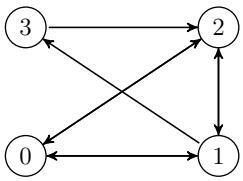
TP PageRank

M. Tami, T. Thonet,
E. Gaussier, (I. Partalas)

Le but de ce TP est d'étudier l'algorithme d'ordonnancement PageRank qui est utilisé pour la recherche d'information sur le Web.

Format des Données. Nous allons utiliser le format suivant pour représenter N pages Web :

```
4 % N web pages
0 1 2 % les liens
1 0 3 2
2 0 1
3 2
```



Pour chaque ligne (sauf la première) le premier numéro définit une page et les numéros suivants les liens sortants de cette page.

Matrice de transition. Écrivez un programme en Python qui calcule la matrice de transition à partir d'un graphe au format exposé ci-dessus. La matrice de transition P est définie par :

$$P_{ij} = \begin{cases} \lambda \frac{A_{ij}}{\sum_{j=1}^{N} A_{ij}} + (1-\lambda)\frac{1}{N} & \text{si } \sum_{j=1}^{N} A_{ij} \neq 0 \\ \frac{1}{N} & \text{sinon} \end{cases}$$

où $A_{ij} = 1$ s'il existe un lien de la page i vers la page j, et 0 sinon.

Vous trouverez des graphes de taille variable sous :

- `http://ama.liglab.fr/~gaussier/Courses/MastereBigData/TP-pagerank/graphe-samples/large-graph.txt`
- `http://ama.liglab.fr/~gaussier/Courses/MastereBigData/TP-pagerank/graphe-samples/medium-graph.txt`
- `http://ama.liglab.fr/~gaussier/Courses/MastereBigData/TP-pagerank/graphe-samples/small-20-pages.txt`
- `http://ama.liglab.fr/~gaussier/Courses/MastereBigData/TP-pagerank/graphe-samples/tiny-graph.txt`

Entrée :
- la matrice d'adjacence A du graphe dirigé ;
- λ, facteur d'amortissement ;
- ϵ, précision pour la critère d'arrêt ;

Initialisation :
- calculer la matrice de probabilité P ;
- vecteur PageRank $R^{(0)} = \left(\frac{1}{N}, \ldots, \frac{1}{N}\right)$;
- $l \leftarrow 0$;

répéter
- $R^{(l+1)} = R^{(l)}P$;
- $l \leftarrow l + 1$;

jusqu'à $||R^{(l+1)} - R^{(l)}|| \leq \epsilon$;

Sortie : vecteur PageRank $R^{(l)}$

Algorithme 1 : Algorithme de PageRank

Algorithme de PageRank. Fixez tout d'abord λ à 0.85 et implantez en Python la méthode des puissances (Algorithme 1).

Questions :

1. Choisissez différentes valeurs pour le critère d'arrêt (e.g. $\epsilon = \{10^{-3}, 10^{-4}, 10^{-5}\}$). Qu'est-ce que vous observez ?
2. Ajouter quelques *hubs* (pages qui ont beaucoup de liens sortant) et *autorités* (pages qui ont beaucoup de liens entrant). Quelles pages sont classées le plus haut ?
3. Essayez d'accroître les rangs de certaines pages. Expliquez votre méthode et validez-la expérimentalement.
4. Essayez différentes valeurs pour le facteur d'amortissement λ. Quel est le comportement de l'algorithme lorsque λ tend vers 0 ?

Complément : génération aléatoire de graphes. Écrivez un programme en Python qui permette de générer aléatoirement un graphe au format vu précédemment et avec un nombre N de pages (c'est-à-dire de nœuds) indiqué en paramètre. Le graphe généré sera sauvegardé dans un fichier dont le nom est également passé en paramètre du programme. Testez ensuite PageRank sur les graphes ainsi créés.