Utilisation du logiciel Galaxies

# Introduction

Le logiciel galaxies est destiné à traiter les réutilisations obtenues avec des logiciels comme Phœbus et Philoline. Ces deux derniers logiciels ont été fusionnés dans une nouvelle version, *text-align*, qui est accessible sur Github à l’URL suivante : <https://github.com/ARTFL-Project/text-align>

On y trouve toutes les instructions pour installer et lancer le programme.

* Voici un corpus TEI qui vous permettra de tester le programme text-align (il s’agit de la comédie humaine de Balzac):
  + <http://condorcet.uchicago.edu/clovis/balzac.tar>

Les sorties du programme *text-align* se présente sous la forme d’un fichier qui comprend des suites d’objets au format JSON.

Voici des liens vers trois fichiers de ce type :

* + Encyclopédie contre TGB : <http://anomander.uchicago.edu/clovis/encyc_tgb.tab>
  + Frantext contre TGB: <http://anomander.uchicago.edu/clovis/frantext_tgb.tab>
  + La Comédie humaine de Balzac contre elle-même : https://cloud.lip6.fr/index.php/s/ZSgagDfalS402KB

# Programme Galaxies

**Le programme Galaxies** peut être téléchargé à l’URL suivante : https://cloud.lip6.fr/index.php/s/nFdQeAdnkppNQAC

**Le programme Louvain** peut lui être téléchargé à l’URL :

https://cloud.lip6.fr/index.php/s/OOwvXDB1gaoTSkU

Il faut placer ce dernier dans le dossier *DirLouvain* qui peut se trouver où l’on veut.

## Paramètres

**Avant de lancer le programme**, il faut aller dans le fichier *parametres.py*

Il y a là deux paramètres à définir absolument : *DirFichier* et *DirLouvain*.

### DirFichier

DirFichier=**'/Users/jean-gabriel/galaxies/ARTFL-Balzac3/'**

Chemin vers le dossier qui comprend le fichier des sorties du programme *text-align* qui sera l’entrée du programme Galaxies

Le programme crée tout seul trois dossiers : BDs, amas et graphes qui sont repérés par les trois variables DirBD, DirAmas et DirGraphes

A priori, il ne faut pas y toucher, sauf dans des cas particuliers.

### DirBD

DirBD = **'/Users/jean-gabriel/galaxies/ARTFL-Balzac3/BDs'**

Chemin vers le dossier qui comprend les résultats du programme Galaxies

### DirAmas

DirAmas = **'/Users/jean-gabriel/galaxies/ARTFL-Balzac3/amas'**

Chemin vers le dossier qui comprend les résultats du programme Louvain qui détecte les communautés

### DirGraphes

DirGraphes = **'/Users/jean-gabriel/galaxies/ARTFL-Balzac3/graphes'**

Chemin vers le dossier qui comprend les graphes au format « .gexf » générés par les différentes fonctions de sauvegarde des galaxies ou des amas.

### DirLouvain

DirLouvain = **'/Users/jean-gabriel/ownCloud/Projets/Use and Reuse/Graphe réutilisations/GalaxiesARTFL\_mai2018/gen-louvain'**

Chemin vers le dossier qui comprend les le programme Louvain

### Métadonnées

Il existe quatre paramètres qui sont utilisés pour interroger la base de galaxies (voir plus loin la section *Langage de requêtes*). Ils définissent deux champs des objets JSON qui sortent du programme text-align qui feront l’objet d’interrogation et leurs types respectifs. Ces paramètres sont : MetaDataSource et MetaDataCible. Par exemple, pour interroger l’encyclopédie, on a décidé de prendre :

* metaDataSource = **'source\_generatedclass'**
* metaDataSourceType = **'TEXT'**
* metaDataCible = **'target\_birth'**
* metaDataCibleType = **'NUM'**

### Grosses galaxies

Le paramètre *tailleMinimaleGrosseGalaxies* indique la taille au dessus de laquelle les galaxies sont considérées comme trop grosses pour être visualisées. Dans ce cas de figure, on fait appel à Louvain pour détecter les communautés.

## Lancement du programme Galaxies

Pour lancer le programme Galaxies sur le fichier stocké dans le dossier DirFichier, il suffit de faire : lancement\_programme()

Cela peut prendre un peu de temps. Deux fichiers sont générés dans le dossier DirBD :

* galaxies.db (c’est une base de données squlite qui contient toutes les sorties du programme)
* listGalaxies.db : c’est un fichier shelves utilisé par python.

## Exploration des galaxies

Avant d’explorer les galaxies, il est bon d’ouvrir la base de données galaxies.db après avoir installé un logiciel d’exploration de sqlite. Il en existe beaucoup. On peut même installer un plugin sur firefox pour cela. En allant sur la table degreGalaxies on a la liste des galaxies avec le degré de chacune (c’est-à-dire le nombre de nœuds qu’elle contient) d’entre elle, la longueur totale des textes et leur longueur moyenne.

Pour explorer les galaxies générées par le programme, il existe différentes fonctions :

Les plus commode sont *impressionTexteEtReferenceLongueur(N°Galaxie,P)* et *impressionTexteLongueur(n, p)* (sans les références)

* *N°Galaxie* est le numéro de la galaxie dont on veut imprimer les composantes
* *P* limite le nombre de composantes imprimées. Si *P*=0, elles sont toutes imprimées.

Longueur veut dire qu’elles sont ordonnées par longueur et que l’on imprime les plus longues d’abord.

Il existe une autre fonction qui ordonne par ancienneté : *impressionTexteEtReferenceAnciennete(n, p)*

Enfin, il en existe qui n’ordonnent pas : *impressionTexteEtReference(n, p)* et *impressionTexte(n, p)* (sans les références)

## Visualisation du graphe des galaxies

Il est possible d’exporter un fichier au format GEXF avec l’extension « .gexf » pour chaque galaxie. Ce fichier qui est généré dans le dossieur DirBD peut ensuite s’ouvrir avec Gephi ou avec d’autres logiciels de visualisation de graphe comme Cytoscape.js (cf. [js.cytoscape.org](http://js.cytoscape.org/)).

Pour procéder à cette exportation, il suffit d’appeler la fonction *sauveGrapheGalaxie(N°Galaxie)* où *N°Galaxie* est le numéro de la galaxie dont ou veut exporter le graphe. Plus exactement, comme cette fonction se trouve dans le module extractionGalaxies, il faut appeler *extractionGalaxie.sauveGrapheGalaxie(N°Galaxie)*

Le fichier généré s’appelle *graphe\_galaxie\_N°Galaxie.gexf* ; on peut l’ouvrir avec Gephi

## Génération des amas

Lorsque les galaxies sont trop grosses (plus de 300 nœuds), on fait appel à l’algorithme Louvain qui regroupe les nœuds qui ont beaucoup de liens entre eux. Cet algorithme correspond à ce que l’on appelle la recherche de communautés. Pour l’appeler, il suffit de créer un objet A :

*A = amas.constructionAmas(N°Galaxie)*

Il existe une fonction qui permet d’imprimer les différents groupes de la galaxie A. Pour l’appeler, il suffit de faire : *A.impression\_groupes\_degre()*

On peut aussi sauvegarder ces groupes dans un fichier GEXF pour les visualiser avec Gephi.

# Langage de requête

Il arrive que l’on veuille se centrer sur des galaxies qui ont des propriétés. Pour cela, on a défini un langage d’interrogation de la base de Galaxies.

On filtre les galaxies avec la fonction : *extractionGalaxies.galaxiesFiltre(requete)* où la requête est une dictionnaire python dont les clefs sont les suivantes :

* **‘auteur’**: il faut donner une *liste* qui comprend les noms et prénoms qui devront tous être présent sur au moins un nœud dans la galaxie
* **‘-auteur’**: il faut donner une *liste* les noms et prénoms n’apparaitront pas sur au moins un nœud dans la galaxie
* **‘date’**: c’est un *couple* qui définit la borne inférieurs et la borne supérieure de la valeur de la date.
* **‘mots\_titre’**: liste de mot qui doivent appartenir au titre d’un ouvrage présent dans l’une des réutilisations de la galaxie
* **‘-mots\_titre’**: liste de mot dont aucun n’appartient au titre d’un ouvrage présent dans l’une des réutilisations de la galaxie
* **‘empan’**: il faut donner un *nombre* qui correspond à la longueur minimal du texte du nœud sur lequel porte la requête
* **‘longueur\_texte\_maximal’** : il faut donner un *nombre* qui correspond à la longueur minimal du texte le plus long des nœud de la galaxie
* **‘nbre\_minimal\_noeuds’**: c’est le *nombre* minimal de nœuds que comprend la galaxie pour être considérée

À ces différentes clefs, on en ajoute deux qui sont définies au grès de l’utilisateur dans les paramètres et qui correspondent à metaDataSource et à metaDataCible. L’interrogation dépend du type qui peut être soit un nombre (type ‘NUM’), auquel cas on donne un couple [borne inférieure, borne supérieure] ou un texte (type ‘TEXT’) auquel cas on teste l’inclusion de la chaîne de caractères dans la métadonnée correspondante.

Dans le cas de l’encyclopédie, voici les deux métadonnées que l’on a rajoutées :

* metaDataSource = **‘source\_generatedclass’**
* metaDataSourceType = **‘TEXT’**
* metaDataCible = **‘target\_birth’**
* metaDataCibleType = **‘NUM’**

Voici quelques exemples de requêtes sur l’encyclopédie :

requete = {**'auteur'**:[**'d\'Holbach'**], **'empan'**:100, **'date'**:[1800,**'-'**], **'nbre\_minimal\_noeuds'**:4, **'longueur\_texte\_maximal'**:500}

requete = {**'auteur'**:[**'Jean-Jacques'**, **'Rousseau'**], **'empan'**:100, **'date'**:[1800,**'-'**], **'nbre\_minimal\_noeuds'**:4, **'longueur\_texte\_maximal'**:500}

requete = {**'auteur'**:[**'Jean-Jacques'**, **'Rousseau'**], **'mots\_titre'**: [**'Contrat'**, **'Social'**], **'empan'**:100, **'date'**:[1800,**'-'**], **'nbre\_minimal\_noeuds'**:4, **'longueur\_texte\_maximal'**:500}

requete = {**'auteur'**:[**'d\'Holbach'**, **'Diderot'**], **'empan'**:100, **'date'**:[1800,**'-'**], **'nbre\_minimal\_noeuds'**:4, **'longueur\_texte\_maximal'**:500}

requete = {**'source\_generatedclass'**:**'Jurisprudence'**, **'empan'**:100, **'date'**:[1800,**'-'**], **'nbre\_minimal\_noeuds'**:4, **'longueur\_texte\_maximal'**:500}

requete = {**'source\_generatedclass'**:**'Morale'**, **'empan'**:100, **'date'**:[1800,**'-'**], **'nbre\_minimal\_noeuds'**:4, **'longueur\_texte\_maximal'**:500}

requete = {**'source\_generatedclass'**:**'Belles-Lettres'**, **'empan'**:100, **'date'**:[1800,**'-'**], **'nbre\_minimal\_noeuds'**:4, **'longueur\_texte\_maximal'**:500}

requete = {**'source\_generatedclass'**:**'Métaphysique'**, **'empan'**:100, **'date'**:[1800,**'-'**], **'nbre\_minimal\_noeuds'**:4, **'longueur\_texte\_maximal'**:500}

requete = {**'source\_generatedclass'**:**'Littérature'**, **'empan'**:100, **'date'**:[1800,**'-'**], **'target\_birth'**:[1765,**'-'**], **'nbre\_minimal\_noeuds'**:4, **'longueur\_texte\_maximal'**:500}

requete = {**'source\_generatedclass'**:**'Mythologie'**, **'empan'**:100, **'date'**:[1800,**'-'**], **'target\_birth'**:[1900,**'-'**], **'nbre\_minimal\_noeuds'**:4, **'longueur\_texte\_maximal'**:500}

On a rajouté deux autres fonctions qui filtrent sur des listes de requêtes : *galaxieFiltreListe(Lrequete, numero)* et *galaxiesFiltreListe(Lrequete)*

Enfin, il existe des possibilités de filtrage des graphes dans l’affichage avec une requête. Dans un premier temps, cela correspond à l’une des requêtes de la liste. Pour cela, on appelle la fonction *extractionGalaxies.galaxiesFiltreListeAffiche(Lrequetes)* qui filtre l’ensemble des galaxies sur la liste de requêtes *Lrequetes* et qui ensuite propose de filtrer l’affichage sur l’une des requêtes de la liste *Lrequetes*.

## Génération et visualisation des amas

Lorsque les galaxies sont trop grosses (plus de 300 nœuds), on fait appel à l’algorithme Louvain qui regroupe les nœuds qui ont beaucoup de liens entre eux. Cet algorithme correspond à ce que l’on appelle la recherche de communautés. Pour l’appeler, il suffit de créer un objet A :

*A = amas.constructionAmas(N°Galaxie)*

Il existe une fonction qui permet d’imprimer les différents groupes de la galaxie A. Pour l’appeler, il suffit de faire : *A.impression\_groupes\_degre()*

On peut aussi sauvegarder ces groupes dans un fichier GEXF pour les visualiser avec Gephi.

De même que l’on filtre les galaxies avec des requêtes, on peut filtre les amas que l’on veut regarder avec des requêtes. Pour cela, il suffit de faire appel à la méthode *filtrageAmas* avec la requête comme argument. A titre d’illustration, si l’on souhaite filtrer et sauvegarder les amas de la galaxie dont le numéro est N°Galaxie, il suffit de faire :

*A=amas.constructionAmas(N°Galaxie)  
A.filtrageAmas(requete)*

Il est aussi possible d’utiliser une liste de requêtes avec la fonction *filtrageAmasListe*. Cela donne :

*A.filtrageAmasListe(Lrequete)*

Enfin, on peut aussi filtrer les graphes à l’affichage. Cela signifie que l’on cherche les amas qui satisfont une liste de requêtes, à savoir qui comprennent un lien caractérisé par chacune des requêtes, puis qu’ensuite, on enlève certains nœuds qui ne satisfont pas l’une des requêtes de la liste. Pour cela, on appelle la fonction *filtrageAffichageAmasListe*. Pour des raisons de commodité, à savoir pour ne pas avoir à saisir une requête à l’écran, elle demande de choisir l’une des requêtes de la liste et l’applique pour filtrer l’affichage :

*A.filtrageAffichageAmasListe(Lrequete)*