

1 TD 6 : Graphe minuté

(correction page ??)

Abordé lors de cette séance	
programmation	classes
algorithme	algorithme sur les graphes

Les graphes¹ sont un peu partout sans qu'on s'en rende compte car ils sont une façon simple et visuelle de représenter des relations. Un réseau social est un graphe, l'illustration 1 est un graphe.

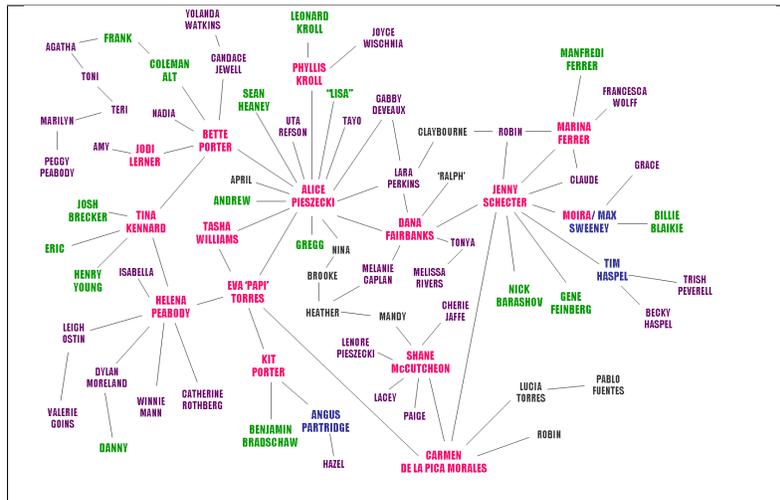


FIGURE 1 : Image extraite de http://fr.wikipedia.org/wiki/The_L_Word.

Première demi-heure : construire un graphe

On considère la liste des relations décrite comme suit² :

```
colemans,frank
dana,lara
dana,melanie
dana,ralph
danny,dylan
dona,jenny
...
```

Vous pouvez le récupérer directement en utilisant la fonction suivante :

```
def lit_donnees_depuis_un_fichier_ou_internet (fichier) :
    """
```

1. http://fr.wikipedia.org/wiki/Th%C3%A9orie_des_graphes
2. La liste complète est disponible à l'adresse suivante : http://www.xavierdupre.fr/enseignement/td_python/python_td_minute/data/lworld/pairs.txt.

```

cette fonction lit un fichier où le télécharge depuis un site internet,
si le fichier n'existe pas, il est téléchargé puis enregistré sur disque
"""
if not os.path.exists (fichier) :
    url = "http://www.xavierdupre.fr/enseignement/complements/" + fichier
    import urllib2
    f = urllib2.urlopen (url)
    t = f.read()
    f.close()
    f = open(fichier, "w")
    f.write(t)
    f.close()

```

1) Le programme suivant est à compléter (...), il permet de récupérer une matrice à partir du fichier téléchargé :

```

f = open (".....", "r")
lines = f.readlines ()
f.close ()

# la partie après le if sert à enlever les lignes vides
couples = [ l.strip("\n\r").split ( ... ) for l in lines if ... ]

```

2) Comment compter le nombre de prénoms différents ?

3) On souhaite compter le nombre de relations directes pour chaque individu, que suggérez-vous en vous inspirant de la question précédente ?

Seconde demi-heure : les amis

Si on considère que chaque arc relie deux amis, on souhaite connaître pour chaque personne combien elle a d'amis d'amis. C'est l'objectif des questions suivantes.

4) Si vous avez deux amis qui ont chacun 50 amis, avez-vous nécessairement 100 amis d'amis ?

5) On considère la classe suivante :

```

class Personne :
    def __init__ (self, nom):
        self.nom = nom
        self.amis = []

    def ajout_ami (self, ami) :
        self.amis.append (ami)

```

Ajouter une méthode qui retourne le nombre d'amis.

6) On récupère la liste des prénoms distincts dans un dictionnaire :

```

graph = { }
for prenom in liste_prenoms :
    graph [ prenom ] = Personne ( prenom )

```

Complétez le programme suivant (`couples` est la liste créée à la première question) :

```
for a,b in couples :  
    graph [ ... ].ajout_ami ( ... )  
    graph [ ... ].ajout_ami ( ... )
```

Pourquoi faut-il ajouter la seconde ligne ? Que se passe-t-il si le fichier contient une relation dans un sens puis dans l'autre ?

Troisième demi-heure : les amis d'amis

7) Ecrire une méthode qui compte les amis d'amis :

```
class Personne :  
    ...  
    def nombre_amis_amis(self) :  
        nb = 0  
        ....  
        return nb
```

8) Créer une seconde fonction pour calculer les amis d'amis d'amis ?

9) Pourquoi le terme d'amis d'amis d'amis n'est-il pas très exact ?

Quatrième demi-heure : composantes connexes

10) On propose de créer une fonction qui calcule la somme de tous les amis, amis d'amis, amis d'amis... La figure 2³ représente le graphe déduit des relations présentes dans le fichier de la première question.

Expliquer pourquoi, quelque soit la personne choisie, il n'y a que deux réponses possibles.

11) Implémenter la fonction qui retourne le nombre total d'amis ?

12) Vérifiez qu'il n'y a que deux réponses différentes quelque soit la personne.

Pour aller plus loin ou pour ceux qui ont fini plus tôt

13) Comptez le nombre de triplets d'amis tous connectés entre eux ?

14) Une personne lance une rumeur. En un jour, il propage la rumeur à tous ses amis. En deux jours, tous ses amis d'amis la connaissent et ainsi de suite les jours suivant. Dans le pire des cas, combien de jours faut-il pour qu'une rumeur se propage dans la plus grande composante connexe ?

La réponse à cette dernière question n'est pas évidente sauf si on connaît déjà certains algorithmes comme celui de Bellman-Ford mais il est possible dans un premier temps de commencer à y réfléchir seul.

3. Ce graphe a été réalisé grâce à l'outil *Graphviz* disponible à <http://www.graphviz.org/>.

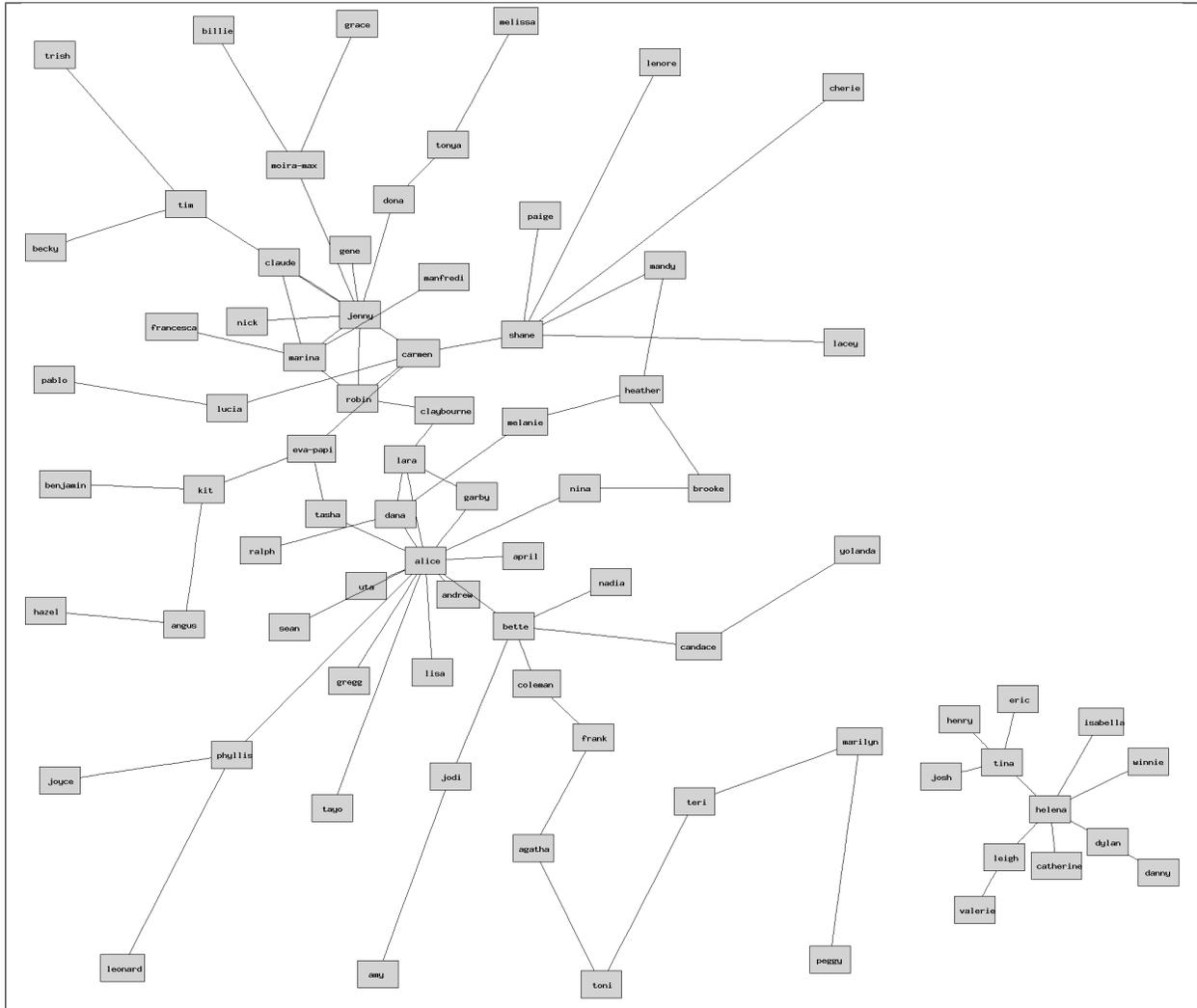


FIGURE 2 : Image qui représente le graphe décrit par les relations présentes dans le fichier téléchargé à la question 1.