Notes de cours (ENS Lyon, M1)

Chapitre 4 : Internet et le Web

Table des matières

4 Internet et le Web

On s'intéresse dans ce chapitre principalement à deux grandes notions :
- Le graphe du Web : structure du Web, modèles du Web et aussi le routage dans le Web.
- Le modèle petit monde et le routage dans un graphe petit monde.

On consacrera aussi une partie à PageRank, le critère utilisé pour mesurer la popularité d'une page sur le Web. C'est un des critères utilisé par Google pour classer les pages.

4.1 Le graphe du Web

Le graphe du Web, qu'est ce que c'est ?
- Sommets = pages Web
- Arcs = liens hypertextes d'une page vers une autre

4.1.1 Structure du Web

Pour connaître la structure du Web et pouvoir étudier le graphe du Web, le premier problème consiste à *explorer* ce graphe. Ce processus d'exploration s'appelle un *crawl* : on considère un ensemble initial de pages, puis on suit les liens sortants ce qui permet de découvrir de plus en plus de pages.

Rien ne garantit en revanche que l'on visite tout le Web (possibilité de page vers laquelle personne ne pointe).

On s'intéresse alors à la structure du graphe, et notamment :
- le diamètre du graphe
- la distribution des degrés sortant et entrant des pages (nombre de liens de la page, et nombre de pages qui possèdent un lien vers cette page)
- le niveau de connexité (composantes connexes du graphe)
- la structure macroscopique.

Exploration du Web : les problèmes

Avant de pouvoir explorer le Web, il s'agit de définir un peu mieux la notion de page Web, car l'*ensemble des pages Web* est une notion un peu floue.

Ainsi, il est impossible de découvrir par une simple exploration toutes les pages Web accessibles sur Internet. Certaines pages sont protégées par un mot de passe ou bien possèdent une URL secrète (aucun lien vers cette page, il faut taper directement l'URL pour accéder à la page). En plus, de nombreuses pages sont dynamiques, par exemple générées par des formulaires. On ne pourra donc pas y accéder simplement en suivant des liens.

De plus, étant donné la grande dynamique du Web et sa taille, pendant le temps nécessaire à la récupération d'un grand nombre de pages, de nombreuses autres pages auront été créées.

Une exploration exhaustive semble donc impossible.

Un autre problème se pose au niveau de l'identification des pages Web. On pourrait définir une page par son URL, vu que toute page est accessible depuis

son URL. Cependant, il se peut que la même page apparaisse sous plusieurs URL différentes, par exemple si un serveur possède plusieurs noms ou bien utilise des liens symboliques. On étudie alors le contenu des pages, et on considère généralement que 2 pages avec exactement le même contenu représentent la même page avec 2 URL différentes. Cela est acceptable vu que rien ne distingue les 2 pages, elles contiennent la même information et pointent sur les mêmes pages. Cela ne permet cependant pas d'identifier une page au cours du temps et de ses modifications.

En effectuant des mesures d'URLs multiples, on remarque que certaines pages ont plusieurs milliers d'URL différentes, dues à des liens symboliques. De plus, près d'un quart des pages Web ont plus d'une URL (la plupart de 2 à 4), et il est donc important d'utiliser un mécanisme pour identifier les réplications.

Pour voir si le crawl est la bonne solution pour observer le Web, des informations de crawl ont été comparées avec les informations présentes dans les fichiers de log d'un serveur.

On découvre alors que de nombreuses pages ne sont trouvées qu'avec l'une des techniques.

R : Ensemble des URLs obtenues par des moteurs de recherche. Ils disposent d'un grand nombre d'URLs de base (personnes qui s'enregistrent dans les moteurs de recherche) puis effectuent un crawl sur une grande partie du Web. Les moteurs de recherche ne crawlent pas forcément très en détail un serveur donné, mais on complète l'ensemble R par un crawl local (en ne suivant que les liens internes au serveur).

B : Ensemble des URLs obtenues dans les browsers Web, à savoir les pages atteintes par des personnes navigant sur Internet.

Ensembles obtenus en regardant les fichiers de log des serveurs, et on peut différencier les pages de B des pages atteintes par des robots R.

Expérience réalisée sur un serveur de l'INRIA, avec plus d'un million d'entrées dans les fichiers de log : 1.2 million pour R et 360000 pour B. Robot local d'exploration du site Web, d'où un grand nombre pour R.

Alors, les pages trouvées par R et B sont les suivantes

- R : 15160 pages, dont 1600 pas dans B
- B : 15340 pages, dont 1780 pas dans R

Les nombres sont comparables, et les deux techniques d'exploration sont d'accord sur presque 90% des entrées. En effet,

- la plupart des personnes cherchent à indexer leurs pages par les moteurs de recherche, d'où un grand nombre de pages de B dans R.
- Réciproquement, les pages indexées par les moteurs de recherche apparaissent souvent comme un résultat et alors l'utilisateur va accéder ces pages, d'où un grand nombre de pages de R dans B.

Usuellement, environ les 2/3 des pages connues par un moteur de recherche ont été accédées par les utilisateurs de ce moteur de recherche.

La plupart des différences entre B et R est au niveau des pages personnelles (PP). Seulement 15% des pages étant à la fois dans B et dans R sont des PP, alors que 60% des pages uniquement dans un des ensembles sont des PP.

Certaines PP ne sont pas accessibles depuis le serveur et pas référencées dans les moteurs de recherche, d'où leur appartenance à B-R. Par contre on ne comprend pas pourquoi de nombreuses pages sont dans R-B.

Dans notre exploration B-R est de taille 2000, R-B de taille 1500. On effectue alors un nouveau crawl en partant de B-R. Cela permet de découvrir encore 20000 nouvelles pages qui ne sont ni dans B ni dans R (ensemble N), découvrant notamment 3000 nouvelles PP.

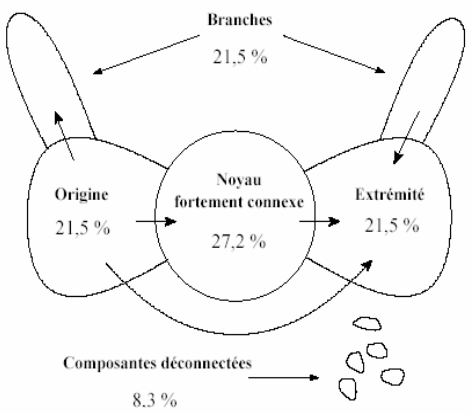
Le Web caché contient ainsi de nombreuses pages normales, pas seulement des pages dynamiques. La technique de crawl est bonne à condition que l'on parte du bon ensemble de départ de pages. Le problème de l'observation du Web se ramène donc à trouver un ensemble de départ convenable pour un crawl.

Structure macroscopique

Pour comprendre la structure macroscopique du Web, de nombreuses explorations ont été effectuées, analysant environ 200 million de pages, et plus de 1.5 milliard de liens.

Une étude effectuée en 1999 par Broder, Kumar, etc... propose de décrire le graphe du Web sous la forme d'un noeud papillon.

- Une composante connexe contenant 90% des sommets du graphe non orienté sous-jacent
- Quelques petites composantes connexes, déconnectées de la grosse composante



En dehors des composantes déconnectées, on remarque 4 zones distinctes :

- Un gros noyau dans lequel on peut aller de n'importe quelle page à n'importe quelle autre en suivant des liens. Coeur du Web, contient plus d'1/4 des pages. Noyau dense : longueur moyenne d'un plus court chemin dans le noyau comprise entre 16 et 20.
- La zone "Origine" : un ensemble de pages qui permettent d'accéder aux pages du noyau, mais qui ne sont pas accessibles depuis les pages du noyau.

– La zone "Extrémité" : pages accessibles depuis le noyau mais à partir desquelles on ne peut pas atteindre le noyau.
– Les "Branches" : on ne peut pas atteindre le noyau, et pages non accessibles depuis le noyau.

On dispose de plus de quelques informations : le diamètre du noyau vaut au moins 28, la distance moyenne entre une page de la zone Origine et une page de la zone Extrémité en passant par le noyau est proche de 900, et enfin la probabilité d'avoir un chemin entre deux pages quelconques est de seulement 24%.

Cette structure classique bien connue du Web peut cependant être mise à défaut en créant une page Web qui permet de générer toutes les URL. Il y a des liens permettant de rajouter n'importe quel caractère à une URL puis en cliquant sur le lien construit, on peut atteindre n'importe quelle URL valide.

Cette page bouleverse donc la structure en noeud papillon. Il est en effet possible de vérifier que cette page appartient au noyau, et vu qu'elle dispose de liens vers toutes les autres pages, la zone "Origine" est intégrée au noyau, alors que les branches et les composantes déconnectées sont intégrées à la zone "Extrémité". La structure du Web est alors uniquement constitué du noyau et de la zone "Extrémité".

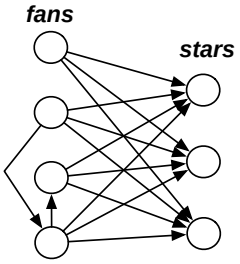
Cela donne à réfléchir sur la notion de graphe du Web, et surtout la difficulté à le définir. Qu'appelle t-on par URL valide ? Comment définir les pages *intéressantes*, ou bien est-ce celles atteintes par un crawl ?

Structure microscopique

On s'intéresse maintenant à l'étude des particularités locales propres au graphe du Web.

On observe notamment une notion de *communauté*, qui peut être définie de différentes façons. Un des problèmes dans l'analyse de la structure microscopique consiste à définir les communautés, un autre à les détecter automatiquement.

Première approche : *fans/stars*. Une communauté est un couple d'ensemble de pages Web telles que toutes les pages du premier ensemble pointent vers toutes les pages du second. De plus, les pages du second ensemble ne pointent pas les unes vers les autres.



Cette structure apparaît souvent : communauté centrée autour d'un sujet de prédilection. Les "fans" ont des pages qui mettent des liens vers leurs "stars". En revanche, les stars n'ont aucune connexion entre elles car ce sont des concurrents (par exemple, des agences de voyage).

Une notion qui revient régulièrement, notamment pour classer les pages Web, est la notion d'*autorité*. On en reparlera lors des méthodes de classifications comme PageRank. Dans le cas d'une communauté fan/star, on voit que les pages des stars font autorité, tout le monde s'y réfère.

Une **deuxième approche** consiste à interpréter la notion de communauté par une collection de pages Web qui possèdent plus de liens vers les pages de la collection que vers les pages externes. Les communautés sont ensuite reliées entre elles par quelques liens.

Cette définition de communauté permet de se rapprocher de problèmes de partitionnement de graphes. De telles communautés sont elles aussi centrées sur un sujet de prédilection.

En plus de ces aspects de structure générale, on s'intéresse à de nombreux aspects de propriétés statistiques sur le graphe du Web. Comprendre le graphe nous permet ainsi de le modéliser et de simuler sur un graphe proche de l'existant.

Propriétés statistiques

Les propriétés les plus étudiées sont :

- **La distance moyenne**. Dans le graphe du Web, la distance moyenne est faible, de l'ordre de 19.
- **Le coefficient de clusterisation**. C'est la probabilité pour que deux voisins d'un même sommet soient eux-mêmes voisins. Ce coefficient est élevé dans le cas du graphe du Web. Cela correspond au fait que deux amis d'une même personne ont une forte chance d'être amis entre eux.
- **La distribution des degrés**. Un résultat important montre que dans le graphe du Web, la distribution des degrés est en *loi de puissance* : la probabilité pour qu'un sommet ait un degré d est d'un ordre proportionnel à $d^{-\alpha}$, avec $\alpha > 1$. Cela signifie qu'il y a peu de sommets de très fort degré, alors qu'il y en a beaucoup avec un faible degré.

La propriété de distribution des degrés en loi de puissance est notamment vérifiée pour le degré entrant (nombre de liens vers une page), ce qui est cohérent avec le fait que la plupart des pages sont très peu référencées, mais certaines le sont énormément. L'étude de Kumar et al a montré que pour les degrés entrants, le coefficient $\alpha = 2.1$, et on a $\alpha = 2.7$ pour les degrés sortants.

Bien sur, ces propriétés sont obtenues uniquement sur des observations partielles du graphe du Web, et biaisées par les méthodes d'exploration utilisées.

Comparaison avec d'autres graphes classiques

Ces propriétés peuvent être retrouvées dans des graphes dans de nombreux autres contextes à la mode :

- Graphe de l'Internet : deux machines reliées s'il existe un chemin entre les deux au travers de routeurs. Ainsi, par exemple, la propriété de clusterisation correspond au fait que 2 ordinateurs reliés à un 3ème ont une forte chance d'être reliés entre eux.
- Graphe des acteurs : deux acteurs reliés s'ils ont joué dans un même film
- Graphe des co-auteurs : deux scientifiques reliés s'ils ont co-signé un même article
- Graphe de dépendance des espèces : deux espèces sont reliées si l'une se nourrit de l'autre
- etc...

Tous ces graphes ont une distance moyenne faible, un fort coefficient de clusterisation et une distribution des degrés en loi de puissance.

4.1.2 Modélisation du graphe du Web

Les propriétés statistiques observées sur le Web ont engendré de nombreux travaux de modélisation. Le but est de générer un graphe ayant les propriétés rencontrées en pratique, afin d'avoir des topologies réalistes pour effectuer des simulations, et afin de pouvoir comprendre les phénomènes sous-jacents...

Plusieurs modèles ont été créés, certains se concentrant sur un paramètre donné, d'autres essayant de tous les prendre en compte. Ces propriétés sont elles le fruit du hasard, et reflètent simplement le fait que les graphes sont quelconques ?

Modèle de Erdös et Rényi (1959)

Ce modèle est un des plus simple, fondé sur des graphes aléatoires. Pour n, m entiers, le graphe $G_{n,m}$ est un graphe à n sommets obtenu en tirant aléatoirement m paires de sommets qui forment les arêtes.

Utilité de ce modèle limitée car seule la propriété de distance moyenne faible est conforme aux observations faites sur le graphe du Web. Mais les autres propriétés statistiques ne sont pas vérifiées sur un graphe aléatoire.

- La probabilité que deux voisins d'un sommet donné soient voisins entre eux n'est pas plus forte que pour n'importe quelle paire de sommets, d'où un faible coefficient de clusterisation.
- Tous les sommets jouent un rôle similaire, donc les degrés sont surtout concentrés autour de la valeur moyenne, la distribution des degrés suit une loi de Poisson.

Modèle de Watts et Strogatz (1998)

Le modèle de Watts et Strogatz cherche à prendre en compte la forte clusterisation rencontrée en pratique.

On part d'un anneau de sommets, où chaque sommet est relié à ses k plus proches voisins sur l'anneau, pour un k donné (anneau standard pour $k = 1$). Pour chaque arête, une des extrémités est remplacée avec une probabilité p par une nouvelle extrémité choisie aléatoirement.

Lorsque $p = 0$, aucune arête est modifiée, donc le coefficient de clusterisation est élevé (exemple avec $k = 2$), mais en revanche la distance moyenne est grande. Par contre, si $p = 1$, on a modifié toutes les arêtes et donc le graphe est un graphe aléatoire comme le modèle de Erdös et Rényi, avec une distance moyenne faible mais également une faible clusterisation.

Il faut alors choisir un p intermédiaire pour obtenir les valeurs voulues : des distances moyennes courtes mais une forte clusterisation.

Ce modèle a fait un premier pas dans la direction de modèles réalistes de topologies. Cependant, il ne capture pas la distribution des degrés en loi de puissance.

Modèle d'Albert et Barabasi (1999)

Albert et Barabasi ont cherché à obtenir la distribution des degrés en loi de puissance. Pour cela, les sommets sont ajoutés un à un et reliés aléatoirement aux sommets préexistants. On parle d'*attachement préférentiel* : un nouveau sommet se lie beaucoup plus facilement avec un sommet déjà beaucoup lié qu'avec un sommet peu lié. En fait, la probabilité d'être relié à un sommet donné croît avec son degré.

L'attachement préférentiel correspond au fait qu'une personne créant une nouvelle page Web aura tendance à pointer vers une page qui est déjà bien connue, donc bien référencée. En effet, il y a peu de chance que la personne connaisse une page peu connue.

La distribution des degrés suit bien une loi en puissance, et en plus la distance moyenne entre les sommets reste faible. En revanche, ce modèle ne respecte pas le coefficient de clusterisation observé dans le graphe du Web.

Modèles de Guillaume et Latapy (2004)

Les modèles les plus récents permettent d'assurer les trois principales propriétés statistiques évoquées précédemment. Ainsi, Guillaume et Latapy présentent un modèle basé sur les graphes biparti.

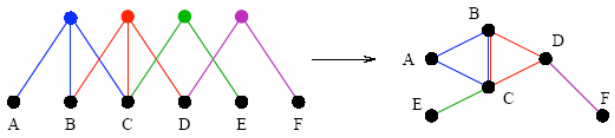
Graphe biparti : $G = (V, V', E)$ où V et V' sont deux ensembles disjoints de sommets, et $E \subseteq V \times V'$ est l'ensemble des arêtes. Les arêtes relient donc uniquement les sommets de V aux sommets de V'.

Certains réseaux complexes sont naturellement modélisés par un graphe biparti, ainsi par exemple le graphe des acteurs est représenté avec un ensemble V représentant les films, et un ensemble V' représentant les acteurs. Les arêtes indiquent quels acteurs jouent dans quels films.

La version classique du graphe biparti, ou projection du graphe biparti sur V', représente effectivement dans notre exemple le graphe des acteurs comme défini précédemment (deux acteurs reliés s'ils ont joué dans le même film). Cette projection $G' = (V', E')$ est définie par :
$\{u, v\} \in E'$ ssi $\exists w \in V | \{u, w\} \in E$ et $\{v, w\} \in E$
En d'autres mots, u et v sont connectés au même sommet de l'ensemble V du graphe biparti G.

Dans G', chaque sommet $w \in V$ correspond à une clique de taille égale au degré de w dans G.

Exemple de transformation de graphe biparti :



En revanche, la transformation inverse n'est pas forcément unique, en particulier pour un graphe qui n'a pas naturellement une structure bipartie. Pour vérifier les propriétés du graphe du Web, il existe des heuristiques pour créer une version bipartie. La distribution des degrés dans la version bipartie est similaire à la distribution en loi de puissance du graphe classique.

On peut alors montrer que la structure bipartie permet d'assurer les 3 propriétés qui nous intéressent pour représenter le graphe du Web.

Deux modèles sont proposés avec cette approche.

- Un graphe biparti aléatoire, avec des distributions des degrés suivant une loi donnée. Les sommets de V et V' sont alors reliés aléatoirement en respectant le degré de chaque sommet. Il faut s'assurer que la distribution des degrés sur chaque ensemble donne un même nombre total d'arête.
- Un graphe biparti construit avec la méthode de l'attachement préférentiel évoquée précédemment. L'ensemble des sommets V' est fixé, avec des degrés suivant une loi de puissance, et on rajoute les sommets de V un à un, en choisissant comment créer les liens. Pour le graphe des acteurs par exemple, c'est la méthode très naturelle, lorsqu'un nouveau film sort on le rajoute et on rajoute les arêtes concernant les acteurs qui jouent dans ce film.

4.1.3 Routage dans le Web

Nous pouvons introduire une notion de routage dans le graphe du Web. En effet, naviguer de page en page revient à suivre une route dans le graphe. On retrouve la notion de plus court chemin : l'auteur d'une page veut par exemple savoir en combien de clics sa page est atteignable depuis une page donnée : notion d'existence de routes et de longueur de routes.

Le problème de l'indexation des pages Web consiste à trouver des pages intéressantes étant donné les mots qu'elles contiennent. Pour identifier les plus intéressantes parmi toutes les pages contenant les mots, une idée consiste à juger l'importance d'une page par les routes qui y mènent : probabilité qu'un surfeur aléatoire tombe sur cette page.

On détaille un peu plus ce concept dans PageRank.

De la même façon, on peut introduire la notion de routage dans le graphe de l'Internet. Il y a des milliards de destinations possibles (2^{32} adresses IP possibles), et les routeurs ont moins de $200,000$ entrées dans leur table de routage (déjà lourd à gérer lorsque des milliers de paquets arrivent chaque seconde).

Pour pouvoir gérer le routage d'un paquet, les adresses IP sont regroupées par préfixes. Les adresses sur 32 bits sont notées en base 256, et par exemple le préfixe $p = 128.93.0.0/16$ représente toutes les adresses dont les 16 premiers bits sont communs avec ceux de l'adresse 128.93.0.0.

Lorsqu'une organisation se connecte à Internet, elle réserve un ensemble d'adresses de même préfixe et n'annonce que ce préfixe aux routeurs de plus haut niveau.

Il y a environ $160,000$ préfixes de nos jours, et ce nombre pourrait être réduit en regroupant mieux les systèmes autonomes (environ $20,000$ systèmes autonomes).

Le routage consiste alors à envoyer un paquet destiné à une adresse a donnée au noeud qui annonce le plus long préfixe de a. Un des algorithmes critiques dans les routeurs Internet consiste donc à trouver dans un ensemble de préfixes celui qui a le plus de bits communs avec une suite de bits donnés.

4.2 Comment classifier les pages Web

L'analyse du graphe du Web a permis d'améliorer la performance des moteurs de recherche actuels. La recherche sur un sujet donné peut produire un très grand nombre de résultats (si le sujet est vaste), et il s'est donc avéré nécessaire de trouver un moyen de classer les résultats de façon pertinente.

4.2.1 Historique des moteurs de recherche

En 1997-1998, AltaVista est le premier moteur de recherche qui classifie automatiquement les résultats pour faciliter la sélection des pages "pertinentes". Le seul critère utilisé alors concernait le contenu de la page, et les balises HTML. Parmi les résultats obtenus, nombreux étaient ceux n'ayant pas d'intérêt pour l'utilisateur, qui devait parcourir de nombreuses pages résultat avant de trouver ce qu'il cherchait effectivement.

Cette méthode, appelée souvent *tri par pertinence*, est donc fondée sur le contenu sémantique d'une page : elle étudie le nombre d'occurrences des termes de la recherche, leur proximité, leur place dans le texte, ... Cette méthode a cependant plusieurs inconvénients :

- Facilité pour un auteur de page de propulser sa page en première position dans le classement : il suffit de répéter les mots importants dans l'entête de la page, ou bien utiliser des techniques de *spamming* (écrire le texte en blanc sur fond blanc).
- Un terme peu précis peut entraîner un utilisateur vers des pages au contenu ambigu.

Les limites du tri par pertinence ont conduit à rechercher d'autres critères de tri pour pouvoir classifier les résultats de recherche. L'idée était de trouver des

critères indépendants du contenu de la page. C'est ainsi qu'ont été inventées des méthodes de tri fondées sur une notion de popularité. On cherche à identifier des pages qui font *autorité* sur le sujet, à savoir des pages qui auront une signification et une utilité pour les utilisateurs qui effectuent une recherche sur le sujet.

Une première méthode, fondée sur la co-citation, se base sur l'algorithme PageRank ou l'algorithme HITS, que l'on va détailler dans la suite. Une deuxième méthode est fondée sur la mesure de l'audience, il s'agit de trier les pages en fonction du nombre de visites qu'elles reçoivent. Ainsi, les pages les plus populaires auront un meilleur classement.

4.2.2 PageRank

Le classement proposé par l'algorithme PageRank exploite la structure de graphe orienté du Web. La méthode utilisée simule le comportement d'un internaute surfant sur le Web, à savoir le parcours aléatoire qu'un internaute va suivre. C'est cette méthode qui a donné naissance au célèbre moteur de recherche **Google** en 1998.

Différentes variantes existent, nous présentons les bases de la méthode.

Notion de rang

A la base de l'algorithme PageRank se trouve un indice numérique, le "rang". Cet indice est calculé pour chaque page référencée par Google, et il représente le nombre de liens pointant sur les pages et leur importance. Chaque page contenant un lien est analysée et évaluée par Google. Les liens qui proviennent de pages "importantes" ont plus de valeur, ils aident à découvrir d'autres pages importantes. Le tri est donc totalement indépendant du contenu, mais seulement fonction des liens.

En amont du processus, le Web est parcouru continuellement par des robots dans le but de découvrir de nouvelles pages, ou bien mettre à jour les anciennes. Ces pages sont stockées dans une base de données, et les liens entre les pages sont stockées séparément, afin de former un sous-graphe du Web (le graphe connu par Google). L'algorithme PageRank est utilisé sur ce graphe pour ordonner les résultats d'une requête d'un utilisateur.

Attention au fait que le "rang" est propre à une page donnée, et ne reflète pas l'importance d'un site Web entier. Si l'on parle d'un site de rang n, il s'agit d'un abus de langage, décrivant le rang de la page d'accueil du site.

La fonction de rang pondérée

Soit G=(V, U) le graphe considéré. V est constitué de n pages, et U représente les liens hypertexte reliant les pages.

Soit $u \in V$ une page Web. $S_u \subseteq V$ est l'ensemble des pages pointées par u, et $T_u \subseteq V$ l'ensemble des pages pointant sur u.

Une première définition simple du rang R serait

$$R(u) = c \sum_{v \in T_u} \frac{R(v)}{|S_v|}$$

Le terme $\frac{1}{|S_v|}$ représente la probabilité d'aller de la page v vers la page u ($|S_v|$ est le degré sortant de v) si v pointe vers u ($v \in T_u$). La constante c permet de garantir une certaine normalisation, par exemple la somme des rangs égale à 1.

On prend en fait également en compte le fait qu'un internaute peut à tout moment aller voir une autre page, *zapper*, au lieu de suivre un lien existant. On introduit pour chaque page u le vecteur de zapping $E(u)$ qui représente la probabilité de zapper lorsqu'on est sur la page u (sauter aléatoirement vers une autre page du Web), et un facteur d'amortissement d, choisi dans $[0, 1]$. E est tel que la somme des $E(u)$ vaut 1, et $E(u) > 0$ pour tout u. Il s'agit d'une loi de distribution sur l'ensemble des pages de V. Souvent, E suit une loi uniforme, $E(u) = 1/|V|$, bien qu'on puisse également personnaliser cette loi.

La définition du rang devient

$$R(u) = c \left(d \times \sum_{v \in T_u} \frac{R(v)}{|S_v|} + (1 - d) \times E(u) \right)$$

Intuitivement, cette équation de rang simule un parcours aléatoire d'un internaute surfant sur le Web. Sur chaque page, l'internaute peut

- soit cliquer sur l'un des liens sortants,
- soit zapper sur une page choisie aléatoirement selon la distribution de E. Le zapping permet d'échapper aux circuits sans issue.

L'algorithme PageRank

Pour calculer le PageRank d'une page, il faut donc avoir calculé celui de toutes les pages pointant vers elle. On initialise avec des valeurs arbitraires de PageRank, par exemple 1. Cette valeur n'a pas d'influence sur le résultat final, mais il est nécessaire de donner la même valeur à toutes les pages.

L'algorithme fonctionne ensuite de manière itérative. Les valeurs du rang sont mises à jour à chaque itération, se rapprochant ainsi de plus en plus de la valeur réelle.

En pratique, la convergence de l'algorithme est assez rapide, ainsi par exemple sur un graphe avec 322 millions de liens, on atteint une approximation raisonnable des rangs en 52 itérations (et la moitié des données a convergé en 45 itérations).

Pour résoudre le système, on peut le réécrire sous forme vectorielle. E et R sont les vecteurs colonne formés respectivement des $E(u)$ et $R(u)$. A est la matrice telle que $A[u, v] = 0$ s'il n'existe pas de liens sortants de la page u vers la page v, et sinon $A[u, v] = \frac{1}{|S_u|}$.

L'équation du rang s'écrit alors

$$R = c(dA^t R + (1-d)E)$$

(A^t est la matrice transposée de A). On cherche à avoir c maximal et la somme des rangs égale à 1. Dire que la somme des rangs vaut 1 revient à dire que la norme 1 du vecteur R vaut 1.

Soit $\mathbf{1}$ le vecteur ligne ne contenant que des 1. Vu que la somme des rangs vaut 1, $\mathbf{1} \times R = 1$, et on peut réécrire la formule : $R = c(dA^t + (1-d)E \times \mathbf{1})R$.

On peut alors en déduire l'existence de solutions à l'aide du théorème de Perron-Frobenius : R est un vecteur propre de la matrice $B^t = dA^t + (1-d)E \times \mathbf{1}$ pour la valeur propre $1/c$.

Notons que si $d = 1$, on n'obtient le résultat que si A est fortement connexe, et ce n'est pas le cas dans le graphe du Web. D'où la nécessité de rajouter le facteur de zapping.

Lorsque la matrice A est *stochastique* (toutes les lignes ont une somme qui vaut 1), la valeur propre principale est 1. On a alors une solution avec $c = 1$, et R peut être interprété comme le vecteur de *probabilités stationnaires* de la chaîne de Markov dont la matrice de transition est B. Les résultats classiques permettent alors de savoir quand le système a une solution unique, et ce qui se passe sinon.

En général, A est une matrice sous-stochastique : seulement une partie des lignes a une somme qui vaut 1. En effet, il existe des pages dont aucun lien ne permet de sortir, et la ligne correspondant à une telle page a une somme nulle. La valeur propre principale est donc telle que $c > 1$.

4.2.3 HITS : Hyperlinked Induced Topic Search

HITS est un algorithme de classement des pages Web, un des plus célèbres avec PageRank, développé par Jon Kleinberg. Tout comme PageRank, cet algorithme se base sur la structure du graphe du Web, et part de la constatation que tous les sites Web n'ont pas la même importance, et ne jouent pas le même rôle.

On peut distinguer :

- Les sites de référence, souvent référencés par d'autres sites, ce sont les véritables sites qui contiennent l'information. On parle souvent de sites qui font **autorité** (*authorities*).
- D'un autre côté, il y a les sites qui jouent un rôle aussi important, mais ne contiennent pas à proprement parler de contenu informatif. Il s'agit de sites contenant des liens vers les autorités, qui permettent ainsi de structurer le Web en indiquant où sont les pages qui font autorité sur un sujet donné. Ces sites sont appelés **pivot**, ou *hub*.

Un pivot possède de nombreux liens sortant pointant vers les autorités, et les autorités sont caractérisées par de nombreux liens entrant venant des pivots. Penser par exemple au modèle fans/stars que l'on a évoqué précédemment : les pivots sont les fans, et les autorités les stars. L'analyse des pivots et autorités

permet ainsi de distinguer l'existence de communautés. C'est l'un des avantages du HITS par rapport à PageRank, qui ne permet pas de repérer les communautés aussi facilement.

HITS, s'il n'a pas été utilisé rapidement dans un moteur de recherche comme Google, a fait ses preuves très tôt dans des développements destinés à étudier des portions limitées du Web, ou pour repérer des communautés sur le Web. Il est utilisé maintenant notamment dans l'application Webfountain d'IBM, ce qui montre qu'il permet le développement d'applications de grande envergure.

Le principe

HITS considère la recherche comme provenant d'une requête de l'utilisateur, et le traitement de chaque requête peut nécessiter un traitement différent.
Les requêtes sont classifiées en

1. *Requêtes spécifiques* : "Est ce que Netscape supporte Java JDK1.1 ?"
2. *Requêtes larges* : "Informations sur Java"
3. *Requêtes de pages similaires* : "Trouver des pages similaires à `java.sun.com`"

Les requêtes de type 1 vont avoir très peu de résultats, et le problème consiste à trouver les pages contenant l'information. En revanche, les requêtes de type 2 vont avoir trop de résultats et il faut filtrer parmi toutes les pages possibles.

La notion d'autorité vise à choisir les pages les plus significatives lors d'une requête large de l'utilisateur qui amène de nombreuses réponses. Il n'est à priori pas évident de trouver les pages qui font autorité car ce ne sont pas forcément les pages qui vont citer les termes pour lesquelles elles font autorité. Pour cela, on étudie le graphe du Web pour déterminer quelles pages font effectivement autorité sur le domaine. Il faut tenir compte des liens insérés uniquement pour faciliter la navigation, ou des liens publicitaires, qui ne contribuent en rien à l' "autorité".

Construction d'un sous-graphe du Web

Si l'utilisateur a une requête, spécifiée par la chaîne de caractères s, on cherche à trouver les pages faisant autorité pour s. Une première étape consiste à utiliser un moteur de recherche classique pour obtenir un sous-ensemble de pages en rapport avec s. L'algorithme HITS tourne alors sur ce sous-graphe.

Ce sous-graphe n'est pas forcément constitué de l'ensemble des pages qui contiennent la chaîne s, car cet ensemble peut contenir un très grand nombre de pages, et de plus certaines pages faisant autorité dans le domaine peuvent ne pas contenir la chaîne s.

Le sous-graphe doit posséder idéalement les propriétés suivantes (S_s est l'ensemble des pages du sous-graphe).

1. S_s doit être petit (algorithmes rapides) ;
2. S_s doit posséder un grand nombre de pages pertinentes (plus facile d'extraire des bonnes autorités) ;
3. S_s doit contenir la plupart des autorités.

Pour construire ce sous-graphe, on part de l'ensemble des t meilleures pages fournies par un moteur de recherche (typiquement, $t = 200$), noté R_s. Cet ensemble vérifie les propriétés 1 et 2, mais généralement pas la 3 (comme on a dit précédemment, de nombreuses pages faisant autorité ne contiennent pas la chaîne s). De plus, on remarque que R_s possède très peu de liens entre différentes pages de l'ensemble. Si l'on recherche par exemple $s = java$, on obtient 15 liens entre pages de R_s, à comparer avec les $200 \times 199 = 39800$ liens potentiels.

Un site qui fait autorité, en revanche, a de fortes chances d'être pointé par une page de R_s. Pour se rapprocher de la propriété 3, on rajoute donc les liens sortant. On rajoute également les liens entrant, en limitant à d liens entrant par page (si une page de R_s est pointée par plus de d pages, on en choisit d au hasard pour rajouter à l'ensemble S_s).

Typiquement, avec $t = 200$ et $d = 50$, on obtient un ensemble S_s de taille comprise entre 1000 et 5000, et qui possède les 3 propriétés.

L'étape suivante consiste à calculer les pivots et les autorités dans le sous-graphe constitué des pages S_s.

Mais tout d'abord, on filtre un peu le sous-graphe.

Filtrage du sous-graphe

On remarque en effet que de nombreux liens ne servent qu'à la navigation au sein d'un même site. Une heuristique simple permet de supprimer ces liens. Un lien est dit *transverse* s'il va d'une page d'un site Web vers un autre site Web (le plus haut niveau de la chaîne URL est différent). Sinon, le lien est *intrinsèque*.

Les liens intrinsèques servent souvent à des fins de navigation au sein d'un site Web, et ils contiennent moins d'information sur l'autorité des pages reliées que les liens transverses. On efface donc tous les liens intrinsèques du graphe.

Il est aussi possible d'avoir une heuristique pour filtrer les liens publicitaires, par exemple en éliminant les liens lorsqu'il y a plus de m liens d'un site vers une page spécifique (par exemple, une référence *"Ce site a été créé par ..."* et un lien vers le concepteur du site). Typiquement, $m = 4..8$.

L'utilisation de ces heuristiques toutes simples a montré son efficacité, en enlevant le problème de traiter tous les liens avec la même importance, et notamment les liens qui ne confèrent pas forcément de l'autorité.

Calcul des pivots et autorités

L'idée la plus simple pour trouver les autorités consiste à regarder les degrés entrants des pages dans le sous-graphe. Cependant, ces pages n'ont pas forcément un degré entrant fort pour la requête considérée, et donc ne sont pas pointées par les pivots. Ainsi, sur la requête "java", on trouve parmi les pages de plus fort degré à la fois l'URL `java.sun.com`, mais aussi la page des livres sur `Amazon`. Il faut donc faire la différence entre les pages faisant autorité pour la requête, et les pages populaires, donc très pointées, mais sans unité thématique.

La présence des pivots permet de faire la différence : une bonne page qui fait autorité est pointée par de nombreux bons pivots, et les bons pivots pointent

vers des pages ayant autorité. Pour rompre le cycle, on utilise un algorithme itératif.

On associe à chaque page un poids de pivot PP et un poids d'autorité PA, ces poids étant des valeurs non négatives. Les poids sont normalisés pour que la somme des carrés pour chaque poids soit égale à 1.

Au départ, tous les poids sont initialisés à 1. L'entier k fixe le nombre d'itérations ($k = 20$).

Alors, on effectue k fois les opérations suivantes :

– Pour chaque page u, calculer $PA'(u) = \sum_{v \in T_u} PP(v)$

– Pour chaque page u, calculer $PP'(u) = \sum_{v \in S_u} PA'(v)$

– Normaliser PA' et PP' pour obtenir PA et PP.

(Rappel : Soit $u \in V$ une page Web. $S_u \subseteq V$ est l'ensemble des pages pointées par u, et $T_u \subseteq V$ l'ensemble des pages pointant sur u.)

On en déduit les meilleures pages autorité et pivot en prenant les pages ayant les plus grandes valeurs de PA et PP.

Il est possible de prouver la convergence de cet algorithme à l'aide de vecteurs propres et valeurs propres, comme pour PageRank. On utilise dans ce cas un résultat de Golub et Van Loan. Les détails sont dans le papier HITS de Kleinberg.

4.2.4 Conclusion

Pour conclure cette section sur les moteurs de recherche, les deux méthodes présentées ici tirent parti de la structure orientée du graphe du Web.

PageRank, utilisé dans le moteur de recherche grand public Google, se base sur le rang des pages, qui correspond à l'importance relative de chaque page du Web.

HITS classifie les pages importantes comme les pages faisant autorité. Un moteur de recherche très puissant est développé sur ces technologies, utilisé à des fins industrielles. La pertinence des résultats semble être meilleure avec cette méthode.

Beaucoup de questions se posent encore cependant, notamment sur la notion de "pertinence" d'une page Web, et les critères sur lesquels mesurer la qualité d'un moteur de recherche. Il faut notamment trouver un compromis entre le temps nécessaire pour trouver une réponse et la qualité de la réponse...

4.3 Routage petit monde

Nous présentons dans cette section les systèmes petit monde, leur utilité pour représenter des systèmes complexes (propriétés statistiques vues précédemment), et leur intérêt pour le routage.

4.3.1 L'expérience de Milgram (1967)

La notion de systèmes petit monde tient son nom de l'expression populaire "le monde est petit", désignant la surprise de constater que deux connaissances

d'un même individu, *a priori* sans rapport, se connaissent entre elles.

On peut considérer l'ensemble des relations sociales comme un réseau d'interactions, et les premières études remontent aux années 1930, avec la création de la sociométrie : mesure objective des relations sociales au sein d'un groupe.

Expérience de Milgram : expérience sociologique effectuée en 1967. Il a demandé à 300 habitants du Nebraska (centre des Etats-Unis) et de Boston (côte Est), de faire parvenir une lettre à un habitant de Boston dont ils ne connaissaient que le lieu d'habitation et la profession, en ne la retransmettant qu'à une personne qu'ils connaissaient personnellement, et ce jusqu'à atteindre la personne cible.

Un quart des lettres sont arrivées à destination, et la longueur moyenne d'une chaîne de porteurs était très faible (5.2) en regard du nombre d'individus et de leur éloignement géographique et social.

Expérience reproduite par *e-mails* en 2003 (Dodds) sur 60 000 individus, et on obtient une longueur moyenne de 4.1 entre individus de continents différents.

Les individus n'utilisent que leurs contacts locaux pour renvoyer la lettre, il s'agit donc d'un routage *décentralisé* de la lettre : seule une vue locale du réseau est utilisée pour transmettre le message. Cette découverte de chemins décentralisée est nécessaire pour les réseaux d'interactions réels, qui comportent un très grand nombre de sommets, et une recherche classique des plus courts chemins n'est pas envisageable (trop coûteuse en temps).

Objectifs

Comprendre la nature de la topologie de tels réseaux, pour comprendre pourquoi le routage est si efficace en utilisant uniquement des connaissances locales, et sans rajouter aucune information de structure globale.

Caractéristiques des petits mondes :
- Peu de connectivité et forte clusterisation : chaque individu connaît peu de membres du réseau (basé sur le voisinage, la profession, les loisirs...)
- Faible diamètre : le chemin pour aller d'un individu à un autre est court (rapidité pour que la lettre arrive dans l'expérience).
- Routage décentralisé très efficace : proche des plus courts chemins sans utiliser d'informations globales.

4.3.2 Le modèle petit monde de Kleinberg

Nous avons vu divers modèles utilisés pour capturer les propriétés des réseaux complexes : graphe aléatoire uniforme d'Erdös et Rényi, modèle d'Albert et Barabasi pour la distribution des degrés, modèles basés sur des graphes bipartis... Le modèle de Watts et Strogatz est un modèle de petit monde : petit diamètre et forte clusterisation.

Cependant, sur un tel graphe, on peut montrer que tout algorithme de routage décentralisé calcule, entre deux sommets de ce graphe, un chemin de longueur au moins polynomiale en n, même s'il existe des chemins de longueur

polylogarithmique (grâce aux liens aléatoires introduits). Il n'existe pas d'algorithme pouvant découvrir ces chemins avec une vue locale uniquement.

Kleinberg a introduit le premier modèle de petit monde (diamètre polylogarithmique) ayant une propriété dynamique de *navigabilité* : possibilité de découvrir de façon décentralisée un chemin de longueur polylogarithmique.

Il s'agit d'une grille carrée de dimension $d = 2$, de côté n, que l'on augmente en ajoutant à tout sommet u un nombre $k > 0$ constant d'arcs aléatoires tirés indépendamment. La destination du j-ème arc de u, pour $1 \leq j \leq k$, est v avec une probabilité proportionnelle à $1/|u - v|^s$.

- $|u - v|$ est la distance entre $u = (u_1, u_2)$ et $v = (v_1, v_2)$ dans la grille : $|u - v| = \sum_{i=1}^{2} |u_i - v_i|$
- $s > 0$ est une constante

Les liens longue distance représentent les connaissances aléatoires. Ce n'est pas totalement aléatoire vu que l'on favorise les voisins proches.

Notons que la distribution des degrés suit une loi uniforme, et on est donc loin d'une distribution en loi de puissance comme on l'observe dans les systèmes complexes. C'est un défaut des modèles petit monde, un peu compensé par la propriété de navigabilité.

Seul le cas où $s = 2$ permet d'obtenir les propriétés de navigabilité. Il existe dans ce cas un algorithme de routage décentralisé qui calcule des chemins de longueur en moyenne en $O(\log^2 n/k)$ entre tout couple de noeuds. Il s'agit de l'algorithme glouton, qui transmet en chaque noeud le message au voisin le plus proche de la cible.

Plus généralement, le résultat est correct sur une grille de dimension d si et seulement si $s = d$. Lorsque $s \neq d$, tout algorithme de routage décentralisé calcule, entre deux noeuds, un chemin de longueur au moins polynomiale en espérance. En particulier, aucune distribution uniforme ($s = 0$) est navigable sur le tore d-dimensionnel, comme c'est le cas pour Erdös-Rényi et Watts-Strogatz. Dans ce cas, il existe une constante α indépendante de n, et la longueur d'un chemin donné par un algorithme décentralisé est au moins $\alpha n^{2/3}$. En effet, la distribution uniforme empêche un algorithme décentralisé d'utiliser des indices fournis par la géométrie de la grille.

Pour donner une indication de la preuve, soit U l'ensemble des noeuds à distance inférieure à $n^{2/3}$ de la destination. La source est en dehors de U avec une forte probabilité, et si le message ne passe jamais par un lien long à un noeud de U, le nombre d'étapes pour atteindre la destination sera au moins proportionnelle à $n^{2/3}$. Cependant, la probabilité que l'un des porteurs du message ait un lien long vers un noeud de U est environ $n^{-2/3}$ [$4j$ *noeuds à distance* j *de la destination, donc* $O((n^{2/3})^2)$ *noeuds dans* U*, et la probabilité est en* $n^{4/3}/n^2 = n^{-2/3}$]. Ainsi, le nombre d'étapes avant que l'on trouve un noeud ayant un lien long vers un noeud de U est également au moins proportionnel à $n^{2/3}$.

4.3.3 Propriétés du modèle

Nous prouvons maintenant le théorème de navigabilité, pour $s = 2$.

Déjà, remarquons qu'il y a $4j$ noeuds à distance j d'un noeud $u = (u_1, u_2)$ (on ne considère pas les cas aux bornes, on suppose la grille suffisamment grande) :
- $(u_1 + 1, u_2 + j - 1), (u_1 + 2, u_2 + j - 2), ..., (u_1 + j, u_2)$
- $(u_1, u_2 + j), (u_1 - 1, u_2 + j - 1), ..., (u_1 - j + 1, u_2 + 1)$
- et idem avec $u_2 \dots u_2 - j$, et $u_1 - j \dots u_1 + j$

Donc

$$\sum_{w \neq u} |u - w|^{-2} \leq \sum_{j=1}^{2n-2} (4j).j^{-2} = 4 \sum_{j=1}^{2n-2} j^{-1} \leq 4 + 4\ln(2n-2) \leq 4\ln(6n)$$

($e = 2.71 < 3$).

La probabilité pour que le sommet v soit choisi comme lien long de u est $|u - v|^{-2} / \sum_{w \neq u} |u - w|^{-2}$, et cette probabilité est donc minorée par $1/(4\ln(6n)|u - v|^2)$.

L'algorithme glouton fonctionne de la manière suivante : à chaque étape le porteur de message retransmet à un lien qui est le plus proche possible de la destination. Pour $j > 0$, l'algorithme est en phase j lorsque la distance du porteur du message à la destination est comprise entre 2^j et 2^{j+1}. Phase 0 : distance au plus 2. Valeur initiale de j : au plus $\log n$.

Phase j, avec $\log(\log n) \leq j < \log n$, u possède le message. B_j : ensemble des noeuds à distance 2^j de la destination :

$$|B_j| = 1 + \sum_{i=1}^{2^j} i = 1 + \frac{1}{2}(2^{2j} + 2^j) > 2^{2j-1}$$

u est au plus à distance 2^{j+1} de la destination, donc à distance au plus $2^{j+1} + 2^j < 2^{j+2}$ de tout élément de B_j. Ainsi, chacun des sommets de B_j fait partie des liens longs de u avec une probabilité d'au moins $p_j = 1/(4\ln(6n)(2^{j+2})^2)$. Si l'un de ces noeuds est un lien long de u, ce sera le voisin de u le plus proche de t, donc il sera choisi pour retransmettre le message.

Le message va ainsi entrer dans B_j avec une probabilité d'au moins $|B_j| \times p_j = 2^{2j-1}/(4\ln(6n)(2^{j+2})^2) = 1/128\ln(6n)$

Notons X_j le nombre d'étapes passées dans la phase j (et donc avant d'entrer dans l'ensemble B_j). En notant E l'espérance, et Pr la probabilité, on a :

$$E[X_j] = \sum_{i=1}^{\infty} Pr[X_j \geq i] \leq \sum_{i=1}^{\infty} \left(1 - \frac{1}{128\ln(6n)}\right)^{i-1} = 128\ln(6n)$$

Ceci est vérifié pour $\log(\log n) \leq j < \log n$, et on peut montrer le même résultat avec $j = \log n$. Lorsque $0 \leq j \leq \log(\log n)$, alors $E[X_j] \leq 128\ln(6n)$ tout simplement car l'algorithme va passer au plus $\log n$ étapes dans la phase j, même si tous les noeuds transmettent le message à leurs contacts locaux (liens de la grille).

Le nombre total d'étapes de l'algorithme est $X = \sum_{j=0}^{\log n} X_j$, et par linéarité de l'espérance, on a $E[X] \leq (1 + \log n)(128 \ln(6n))$, et donc on peut trouver une constante α telle que $E[X] \leq \alpha(\log n)^2$.

Ceci clos la démonstration.

4.3.4 Conclusion

Nous avons donné ici un bref aperçu des systèmes petit monde, de leur intérêt pour modéliser des systèmes complexes. Ils permettent d'assurer l'existence de chemins de distance polylogarithmique entre deux noeuds du graphe, et permettent d'avoir un algorithme de découverte de ces routes en temps polylogarithmique, sans posséder aucune connaissance globale sur le système.