Chercher sur le Web : juste un point fixe et quelques algorithmes

Serge Abiteboul, *directeur de recherche Inria à l'École Normale Supérieure de Cachan*

Le Web met à notre disposition une masse considérable d'information, plusieurs dizaines de milliards de documents. Sans les moteurs de recherche, ces systèmes de plus en plus sophistiqués qui nous aident à nous focaliser sur un petit nombre de pages, le Web ne serait qu'une poubelle à ciel ouvert, gigantesque et inutilisable. Le rôle de ces systèmes est de faire surgir de la masse des internautes une intelligence collective pour évaluer, classer, filtrer les informations. Comment les moteurs de recherche gèrent-ils ces volumes d'information véritablement phénoménaux ? Comment aident-ils les utilisateurs à trouver ce qu'ils cherchent dans cette masse ? Retour sur un des plus beaux succès du Web.

Ce Web introduit par Tim Berners-Lee vers 1990 auquel nous nous sommes si rapidement habitués est fait de documents hypermédia. L'information est en langue naturelle (voir l'encadré *Les langues du Web*), non pas en langage informatique, et les textes sont vaguement structurés avec les balises HTML pour, par exemple, des titres ou des énumérations. Des ancres sur lesquelles il peut cliquer permettent à l'internaute de découvrir des images, de la musique, des films. Elles lui permettent aussi de passer de page en page au gré de son humeur. Et pour trouver de l'information dans ce bazar, quoi de plus simple ? Il suffit d'utiliser cette petite merveille technologique qu'est un moteur de recherche du Web. L'internaute choisit quelques mots clés. Le moteur sait retrouver en un instant les pages qui hébergent ces mots. La magie, c'est qu'il sait aussi proposer parmi les dizaines voire centaines de millions de pages possibles, les quelques pages qui contiennent si souvent ce que l'internaute recherche.

Les langues du Web

En 2012, l'anglais représente environ 45 % des pages du Web, l'allemand 7 %, le français est la troisième langue avec 5 % ; 90 % des langues du monde ne sont pas représentées sur Internet.

Un index du Web

Tout débute par un index. Il s'agit d'une liste ordonnée dans le style d'un index en fin de livre, qui associe à chaque mot la liste des pages du Web qui contiennent ce mot. Par exemple, une entrée dans cet index serait :

Casablanca → http://www.imdb.com/title/tt0034583/

qui indique que le mot « Casablanca » est présent dans cette page Web du site imdb, une base de données en ligne sur le cinéma. Si vous donnez à l'index plusieurs mots comme « Casablanca Bogart Bergman », il retournera la liste des pages du Web qui contiennent l'un de ces trois mots, et notamment la page précédente. La difficulté essentielle ici est la taille de cet index, qui représente des téraoctets de données (un téraoctet = mille milliards d'octets = un million de méga-octets) pour quelques milliards de pages Web. Le serveur qui gère un tel index va connaître des problèmes de passage à l'échelle :

1. Pour indexer plus de pages, le serveur a besoin de plus en plus de stockage pour garder l'index, et chaque requête devient de plus en plus coûteuse à évaluer.
2. Si le nombre d'utilisateurs croît, le serveur reçoit de plus en plus de requêtes.

Dans les deux cas, le serveur est vite submergé. Pour résoudre ce problème nous allons utiliser une technique fondamentale de l'informatique, la technique du hachage. L'échelle du Web nécessite de paralléliser cette technique.

Pour répartir l'index sur, disons, $K = 10$ machines M_1, M_{10}… nous allons utiliser une fonction aléatoire H qui transforme un mot en un entier entre 1 et 10. (Cette fonction est appelée fonction de hachage.) La machine

numéro $H(w)$ sera celle chargée du mot w. Par exemple, les données du mot « France » sont stockées sur la machine H(« France »), disons M_7. Si H est bien aléatoire, les données seront donc partagées relativement équitablement entre les dix machines, ce qui résout le premier problème. Supposons que quelqu'un veuille les données correspondant à France, il interroge seulement la machine M_7. Les requêtes aussi sont donc partagées relativement équitablement entre les dix machines, ce qui résout le second problème.

La taille des données ou le nombre d'utilisateurs peuvent croître, il suffit alors d'adapter le système en augmentant le nombre de machines. Le parallélisme nous a tirés d'affaire et nous permet de passer à l'échelle supérieure. Par exemple Google utilise des milliers de machines dans des « fermes » et disperse ses dizaines de fermes aux quatre coins du monde.

Pourquoi est-ce que cela marche ? Grâce au parallélisme. De manière générale, peut-on prendre n'importe quel algorithme et l'accélérer à volonté en utilisant plus de machines ? La réponse est non ! La recherche a montré que tous les problèmes ne sont pas aussi parallélisables, ou bien pas parallélisables de manière aussi simple. Mais il se trouve que la gestion d'index est un problème très simple, très parallélisable : nous pouvons sans frémir envisager d'indexer de plus en plus de pages, des dizaines de milliards ou plus.

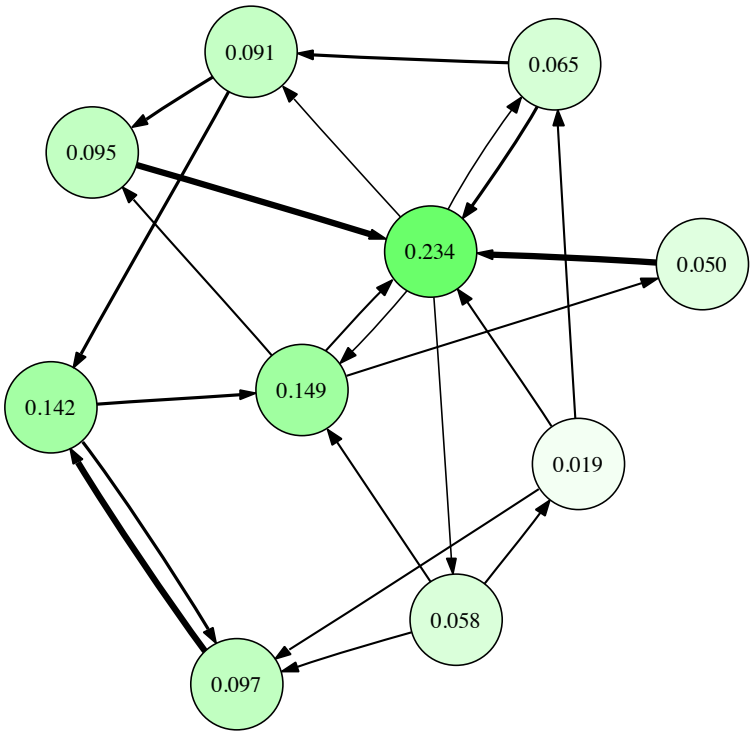

Figure 1. Pagerank : calcul de la popularité des images du web. Chaque cercle représente une image, les flèches et leur épaisseur représentent le nombre de liens d'une page vers l'autre. Les nombres dans chaque cercle, et la couleur, indiquent la popularité.

PageRank

Le cœur du problème reste maintenant de choisir, parmi les millions de pages qui contiennent les mots d'une requête, lesquelles sont les plus intéressantes. C'est essentiel car un utilisateur ira rarement au-delà des dix ou vingt premiers résultats affichés.

Au départ, les moteurs de recherche comme AltaVista utilisaient pour classer les pages uniquement des techniques statistiques de recherche d'information. Une page était jugée plus intéressante si le terme apparaissait dans un titre, ou en caractère gras. Plus un terme est répété dans le document plus il « pèse ». Et également, plus le terme est rare dans l'ensemble des documents, plus il pèse lorsqu'il apparaît. Ce genre de techniques qui marchent bien sur de petits corpus s'avéraient assez décevantes pour le Web.

Les jeunes créateurs de Google ont eu l'idée de baser également le choix des pages sur une technique connue en mathématiques, la marche aléatoire. C'est cette idée inspirée de travaux de Jon Kleinberg qui est à l'origine du succès de Google, un des succès industriels les plus étonnants de l'histoire de l'humanité.

La marche aléatoire

Imaginez un surfeur du Web. Il part d'une page du Web, disons la page www.inria.fr. Et puis il se balade sur le Web en choisissant à chaque étape de cliquer sur un des liens sortant de la page. Si la page n'a pas de lien sortant, il choisit aléatoirement une page n'importe où sur le Web. Et il continue encore et encore, pour toujours, un temps infini. Quel est à l'infini sa probabilité de se trouver sur une page précise ? C'est ce que nous définirons comme la popularité de cette page (cf. figure 1). sur un micro-Web. A priori, une définition abstraite, un joli concept de mathématiques totalement inutile ? Et pourtant ! En pratique cette popularité correspond assez bien à la « popularité » de la page et à ce que les internautes attendent le plus souvent.

Le point fixe

Reste à calculer cette popularité. Pour cela, il faut écrire une équation dont la popularité sera la solution, équation qui code le comportement du surfeur aléatoire. Au cours

des actions du surfeur, chaque page va transférer sa popularité vers les pages vers lesquelles elle pointe (si une page est un cul-de-sac qui ne conduit nulle part, elle partage sa popularité entre toutes les autres pages). En ignorant quelques détails, cela nous conduit à une matrice Q qui capture ces « échanges » de popularité. Le vecteur des popularités pop (la liste de toutes les popularités $pop[i]$ pour chaque page i) se trouve être la solution de l'équation de point fixe :

$$pop = Q \times pop,$$

une notation bien compacte pour un système de dix milliards d'équations à dix milliards d'inconnues (une inconnue pour chaque page Web)…

Et là, banco ! Une technique connue va nous permettre de calculer cette solution. Partons du vecteur pop_0 défini par

$$pop_0[i] = 1/N,$$

c'est-à-dire qu'au départ toutes les pages sont supposées aussi importantes. Et définissons :

$$pop_1 = Q \times pop_0; \quad pop_2 = Q \times pop_1;$$
$$pop_3 = Q \times pop_2 \ldots$$

En continuant ainsi, on arrive assez rapidement à un point fixe qui se trouve être la solution de notre équation. On a donc calculé le vecteur de popularité ! (En pratique six ou sept itérations suffisent pour arriver à une convergence suffisante.)

Pour réaliser ce calcul efficacement avec des volumes de données pareils, il faut des algorithmes très sophistiqués, un engineering de fou. Ce n'est peut-être plus des mathématiques mais c'est de l'informatique de toute beauté. J'ai pu crawler le Web et implémenter un tel algorithme de PageRank avec des étudiants. Cela a été une de mes plus fantastiques expériences de chercheur.

Des recherches à poursuivre

Nous avons présenté une version hyper simplifiée d'un moteur de recherche. Par exemple, celui de Google considère que les textes qui apparaissent près d'un pointeur vers une page font partie du contenu informatif de cette page. C'est le point exploité par le « bombing » (*cf.* encadré *Trouver Chuck Norris*). Les moteurs de recherche doivent se sophistiquer sans cesse, ne serait-ce que pour contrer les attaques des bombeurs qui cherchent à manipuler le Web ou des spamdexeurs qui trichent pour être mieux représentés. Le PageRank de Google actuel utiliserait des dizaines de critères combinés dans une formule gardée secrète. Mais ces moteurs posent encore des problèmes essentiels. Pour n'en citer que quelques-uns :

– Une mesure comme PageRank privilégie la popularité et donc a pour effet d'encourager l'uniformité, les pages populaires devenant de plus en plus populaires et les autres sombrant dans l'anonymat. Est-ce vraiment souhaitable ?

– Pour interroger les moteurs du Web, nous utilisons une série de mots-clés, une « langue » primitive quasiment sans gram-

maire. Ne pourrions-nous pas faire beaucoup mieux ?

– Faut-t-il exclure des pages, parce qu'elles sont racistes, vulgaires, fausses, ou pour favoriser un client, ou ne pas déplaire à un gouvernement ?

– Il y a quelque chose d'extrêmement embarrassant dans la puissance considérable que les moteurs de recherche ont de par leur contrôle de l'information. Devons-nous leur faire confiance sans comprendre leur classement ? Et pourquoi ce secret ?

Pour gérer les volumes d'information du Web, de nouvelles techniques utilisant plus à fond le parallélisme sont sans cesse proposées. De nombreuses techniques sophistiquées sont étudiées pour évaluer, classer, filtrer l'information. Nous pourrions citer par exemple les techniques exploitant les systèmes de notation (l'internaute est invité à noter des services comme dans eBay), les systèmes de recommandation automatisée (comme Netflix ou Meetic), les systèmes qui évaluent l'expertise d'internautes (comme Mechanikal Turk). Le domaine est extraordinairement actif.

Trouver Chuck Norris

À l'heure où ce texte est écrit, si vous tapez « trouver Chuck Norris » sur le moteur de recherche de Google, le premier résultat vous emmène à une page qui dit « Google ne recherchera pas Chuck Norris car il sait que personne ne peut trouver Chuck Norris, c'est lui qui vous trouvera. » Pour arriver à cela, de nombreux internautes ont créé des pages Web qui pointent vers cette page avec comme légende : « trouver Chuck Norris ». Ils sont arrivés à redéfinir collectivement l'expression « trouver Chuck Norris ».

Juste un point fixe et quelques algorithmes

Je me trouvais dans le groupe de recherche sur les systèmes d'information à Stanford en 1995 quand deux jeunes étudiants, Sergei Brin et Larry Page, y travaillaient sur le prototype du moteur de recherche Google. Ils ont développé l'algorithmique nécessaire, mais leurs propositions pouvaient passer pour farfelues. Elles auraient été irréalistes quelques années plus tôt, quand les tailles des mémoires et leur prix auraient demandé d'utiliser un nombre improbable de machines hyper coûteuses. Dans les années 95, cela devenait possible avec un nombre raisonnable de machines bon marché. Les utilisateurs allaient plébisciter leur moteur de recherche. Comme base à ce succès extraordinaire, nous pourrions mentionner un engineering exceptionnel pour faire fonctionner des milliers de machines 24 heures sur 24, des modèles commerciaux révolutionnaires, des techniques de management originales fondées sur un culte de la créativité. Mais en ce qui me concerne, je préfère me rappeler qu'au début, il y avait juste un point fixe et quelques algorithmes.

Pour aller plus loin

Brin S., Page L., (1998). *The Anatomy of a Large-Scale Hypertextual Web Search Engine*. WWW Conference.

Abiteboul S., Manolescu I., Rigaux P., Rousset M.-C., Senellart P., *Web data management*, Cambridge University Press. http://webdam.inria.fr/Jorge

Kleinberg J., (1999). *Authoritative sources in a hyperlinked environment. Journal of the ACM.*

Abiteboul S., Preda M., Cobena G., (2003). *Adaptive On-Line Page Importance Computation*. WWW Conference.