# Analyse de graphes

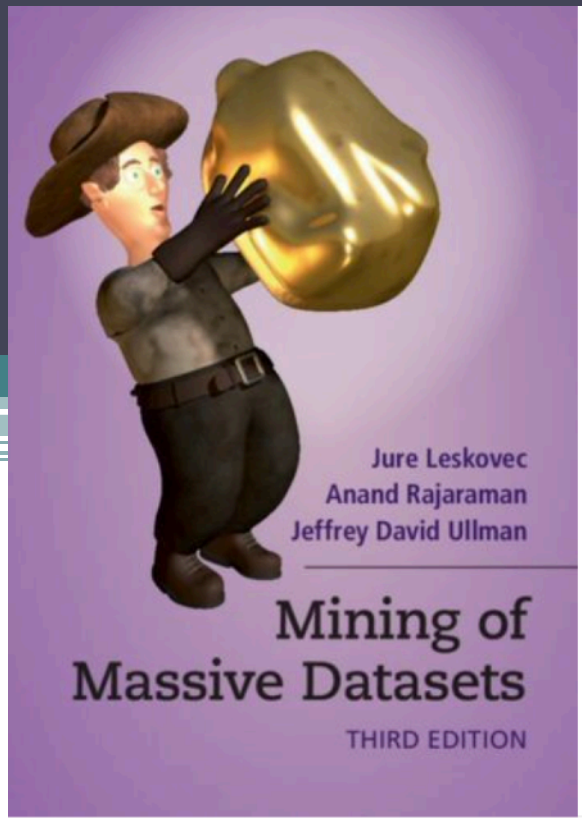
Dario Colazzo

# Plan

- Link analysis
- Graph database management - Neo4J
- Link prediction

# Fouille des liens web

Basé sur le chapitre 5 du livre MMDS, et sur du materiel de Richard Khoury.



# Outline

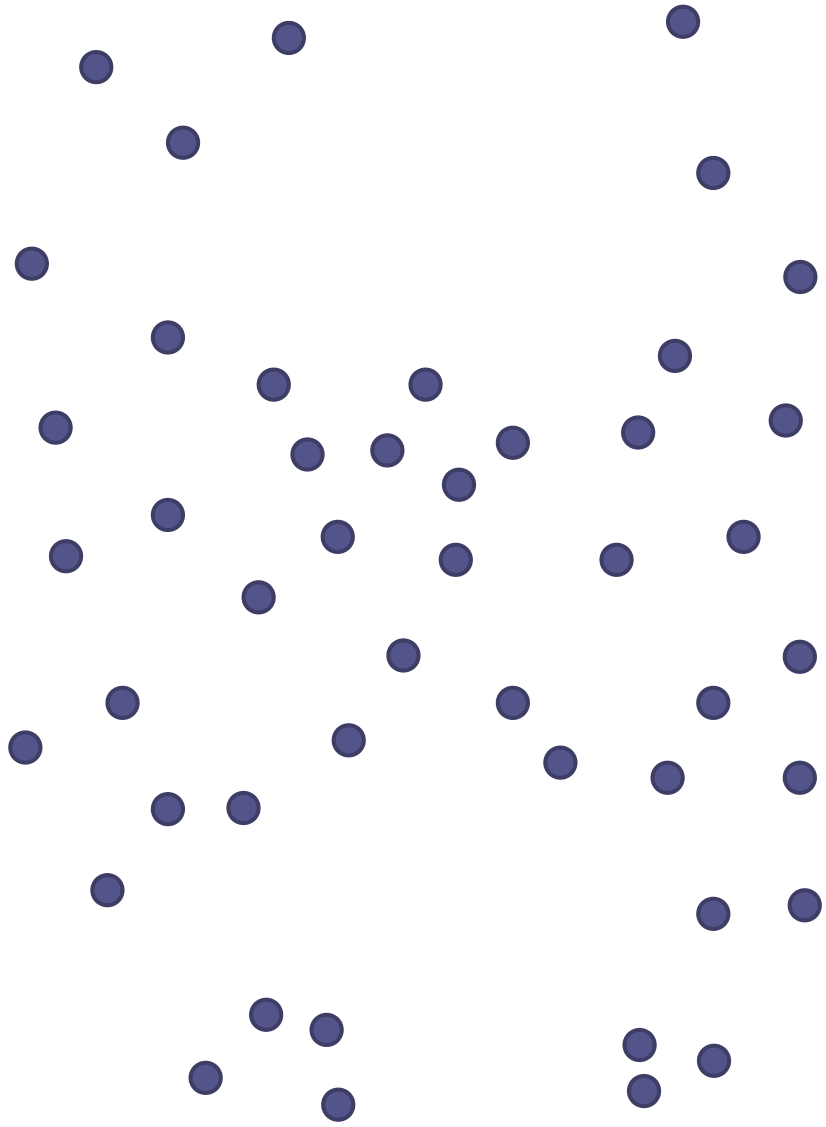
- PageRank
- Problèmes avec PageRank
- Implémentation de PageRank

# Introduction

- Les premiers engins de recherche web utilisaient une variété de techniques
  - Robots d'indexation (*web crawler*) indexaient les mots clefs
    - Facilement manipulés en ajoutant une liste de mots clefs sans pertinence au site web
  - « J'aime » retourné par les utilisateurs après une requête
    - Utilisé de manière inconsistante
  - Sites cliqués par les utilisateurs après une requête
    - Utilisateurs facilement distraits par des liens sans pertinence à la requête mais intéressants
  - Combinaison de résultats de plusieurs outils de recherche
    - Inutile lorsqu'un outil de recherche est toujours meilleur que les autres

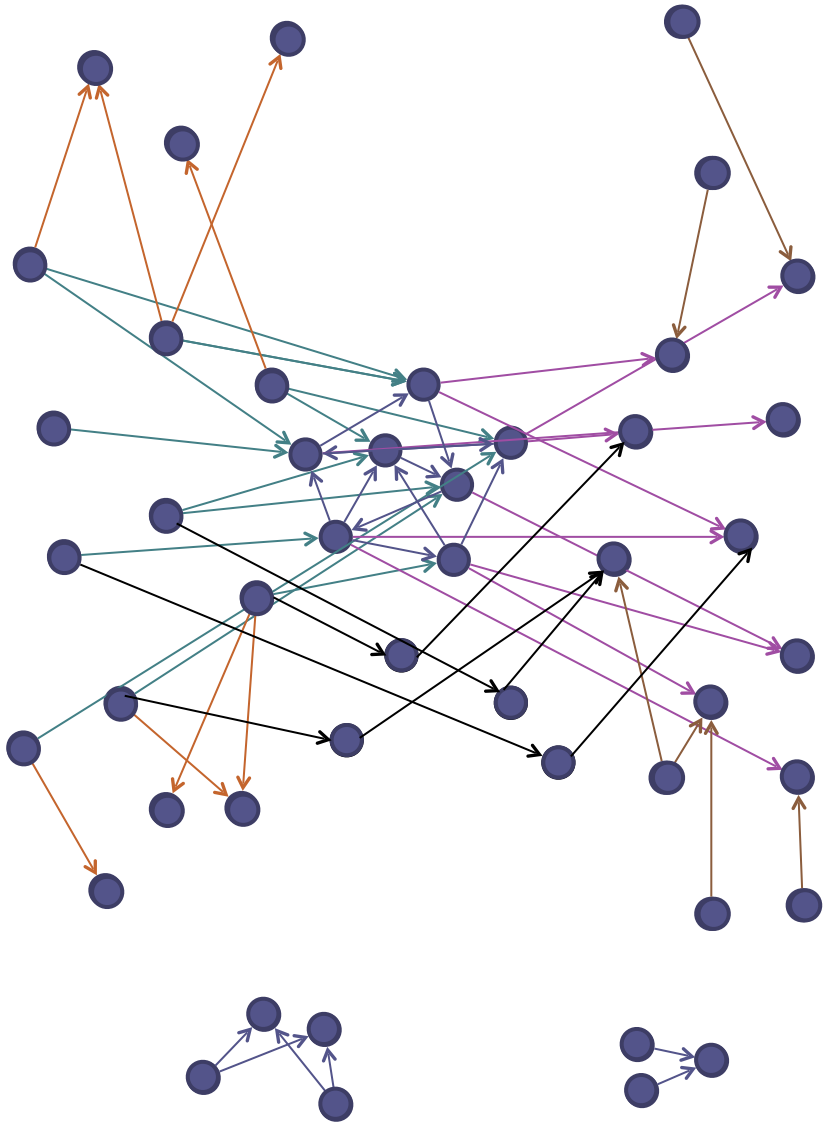
# Introduction

- Problème
  - L'internet est un ensemble de sites indépendants
  - L'information pourrait se trouver n'importe où!
- C'est faux…
  - L'internet est un ensemble de sites interconnectés
  - L'information se trouve sur les sites plus fortement connectés



# Introduction

- Sites fortement connectés (SFC)
  - Sites les plus intéressants du réseau
- Composantes entrantes (CE)
  - Sites qui peuvent atteindre les SFC
- Composantes sortantes (CS)
  - Sites qui peuvent être atteints par les SFC mais ne peuvent pas les atteindre
- Attaches sortantes
  - Sites atteints des CE qui ne peuvent pas atteindre les SFC
- Attaches entrantes
  - Sites qui atteignent les CS mais ne peuvent pas atteindre les SFC
- Tubes
  - Pages liant les CE et les CS qui sautent les SFC
- Composantes isolées
  - Ne peuvent pas atteindre ni être atteintes du reste du réseau

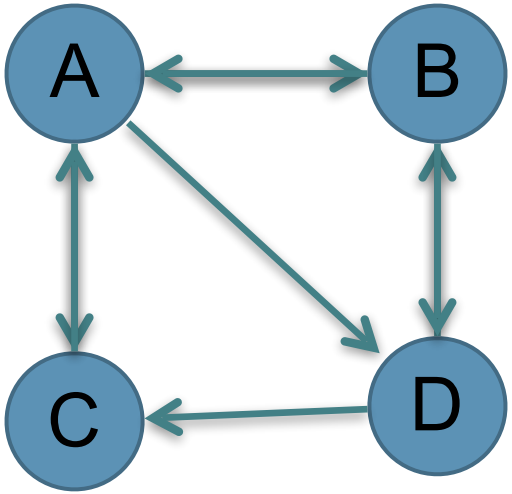


# PageRank

- Juger une page des SFC par son importance et sa pertinence à une requête
  - Pertinence: mots-clefs de la requête apparaissent autour de liens d'autres sites vers la page
  - Importance: nombre de liens d'autres sites vers la page
- Simuler un internaute aléatoire
  - Internaute vote « avec ses pieds », quitte les pages moins importantes/pertinentes pour aller aux pages importantes/pertinentes
  - On peut calculer la probabilité qu'un internaute se retrouve aléatoirement sur chaque page

# PageRank

- Exemple simple de SFC
- La matrice de transitions mesure la probabilité de sauter d'une page à l'autre
  - Matrice stochastique: la somme de chaque colonne est 1




$$\begin{array}{c} \\ \text{Arrive à} \end{array} \begin{array}{c} \begin{array}{cccc} \text{Quitte de} & & & \\ \text{A} & \text{B} & \text{C} & \text{D} \end{array} \\ \begin{array}{c} \text{A} \\ \text{B} \\ \text{C} \\ \text{D} \end{array} \begin{bmatrix} 0 & 1/2 & 1 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} \end{array} = \mathbf{M}$$

# PageRank

- L'internaute est représenté par un processus de Markov
  - Pour un graph connecté
    - Des nœuds connectés par des liens directionnels avec des probabilités associées
  - Si on connait la probabilité d'être sur chaque nœuds à un moment précis
  - Et on connait la probabilité de transition aux autres nœuds par les liens directionnels
  - Alors on peut prédire où on sera probablement au moment suivant

# PageRank

- Simuler l'internaute aléatoire par un processus de Markov:

$$\mathbf{v}_i = \mathbf{M}\mathbf{v}_{i-1}$$

  - $\mathbf{v}_i$ est le vecteur de probabilités d'être sur chaque site à l'itération $i$
- Calculer jusqu'à convergence
  - Typiquement $i \approx 50$
  - Ou utiliser les vecteurs propres de $\mathbf{M}$
  - Ou factoriser l'équation:

$$\mathbf{v}_i = \mathbf{M}^i \mathbf{v}_0$$

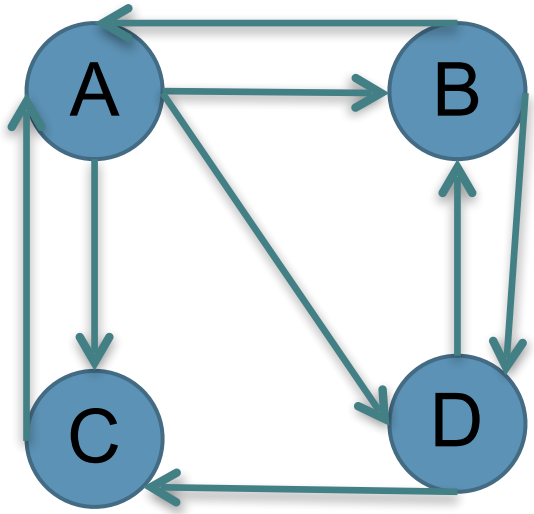
# PageRank

- Où est-ce que l'internaute commence?
  - N'importe où! C'est aléatoire
  - Probabilité égale pour chaque site
  - Pour $k$ sites:

$$\mathbf{v}_0 = \begin{bmatrix} 1/k \\ 1/k \\ \dots \\ 1/k \end{bmatrix}$$

# PageRank (exemple)

- Où se trouvera l'internaute après 0, 1, et 2 itérations?




$$\mathbf{v}_0 = \begin{bmatrix} 0.25 & 0.25 & 0.25 & 0.25 \end{bmatrix}^T$$

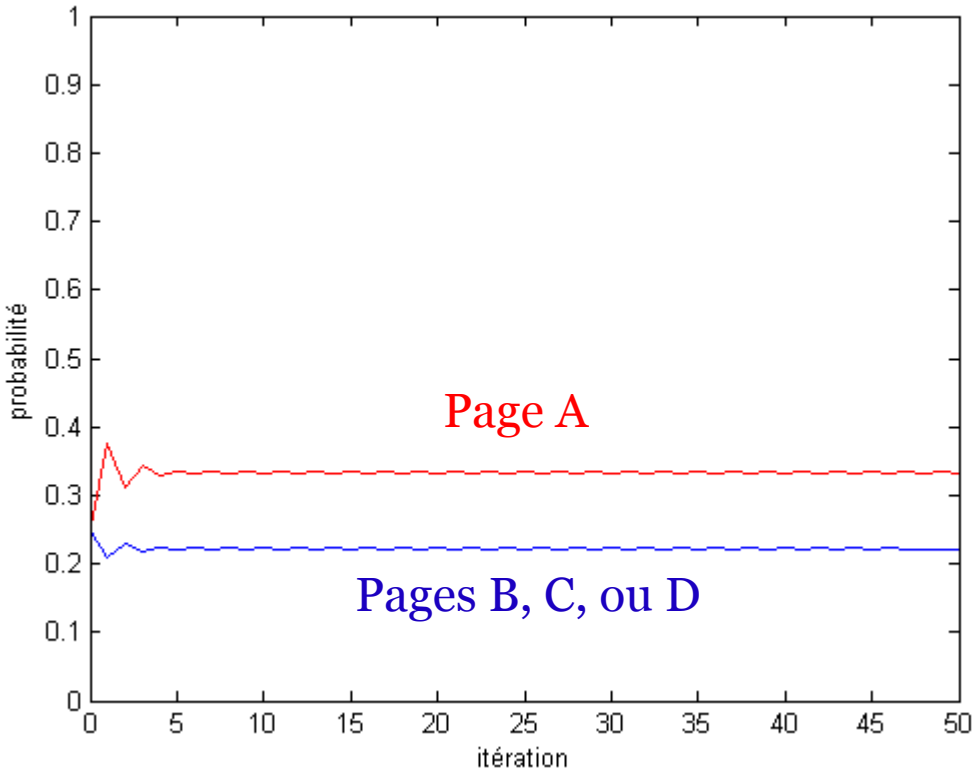
$$\mathbf{v}_1 = \begin{bmatrix} 0 & 1/2 & 1 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix} = \begin{bmatrix} 9/24 \\ 5/24 \\ 5/24 \\ 5/24 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 1/2 & 1 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix}$$

$$\mathbf{v}_2 = \begin{bmatrix} 0 & 1/2 & 1 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} \begin{bmatrix} 9/24 \\ 5/24 \\ 5/24 \\ 5/24 \end{bmatrix} = \begin{bmatrix} 15/48 \\ 11/48 \\ 11/48 \\ 11/48 \end{bmatrix}$$

# PageRank (exemple)

- Convergence de l'exemple sur 50 itérations

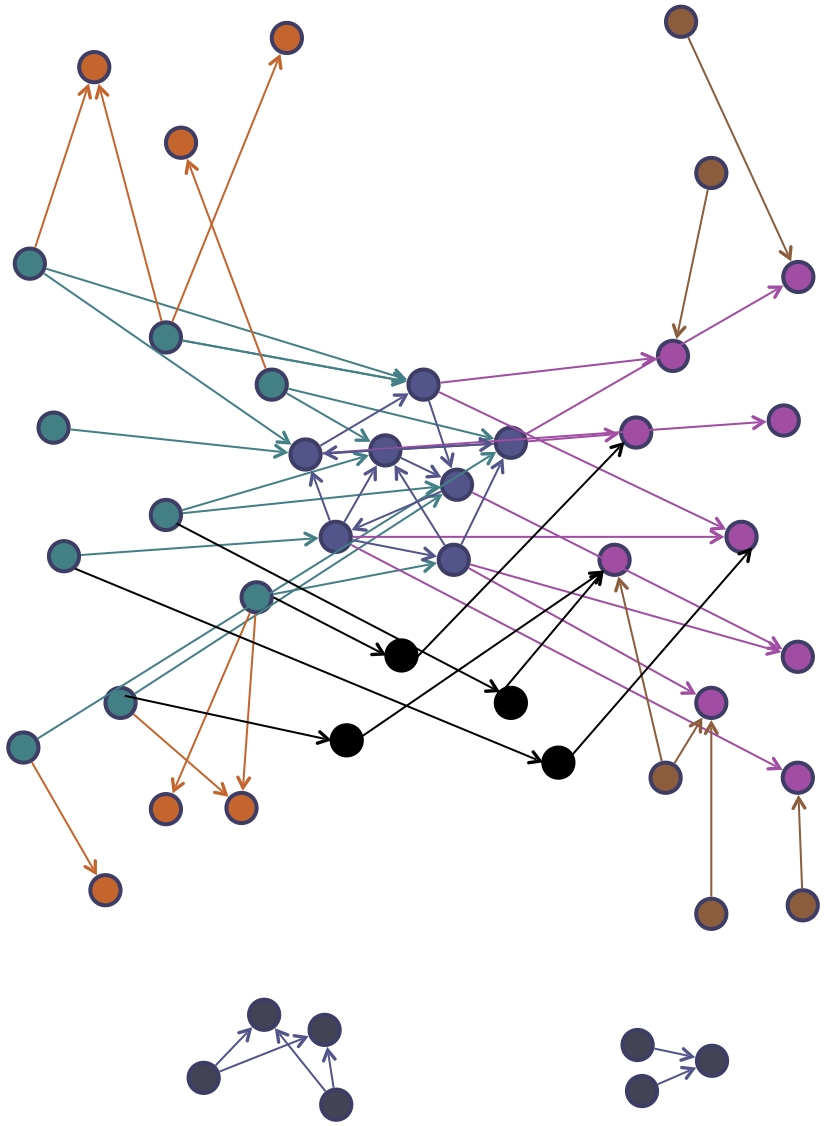


# Problèmes sur le réseau

- Ne pas atteindre les SFC
- Culs-de-sac (pages sans liens)
- Pièges dans la toile (pages qui liens à eux-mêmes)
- Spam de liens (faux liens entrants)

# Ne pas atteindre les SFC

- Problème #1: Tous les sites ne peuvent pas atteindre les SFC
  - Les attaches entrantes et sortantes, les composantes sortantes, les tubes, et les composantes isolées
  - Mais c'est seulement 16% des sites
- Problème #2: Des liens des composantes entrantes ne mènent pas au SFC
  - Mènent aux attachent sortantes et aux tubes
  - Mais c'est une minorité des liens

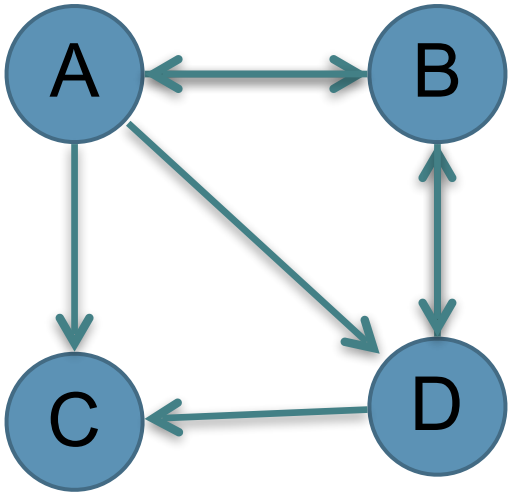


# Cul-de-sac

- PageRank suppose que l'internet est un graph entièrement connecté
- Que ce passe-t'il en cas de cul-de-sac?
  - Page sans liens sortants
  - L'internaute de peut pas partir
  - L'algorithme converge-t'il?

# Cul-de-sac

- Ajoutons un cul-de-sac
- La matrice n'est plus stochastique! (sub-stochastique)




Quitte de

| Arrive à | A | B | C | D |
|---|---|---|---|---|
| A | 0 | 1/2 | 0 | 0 |
| B | 1/3 | 0 | 0 | 1/2 |
| C | 1/3 | 0 | 0 | 1/2 |
| D | 1/3 | 1/2 | 0 | 0 |

$= \mathbf{M}$

# Cul-de-sac

- Où va l'internaute?

$$\mathbf{v}_0 = \begin{bmatrix} 1/4 & 1/4 & 1/4 & 1/4 \end{bmatrix}^T \qquad \sum \mathbf{v}_0 = 1$$

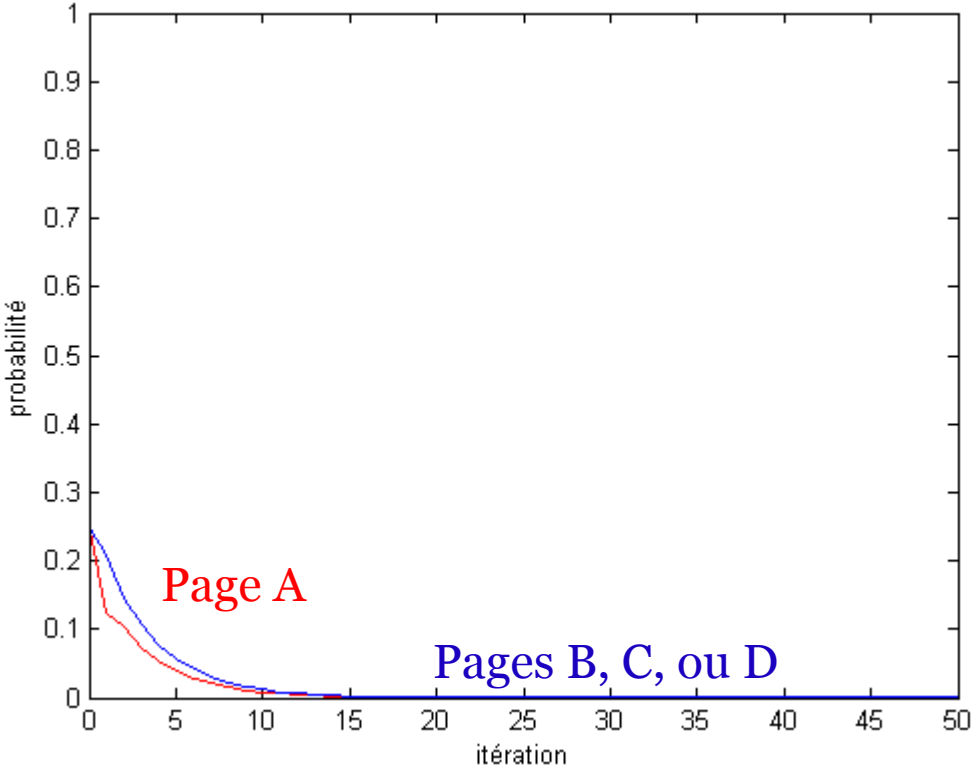
$$\mathbf{v}_1 = \begin{bmatrix} 0 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix} = \begin{bmatrix} 3/24 \\ 5/24 \\ 5/24 \\ 5/24 \end{bmatrix} \qquad \sum \mathbf{v}_1 = 18/24 = 0.75$$

$$\mathbf{v}_2 = \begin{bmatrix} 0 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} \begin{bmatrix} 3/24 \\ 5/24 \\ 5/24 \\ 5/24 \end{bmatrix} = \begin{bmatrix} 5/48 \\ 7/48 \\ 7/48 \\ 7/48 \end{bmatrix} \qquad \sum \mathbf{v}_2 = 26/48 = 0.54$$

$$\mathbf{v}_3 = \begin{bmatrix} 21/288 & 31/288 & 31/288 & 31/288 \end{bmatrix}^T \qquad \sum \mathbf{v}_3 = 114/288 = 0.40$$

# Cul-de-sac

- Où va l'internaute?
  - La probabilité d'être sur n'importe quelle page converge à zéro
  - L'internaute quitte l'internet!



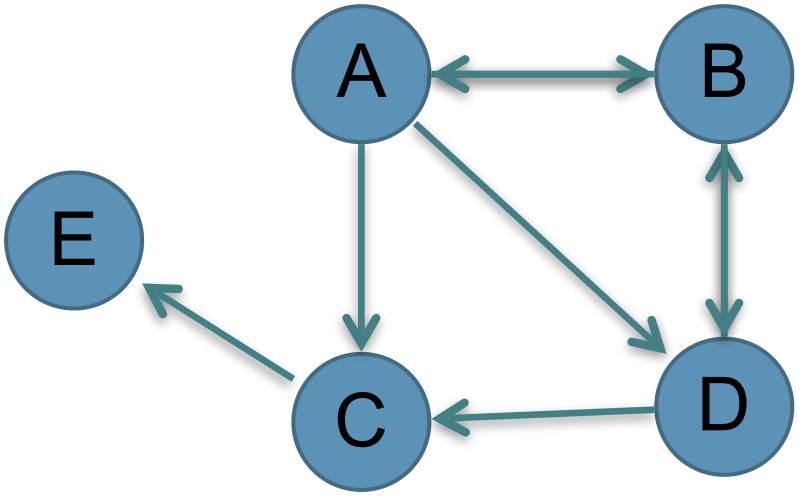
# Cul-de-sac

- Comment régler le problème des culs-de-sac?
- Taxation (on verra plus tard)
- Suppression
  1. Supprimer récursivement les culs-de-sac
     - Récursivement car supprimer les noeux peut créer de nouveaux culs-de-sac
     - Il ne reste que des pages connectées
  2. Calculer PageRank sur le graph connecté
  3. Estimer la probabilité des nœuds supprimés

$$P_{Cul-de-sac} = \sum_{Parents} \frac{\text{Probabilité PageRank du parent}}{\text{Nombre d'enfants du parent}}$$

# Cul-de-sac (exemple)

1. Supprimer récursivement les culs-de-sac
2. Calculer PageRank sur le graph connecté
   - Résultat est
3. Estimer la probabilité des nœuds supprimés




$$P(C) = \frac{P(A)}{Enfants(A)} + \frac{P(D)}{Enfants(D)} = \frac{2/9}{3} + \frac{3/9}{2} = 13/54$$

$$P(E) = \frac{P(C)}{Enfants(C)} = \frac{13/54}{1} = 13/54$$

# Cul-de-sac

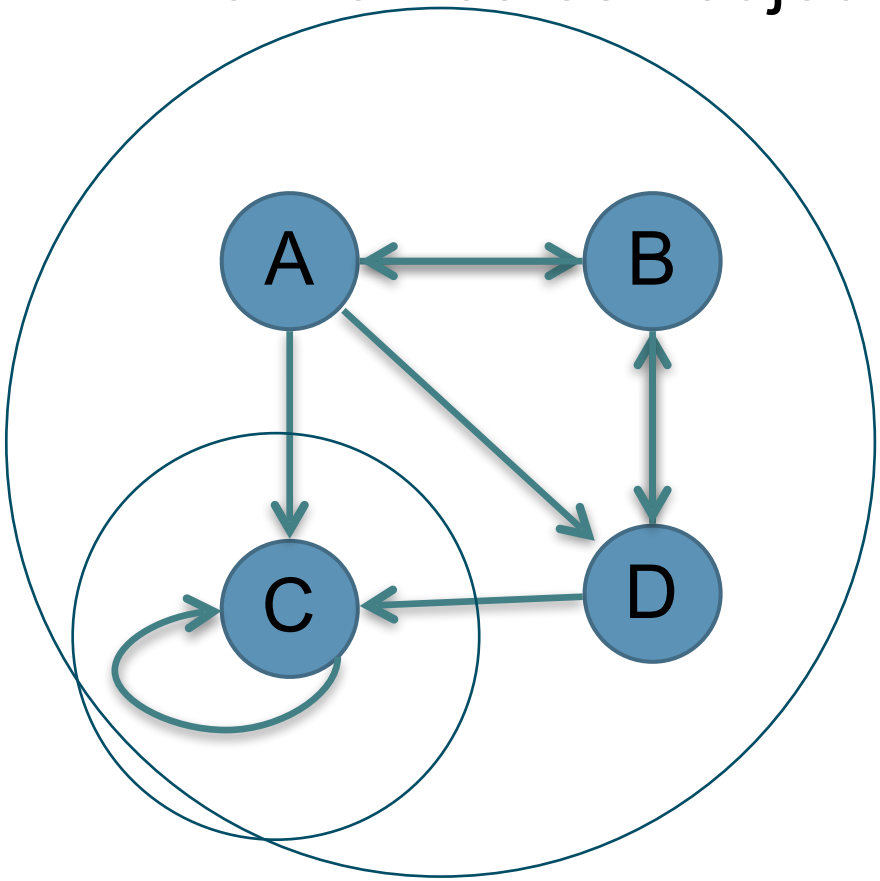
- Remarquez que la somme des probabilités des pages est maintenant plus grande que 1
  - Mais ça marche

# Pièges dans la toile

- Et si quelqu'un essaye de capturer notre internaute?
  - Crée un lien sortant qui ramène à la même page
  - C'est un piège!

# Pièges dans la toile

- Ajoutons un piège
- La matrice est toujours stochastique!



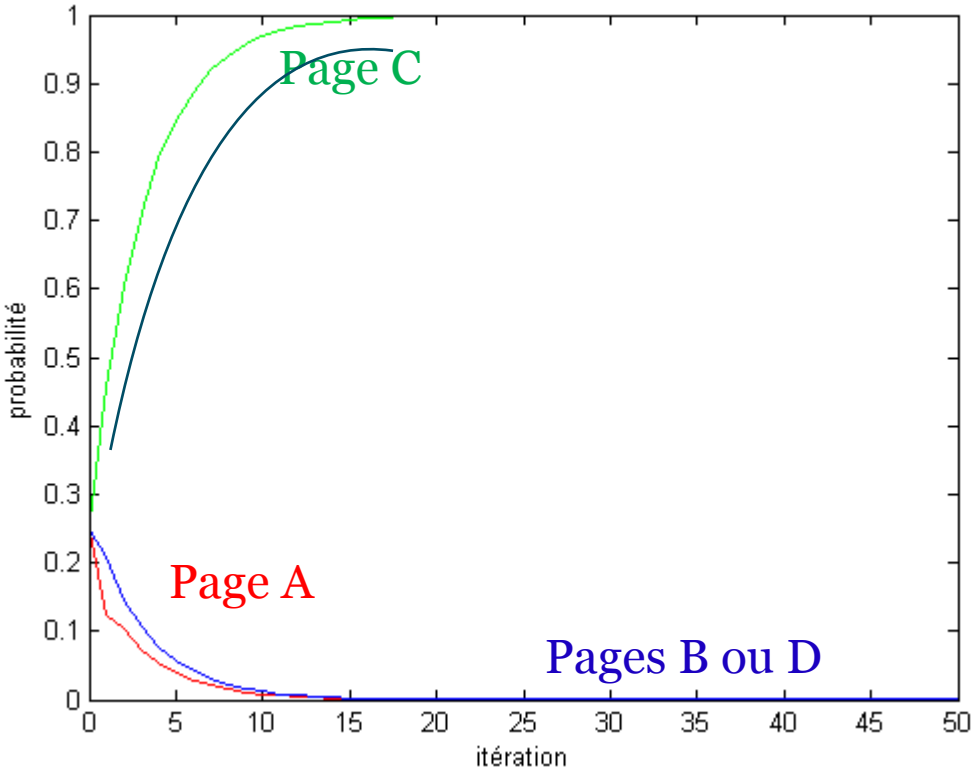

Quitte de

| Arrive à | A | B | C | D |
|---|---|---|---|---|
| A | 0 | 1/2 | 0 | 0 |
| B | 1/3 | 0 | 0 | 1/2 |
| C | 1/3 | 0 | 1 | 1/2 |
| D | 1/3 | 1/2 | 0 | 0 |

$= \mathbf{M}$

# Pièges dans la toile

- L'internaute peut toujours marcher
- Mais il ne peut jamais quitter la page C
- PageRank converge à P(C) = 1 et la probabilité d'être ailleurs = 0



# Pièges dans la toile

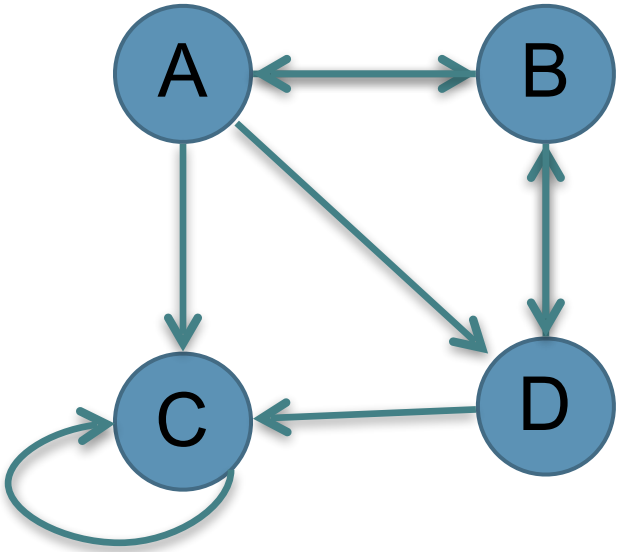
- On ajoute une constante de taxation $\beta$
- Représente la probabilité que l'internaute décide aléatoirement d'arrêter tout et de recommencer la recherche
- Probabilité de $\beta$ que l'internaute continue à marcher, $(1\text{-}\beta)$ qu'il abandonne et recommence

$$\mathbf{v}_i = \beta \mathbf{M}\mathbf{v}_{i-1} + (1-\beta)\mathbf{v}_0$$

- Fonctionne aussi contre les culs-de-sac
  - $(1\text{-}\beta)\mathbf{v}_0$ empêche la probabilité de tomber à zéro

# Pièges dans la toile(exemple)

- Où se trouvera l'internaute après 0, 1, et 2 itérations?




$$\beta = 0.8$$

$$\begin{bmatrix} 0 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 1 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix}$$

$$\mathbf{v}_0 = \begin{bmatrix} 0.25 & 0.25 & 0.25 & 0.25 \end{bmatrix}^T$$

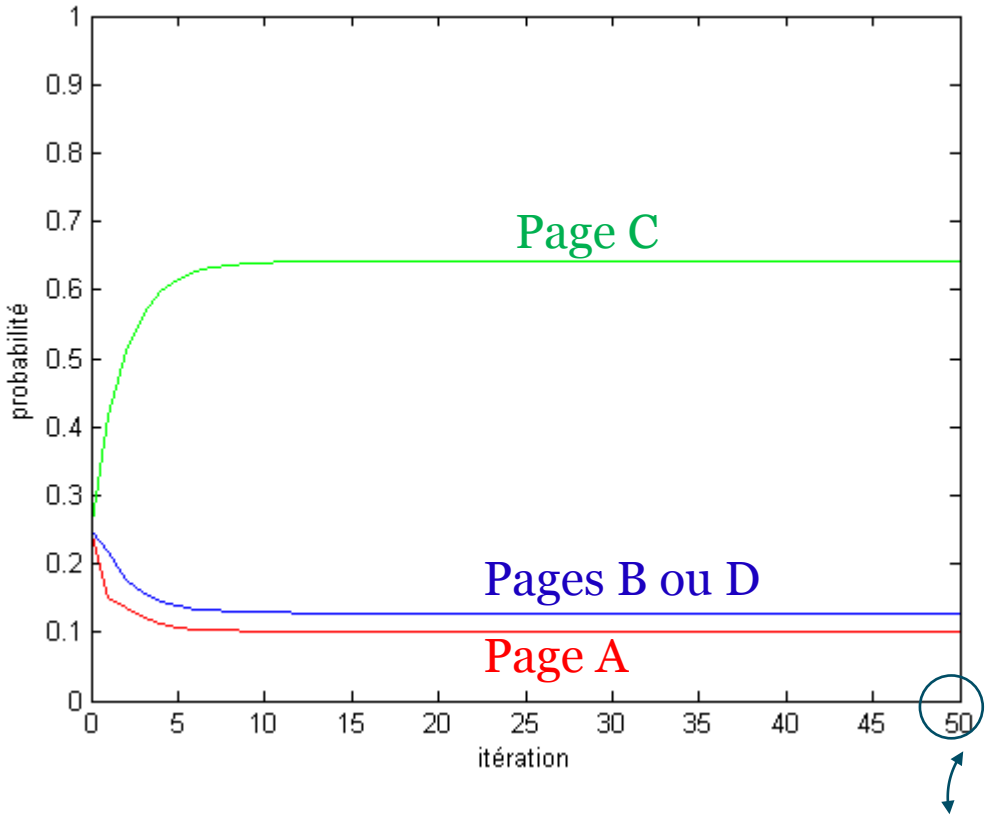
*(A, [B,C, D])*
*(B, [D, A])*
*(D, [B,C])*
*(C, [C])*

$$\mathbf{v}_1 = 0.8 \begin{bmatrix} 0 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 1 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix} + 0.2 \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix} = \begin{bmatrix} 9/60 \\ 13/60 \\ 25/60 \\ 13/60 \end{bmatrix}$$

$$\mathbf{v}_2 = 0.8 \begin{bmatrix} 0 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 1 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} \begin{bmatrix} 9/60 \\ 13/60 \\ 25/60 \\ 13/60 \end{bmatrix} + 0.2 \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix} = \begin{bmatrix} 41/300 \\ 53/300 \\ 153/300 \\ 53/300 \end{bmatrix}$$

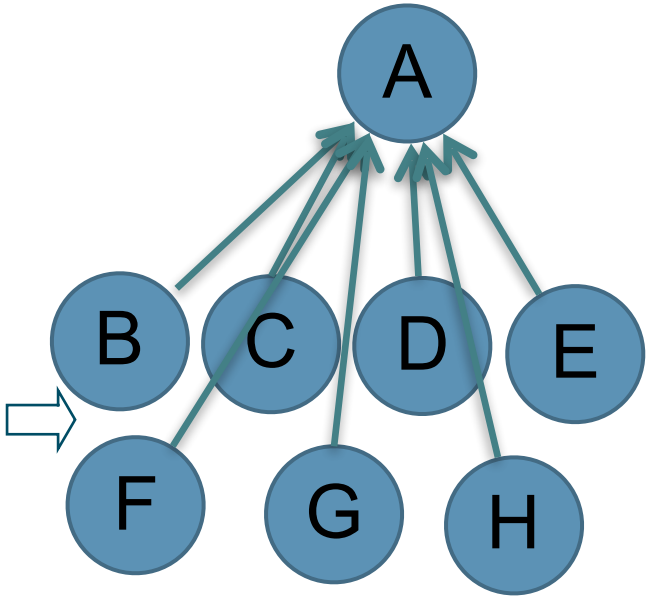
# Pièges dans la toile(exemple)

- La page C reste la plus probable
- Mais les autres pages ne sont pas à zéro



# Spam de liens

- Si quelqu'un crée beaucoup de liens vers leur site sur d'autres pages
  - Utilise des bots pour écrire des commentaires sur des forums/blogs avec le lien
- Le site va dominer dans les probabilités!
  - On doit détecter et éliminer ces pages
  - Mais des pages légitimes ont la même structure (Wikipedia, gouvernement, etc.)
- Quoi faire?



# Spam de liens

- Option 1: TrustRank
- Définir une liste de « sites de confiance » $S$ qui ne sont probablement pas du spam
  - Sites qui contrôlent et limitent qui peut ajouter du contenu: universités, gouvernements, compagnies, médias, etc.
  - Nombre de pages dans $S = |S|$
- Le vecteur de probabilités initiales est:

$$\mathbf{v}_0 = \begin{cases} 0 & \text{si la page n'est pas dans } S \\ \dfrac{1}{|S|} & \text{si la page est dans } S \end{cases}$$

- Crée une marche aléatoire biaisée en faveur des sites pertinents
  - Sont les seuls sites renforcés par taxation
  - Plus de probabilités propagées sur les liens partant de ces pages

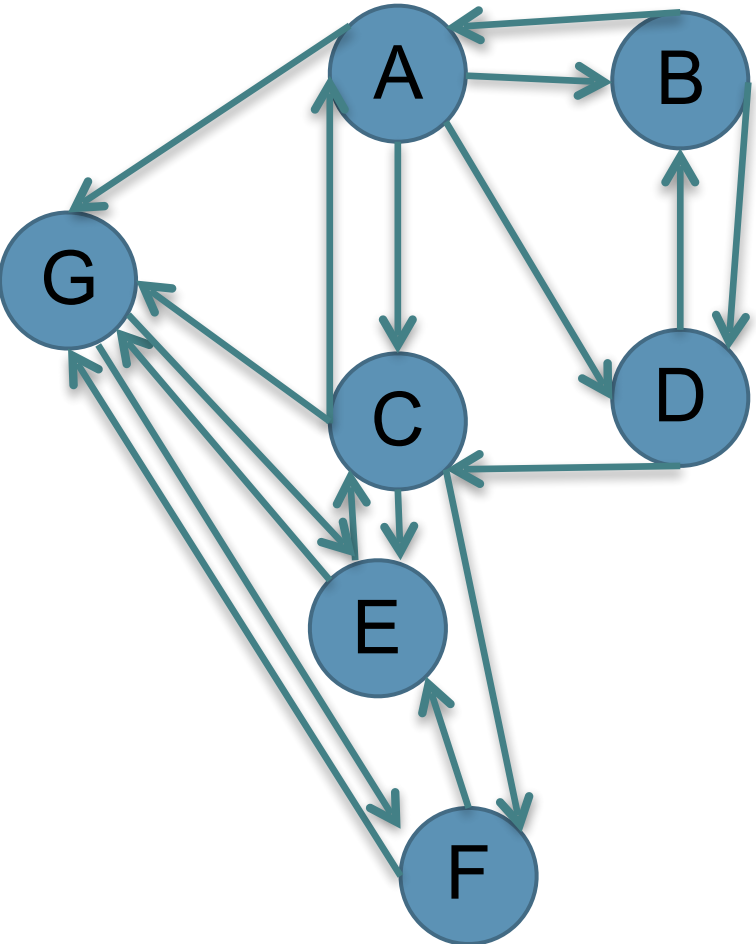
# Spam de liens

- Option 2: Filtrage de masse de spam
- On calcule la probabilité de chaque page $k$ par TrustRank ($t_k$) et PageRank ($p_k$)
- On calcule la masse de spam de la page:

- On filtre les pages ayant une masse plus grande qu'un seuil

# Spam de liens (exemple)

- $S$ = {B, D}, $\beta = 0.8$
- Quelle sera la masse de spam des pages?



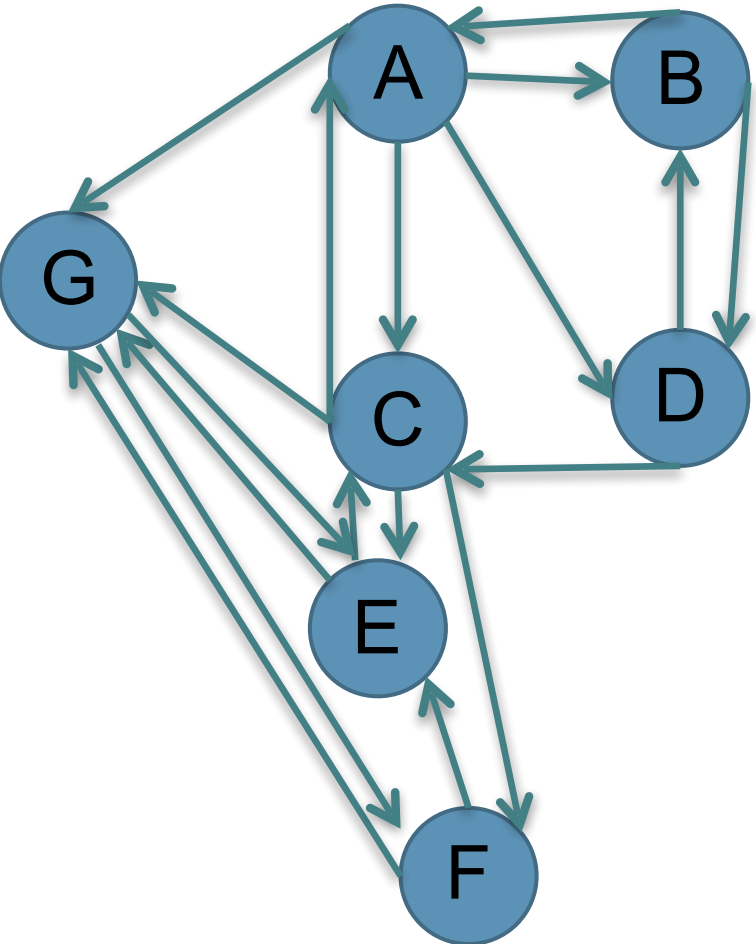

$$\mathbf{M} = \begin{bmatrix} 0 & 1/2 & 1/4 & 0 & 0 & 0 & 0 \\ 1/4 & 0 & 0 & 1/2 & 0 & 0 & 0 \\ 1/4 & 0 & 0 & 1/2 & 1/2 & 0 & 0 \\ 1/4 & 1/2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/4 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 1/4 & 0 & 0 & 0 & 1/2 \\ 1/4 & 0 & 1/4 & 0 & 1/2 & 1/2 & 0 \end{bmatrix}$$

$$\text{PageRank: } \mathbf{v}_0 = \begin{bmatrix} \frac{1}{7} & \frac{1}{7} & \frac{1}{7} & \frac{1}{7} & \frac{1}{7} & \frac{1}{7} & \frac{1}{7} \end{bmatrix}^T$$

$$\text{TrustRank: } \mathbf{v}_0 = \begin{bmatrix} 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 \end{bmatrix}^T$$

# Spam de liens (exemple)

- $S$ = {B, D}, $\beta = 0.8$
- Quelle sera la masse de spam des pages? $(R_{PR}-R_{TR})/R_{PR}$




| Page | PageRank | TrustRank | Masse de spam |
|---|---|---|---|
| A | 0.09 | 0.11 | -0.22 |
| B | 0.08 | 0.20 | -1.50 |
| C | 0.16 | 0.15 | 0.06 |
| D | 0.08 | 0.20 | -1.50 |
| E | 0.21 | 0.12 | 0.43 |
| F | 0.15 | 0.08 | 0.47 |
| G | 0.22 | 0.13 | 0.41 |

# Problèmes sur le réseau

- Remarquez que les problèmes que PageRank rencontrent sont résolus par des constantes et des listes
  - Constante de taxation
  - Liste de sites de confiance
  - Seuil de masse de spam
- L'implémentation de Google a plus de 250 valeurs ajustées précisément

# Sujets dans PageRank

- On a calculé PageRank avec l'importance des pages
- On a fait la supposition que toutes les pages sont également pertinentes
  - Vecteur de probabilités initiales donne une valeur égale à toutes les pages
- ¬Ce n'est pas la réalité
  - L'internaute est intéressé à un sujet spécifique
  - Il commence sur une page pertinente
  - Il clique sur des liens pertinents
  - Il finit sur une page pertinente
- Comment diriger la recherche vers les pages pertinentes?

# Sujets dans PageRank

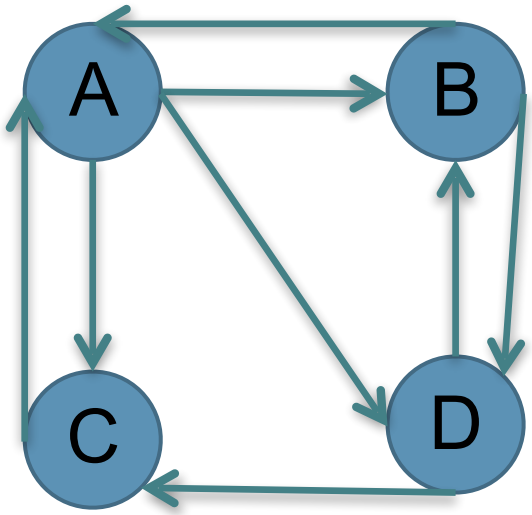
- Suppose qu'on peut déterminer le sujet des pages
- On définit un « ensemble de téléportation » $S$ de pages pertinente à chaque sujet
  - Nombre de pages dans $S = |S|$
- Le vecteur de probabilités initiales est:

$$\mathbf{v}_0 = \begin{cases} 0 & \text{si la page n'est pas dans } S \\ \dfrac{1}{|S|} & \text{si la page est dans } S \end{cases}$$

- Crée une marche aléatoire biaisée en faveur des sites pertinents

# Sujets dans PageRank (exemple)

- $S$ = {B, D}, $\beta = 0.8$
- Où se trouvera l'internaute après 0, 1, et 2 itérations?




$$\begin{bmatrix} 0 & 1/2 & 1 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix}$$

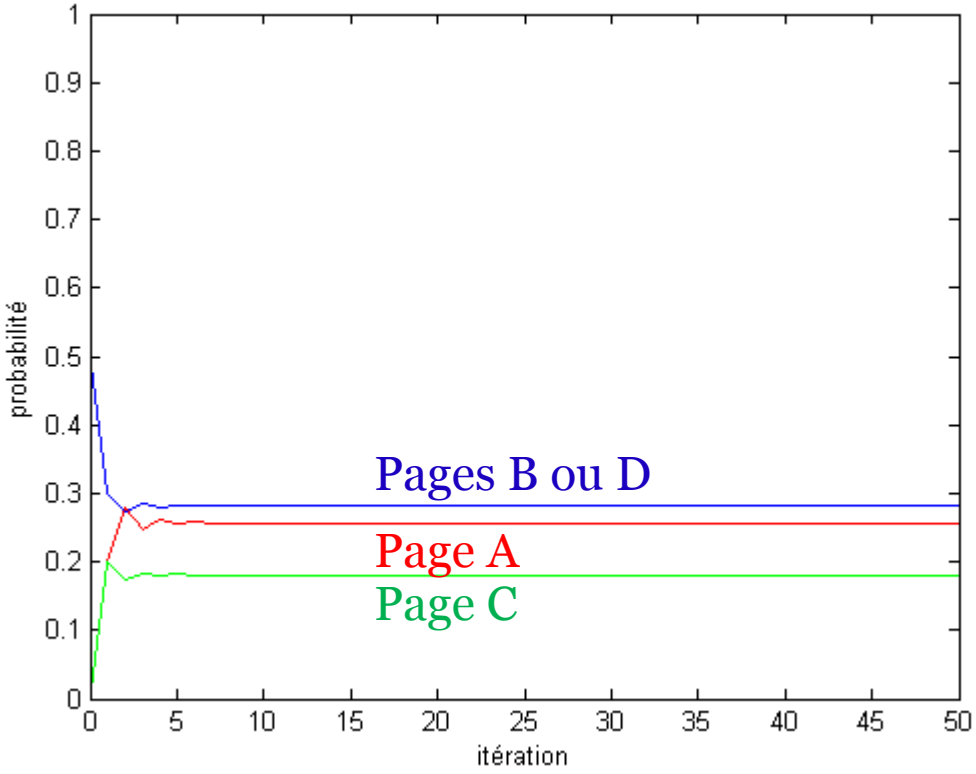
$$\mathbf{v}_0 = \begin{bmatrix} 0 & 0.5 & 0 & 0.5 \end{bmatrix}^T$$

$$\mathbf{v}_1 = 0.8 \begin{bmatrix} 0 & 1/2 & 1 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1/2 \\ 0 \\ 1/2 \end{bmatrix} + 0.2 \begin{bmatrix} 0 \\ 1/2 \\ 0 \\ 1/2 \end{bmatrix} = \begin{bmatrix} 2/10 \\ 3/10 \\ 2/10 \\ 3/10 \end{bmatrix}$$

$$\mathbf{v}_2 = 0.8 \begin{bmatrix} 0 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 1 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} \begin{bmatrix} 2/10 \\ 3/10 \\ 2/10 \\ 3/10 \end{bmatrix} + 0.2 \begin{bmatrix} 0 \\ 1/2 \\ 0 \\ 1/2 \end{bmatrix} = \begin{bmatrix} 42/150 \\ 41/150 \\ 26/150 \\ 41/150 \end{bmatrix}$$

# Sujets dans PageRank (exemple)

- Il est plus probable que l'internaute se retrouve sur les pages pertinentes!



# Sujets dans PageRank

- Comment déterminer le sujet des pages?
- Premièrement, définir un lexique
  - La première version de PageRank indexait 14 millions de mots
  - Exclut les mots vides (*stopwords*) et les mots trop rares
- Deuxièmement, créer des liens
  - Pour chaque mot, lier à une liste des pages où il apparaît

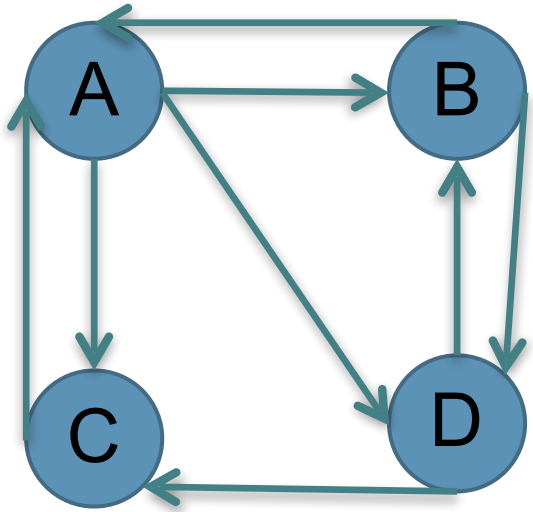
# Sujets dans PageRank

- Troisièmement, mesurer l'importance des mots pour la page
  - Texte spécial (*fancy hits*)
    - URL, titre, meta tags du HTML
    - Texte ancre: texte des liens html de d'autres pages qui mènent à cette page
      - Bonus: permet de cataloguer les images et autres documents sans texte
    - Sont naturellement important
  - Texte ordinaire (*plain hits*)
    - Le reste du contenu de la page
    - Importance dépend des majuscules, de la taille, et de la position sur la page

# Implémentation efficace

- Deux gros problèmes
  - Implémentation efficace de la matrice de transitions
  - Calcul efficace des itérations

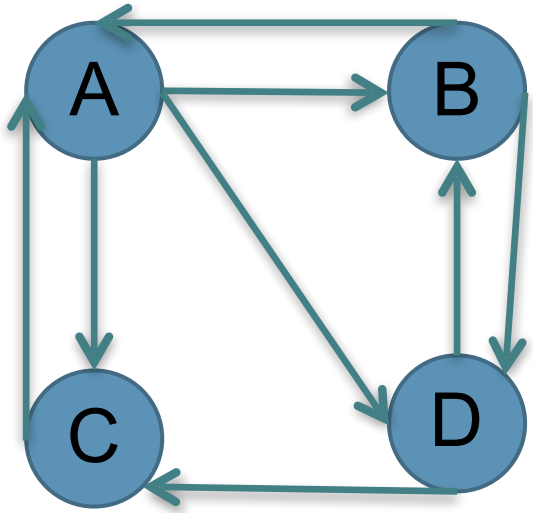
# Implémentation efficace




$$\begin{bmatrix} 0 & 1/2 & 1 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix}$$

- La matrice est creuse (*sparse*)
  - On perdra beaucoup d'espace à emmagasiner des zéros
- Solution standard: emmagasiner uniquement les positions et valeurs différentes de zéro
- On peut faire mieux
  - Chaque élément différent de zéro dans une colonne est 1 divisé par le nombre d'éléments différents de zéro

# Implémentation efficace




| | | |
|---|---|---|
| A | 3 | B,C,D |
| B | 2 | A,D |
| C | 1 | A |
| D | 2 | B,C |

- Emmagasiner:
  - Page actuelle
  - Nombre de liens sortants
  - Liste de destinations de liens
- Rechercher:
  - De la page en cours, vérifier dans la liste si la page cible est une destination valide
  - Si oui, la valeur est 1 / nombre
  - Si non, la valeur est 0

# Implémentation efficace

- Calcul des itérations

$$\mathbf{v}_i = \beta \mathbf{M} \mathbf{v}_{i-1} + (1-\beta)\mathbf{v}_0$$

- $\mathbf{M}$ et $\mathbf{v}$ sont trop gros pour entrer en mémoire
  - Sinon on n'aurait aucuns problèmes à les multiplier
- Solution: diviser $\mathbf{M}$ et $\mathbf{v}$ en sous-matrices et sous-vecteurs, calculer des résultats partiels, puis les combiner
  - Bonus: le calcul est parallélisable ou peut être fait par MapReduce

# Résumé

- PageRank
  - Matrice de transitions
  - Internaute aléatoire avec processus de Markov
- Améliorations
  - Culs-de-sac: suppression
  - Pièges dans la toile: taxation
  - Spam de liens: filtrage par masse de spam
  - Pertinence des sujets: ensemble de téléportation
- Implémentation
  - Représentation de la matrice
  - Distribution des calculs