Comment les moteurs de recherche détectent-ils le contenu dupliqué ?

Retour au sommaire de la lettre

Domaine :	Recherche	**Référencement**
Niveau :	Pour tous	**Avancé**

L'un des grands défis qu'ont à relever les moteurs de recherche actuels est la détection du "duplicate content", très présent sur le Web (selon plusieurs études, 30 à 40% du Web serait dupliqué). Pour arriver à leurs fins, ces moteurs doivent donc mettre en place des méthodes efficaces sans consommer trop de ressources techniques. Voici l'une d'entre elles, couramment utilisée à l'heure actuelle...

Pour un moteur de recherche comme Google, être capable de détecter les copies multiples d'un même contenu est une tâche très importante. En effet, d'après Fetterly, Manasse et Najork (Dennis Fetterly, Mark Manasse, Marc Najork: On the Evolution of Clusters of Near-Duplicate Web Pages. LA-WEB 2003: 37-45), environ 30% du web est redondant, c'est-à-dire composé de pages web qui contiennent un contenu déjà présent, au moins en partie, dans une autre page. D'autres sources vont même jusqu'au chiffre de 40% (Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze: Introduction to information retrieval. Cambridge University Press 2008, ISBN 978-0-521-86571-5, pp. I-XXI, 1-482).

Les raisons de la duplication de contenu sont diverses. Et la plupart sont d'ailleurs tout à fait légitimes. Il existe des miroirs de sites web (pages de *man* par exemple – i.e. des pages présentant des extraits du manuel des commandes unix), des contenus qui existent en divers formats (HTML, PDF, etc.), des textes « canoniques » (mentions légales, contrats, licences logicielles, etc.), des actualités reprises depuis l'AFP ou Reuters, etc.

On retrouve ensuite du contenu dupliqué pour de moins bonnes raisons : par exemple à cause d'erreur de développement (même contenu accessible aux travers de plusieurs URL), mais surtout par malice, lorsqu'un black hat souhaite générer rapidement un site web, et va pour cela le remplir d'un contenu repris ailleurs…

On peut avancer trois raisons qui vont pousser un moteur de recherche à détecter le contenu dupliqué :
- **La réduction de la taille de l'index**. En n'indexant pas le contenu dupliqué, le moteur fait des économies au niveau de ses dépenses en espace de stockage.
- **Le nettoyage des SERP**. Ce qui compte pour l'utilisateur, c'est d'accéder au contenu. Si un contenu est disponible sur 24 pages web *via* le moteur de recherche, c'est 23 de trop ! Les places sont chères, le moteur a intérêt à présenter à l'internaute des pages variées.
- **La priorisation du crawl**. Il n'est pas utile de crawler plusieurs fois le même contenu. En détectant le duplicate, le moteur fait des économies en envoyant ses robots aux endroits où l'information est inédite.

Détecter si un texte est une copie exacte d'un autre, ce n'est pas un problème de calcul de similarité sémantique. C'est au contraire un problème de similarité syntaxique. Les outils qu'un moteur va utiliser pour détecter le contenu dupliqué sont donc très différents de ceux qu'il utilise pour déterminer la pertinence d'un texte à une requête.

Par ailleurs, le contenu exactement dupliqué n'existe pas vraiment sur le web. Plus précisément, les seuls cas de contenu exactement dupliqué sont ceux qui correspondent à l'erreur classique de la page accessible à plusieurs adresses suite à une erreur de développement ou de configuration. Dans tous les autres cas, le contenu est en quasi-duplication, ce qu'en anglais on appelle « near-duplicate ». Même dans les cas légitimes de reprise de contenu il s'agit le plus souvent de quasi-duplication. Il suffit d'une date

différente ou d'une publicité textuelle qui change pour que l'on sorte du cadre de la duplication exacte.

Il existe de très nombreuses méthodes pour déterminer si un contenu est une version quasi-dupliquée d'un autre contenu. Dans cet article, nous n'en présenterons qu'une seule, et plus précisément nous donnerons l'intuition de la méthode, les détails techniques étant particulièrement complexes. Cette méthode a un nom, il s'agit de l'algorithme des shingles, qui a été proposé en premier par Broder, Glassman, Manasse et Zweig (Andrei Z. Broder, Steven C. Glassman, Mark S. Manasse, Geoffrey Zweig: Syntactic Clustering of the Web. Computer Networks 29(8-13): 1157-1166 (1997)) en 1997.

Caractériser deux contenus presque identiques ?

La première chose à faire pour mettre en place une méthode qui détecte les contenus quasiment identiques, c'est de définir formellement ce que l'on appelle la quasi-duplication.

Comme pour beaucoup de définition formelle d'éléments flous, on va avoir une définition mathématique avec un critère « expérimental ». On va alors calculer une distance entre textes. On aura alors une définition formelle. Ensuite, on dira que tous les textes qui sont proches, c'est-à-dire à petite distance l'un de l'autre, sont en quasi-duplication. On va donc obtenir un seuil, défini expérimentalement, pour trier ce qui est du contenu dupliqué de ce qui ne l'est pas.

La distance que l'on va utiliser est la distance d'édition avec mouvements. C'est une généralisation de la distance de Levenshtein (http://fr.wikipedia.org/wiki/Distance_de_Levenshtein). La distance d'édition avec mouvements (EDM pour *Edit Distance with Moves*) entre deux textes correspond au nombre minimum de mots qu'il faut supprimer, ajouter ou inverser pour passer d'un texte à l'autre.
Par exemple, la distance entre :

Les loutres mangent du poisson

et

Les loutres mangent du poisson savoureux

a une valeur de 1 car il suffit de rajouter le mot « savoureux » pour obtenir la deuxième phrase à partir de la première.

Notez également que la distance entre la deuxième phrase et la phrase :

Les loutres mangent du savoureux poisson

est aussi 1.

Le problème de la distance d'édition avec mouvements est qu'elle est calculatoirement très compliquée à calculer (Dana Shapira, James A. Storer: Edit distance with move operations. J. Discrete Algorithms 5(2): 380-392 (2007)). En pratique il faut donc trouver un autre moyen de caractériser la distance entre les textes. On est en fait un peu dans la même problématique que pour le calcul de la pertinence, qui est trop complexe sur des textes complets, et pour laquelle un moteur de recherche va opérer une lemmatisation des phrases, ce qui va permettre de diminuer la complexité de la méthode.

Les shingles

Ainsi, pour travailler sur un ensemble de données plus petit, on va travailler sur une représentation simplifiée des textes. Pour être exact, on va utiliser une représentation des textes sous forme de *shingles*.

Shingles, c'est le mot "fun" pour désigner un n-gram, c'est-à-dire un ensemble de n mots consécutifs d'un texte. Le mot ne vient pas de nulle part, un shingle c'est une ardoise de toit en bitume, qu'on pose en quinconce pour recouvrir une toiture en assurant l'étanchéité :

Pour un texte, un shingle c'est la même chose : un bloc de n mots consécutifs. Tous les shingles d'un texte couvrent complètement le texte, et ils se recouvrent entre eux.

Par exemple, le texte suivant :

Les loutres mangent du poisson

a pour shingles de taille 3 : « les loutres mangent », « loutres mangent du » et « mangent du poisson ». Il y a plus de shingles de taille 2 (« les loutres », « loutres mangent », etc.), moins de taille 4, et un seul de taille 5 (le texte lui-même).

Quand on veut travailler sur des textes, on va donc fixer arbitrairement la taille des shingles qu'on considère, et la plupart du temps on va les représenter par des valeurs entières *via* un codage (mais il s'agit là d'un détail technique que nous n'aborderons pas ici). Ce codage facilite les calculs par la suite.

La distance de Jaccard et les shingles

Une fois qu'on a construit pour chaque texte son ensemble de shingles, on va pouvoir définir une distance entre les textes en calculant l'intersection et l'union des ensembles de shingles.

Reprenons l'exemple du début avec les deux phrases « les loutres mangent du poisson » et « les loutres mangent du poisson savoureux ». On va considérer les shingles de taille 2.

Soit A l'ensemble de shingles de la première phrase, et B celui de la deuxième phrase. On a alors :
A = {les loutres ; loutres mangent ; mangent du ; du poisson}
B = {les loutres ; loutres mangent ; mangent du ; du poisson ; poisson savoureux}

L'intersection entre A et B est {les loutres ; loutres mangent ; mangent du ; du poisson}, elle contient 4 shingles. L'union entre A et B est {les loutres ; loutres mangent ; mangent du ; du poisson ; poisson savoureux}, elle contient 5 shingles.

On va ensuite pouvoir calculer le coefficient de Jaccard entre les deux ensembles.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

A ∩ B est l'intersection entre A et B. A ∪ B est l'union entre les deux ensembles. Ici le coefficient de Jaccard vaut donc 4/5.

Plus le coefficient de Jaccard est haut (c'est-à-dire proche de 1), plus les textes sont en duplication. A noter qu'il y a un lien entre Jaccard et la distance d'édition avec mouvements, mais que ce lien est subtil (notamment, il faut prendre en compte la taille des textes pour trouver une relation entre les deux notions).

Il reste cependant un problème : calculer le coefficient de Jaccard est aussi une tâche coûteuse. L'astuce qui permet de réaliser le calcul à moindre coût fait appel à un tirage aléatoire. Nous allons la décrire maintenant.

La méthode « approchée » des shingles

On va maintenant **piocher au hasard** un shingle dans l'union des ensembles de shingles des deux textes, c'est-à-dire dans A ∪ B.

Ensuite, on va regarder si le shingle qu'on a tiré au hasard appartient aux deux ensembles (est-ce qu'il est dans l'intersection A ∩ B ?). On va itérer ce processus de triage et vérification plusieurs fois. La proportion d'éléments tirés au hasard qui sont dans l'intersection est une estimation du coefficient de Jaccard entre les deux éléments.

Par ce que l'on appelle l'amplification probabiliste, on va alors obtenir un résultat très proche de la vraie valeur du coefficient de Jaccard en tirant très peu d'éléments dans l'union des deux ensembles de shingles.

En pratique, en tirant 200 shingles aléatoirement, si on en trouve 190 qui sont communs aux deux textes, alors on estimera que le coefficient de Jaccard vaut environ 0,95. Avec une telle valeur, on est dans un cas typique de contenu quasi-dupliqué.

Une question qu'on peut légitimement se poser est alors : est-ce qu'avec 200 tirages seulement on peut avoir confiance en la valeur qu'on a trouvé ? N'oubliez pas qu'un texte de bonne longueur va avoir des milliers de shingles avant de répondre… Vous avez répondu non ? Vous avez tort, avec 200 tirages on a une estimation à 0,1 près de la vraie valeur du coefficient de Jaccard, et ce avec une probabilité d'avoir raison de 99% !

Dans l'implémentation au sein d'un moteur, de nombreuses autres optimisations sont réalisées. Par exemple, les 200 shingles sont pré-tirés, ce qui permet de preprocesser chaque document pour être capable de filtrer le duplicate content plus rapidement. Mais en pratique, il existe maintenant des algorithmes plus performants, qui sont utilisés par les moteurs modernes (Monika Rauch Henzinger: Finding near-duplicate web pages: a large-scale evaluation of algorithms. SIGIR 2006: 284-291).

Conclusion

Il n'y a pas de grande conclusion à tirer de toute cette explication. On peut simplement dire que déterminer si un contenu est dupliqué est assez simple, et surtout qu'un algorithme comme celui décrit dans cet article à un paramètre qui détermine la « force » de filtrage. En effet, selon la valeur du seuil sur le coefficient de Jaccard, on sera plus ou moins strict sur ce que l'on considère être du contenu dupliqué.

***Sylvain Peyronnet**, Professeur des Universités à l'Université de Caen Basse-Normandie (http://sylvain.berbiqui.org/) et **Guillaume Peyronnet**, gérant de Nalrem Médias (http://www.gpeyronnet.fr/nalrem-medias.html). Ensemble, ils font des formations (http://www.peyronnet.eu/blog/masterclass-moteurs-seo/) et essaient de battre les loutres à la pêche à la truite.*