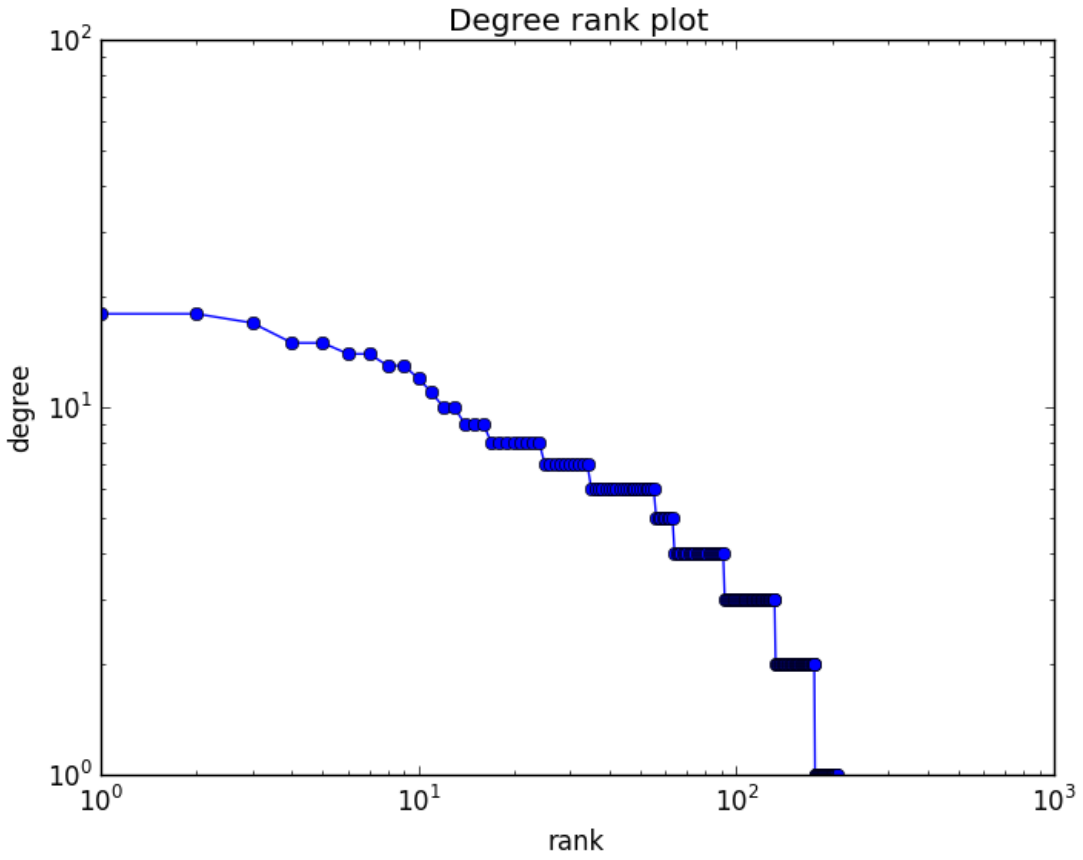

FIGURE 4 – Représentation log-log de la répartition des nœuds dans les communautés

Étude de la stabilité.

Comme énoncé dans la partie précédente, le seul paramètre jouant sur le résultat final est l'ordre de parcours des nœuds par l'algorithme. Afin de modifier cet ordre nous avons essayé de construire le graphe de différentes manières pour tester la stabilité de la méthode. Nous avons commencé par présenter les sommets dans l'ordre inverse du premier graphe mais cela

nous a donné exactement la même subdivision.

On a alors mélangé de manière aléatoire les sommets et dans ce cas on a observé une légère différence dans les classes obtenues en sortie. Les différences sont minimes et sont essentiellement situées dans les petites classes comme on peut le voir sur le graphe 5. Les communautés les plus grosses restent insensibles à cette perturbation.

En étudiant la modularité des deux partitions, on s'aperçoit que le premier graphe obtient un meilleur score avec 0.64916 alors que la permutation aléatoire donne 0.64799. La différence est minime, de l'ordre de 10^{-3}, indique donc bien que la méthode de Louvain est plutôt stable dans sa détection de communautés.

Par conséquent un physicien peut utiliser cette méthode car elle s'avère satisfaisante sur le temps d'exécution et sur la stabilité des groupes obtenus. Le scientifique pourra donc déterminer le sujet à opter pour être présent dans une grande communauté.

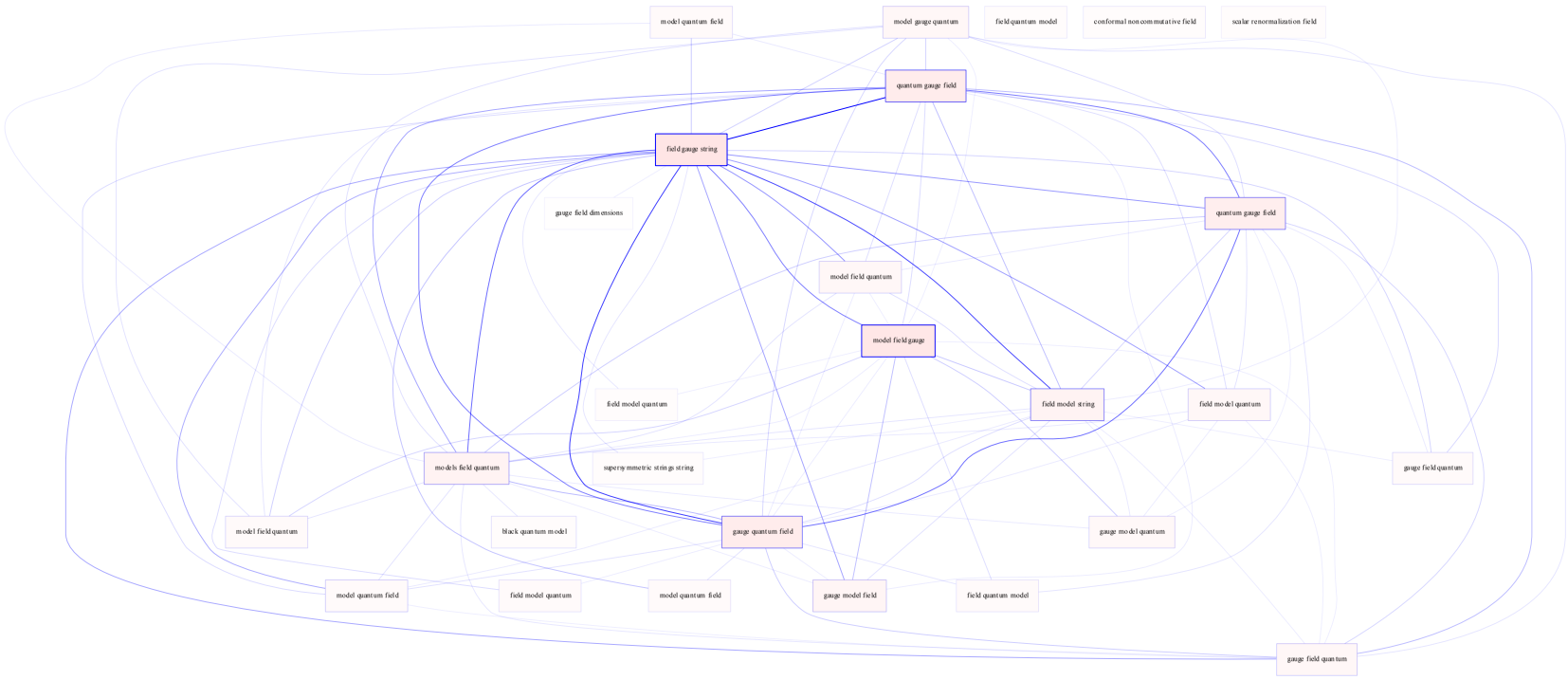

FIGURE 5 – Graphe des communautés pour les nœuds proposés aléatoirement

4 Limites et ouvertures

Notre travail présente néanmoins des limites sur deux échelles. La première concerne des données. Notre base de données ne donnant pas le lien entre les articles et leur numéro associé, nous avons du la créer manuellement en liant les articles et les auteurs. Mais comme les sélections faites sur le noms des auteurs ne suivent pas un format identique, nous nous sommes retrouvés avec des prénoms dans la liste ou des noms de facultés qui perturbent la pertinence des résultats obtenus.

L'autre niveau qui limite notre étude est l'algorithme. Il n'est pas modulable à souhait et se contente de renvoyer la meilleure solution en terme de modularité. On ne peut pas choisir le nombre de communautés par exemple. De plus cette méthode peut ne pas considérer une petite classe en tant que telle et donc la rattacher de force (par l'optimisation de la modularité) à un groupe plus important. De plus, tout repose sur la notion de modularité qui peut être en concurrence avec d'autres outils pour mesurer la qualité de représentation des communautés d'un graphe.

À l'opposé, l'algorithme est performant face à ces compétiteurs en terme de temps d'exécution et de modularité finale obtenue. Malgré le fait que nous n'ayons pas étudié cela par nous-mêmes, l'efficacité est avéré.

Une voie potentielle pour compléter notre étude est de s'intéresser à l'évolution temporelle des groupes et des sujets de recherche en physique des hautes énergies. Nous aurions pu alors déterminer si les mêmes thèmes étaient récurrents ou si de nouveaux apparaissaient au fil du temps.

Références

[1] J. Kleinberg J. Leskovec and C. Faloutso. Fast unfolding of communities in large networks. 2007.

[2] Renaud Lambiotte Vincent D. Blondel, Jean-Loup Guillaume and Etienne Lefebvre. Fast unfolding of communities in large networks. 2008.

Packages utilisés :

— NetworkX : pour manipuler des graphes.

— Community : pour la méthode de Louvain, le package est téléchargeable ici