Explorer comment repenser la problématique des moteurs de recherche

Antoine BOURRY et Pierre LECAVELIER

2006-2007

Table des matières

Première partie
Introduction

Dans le cadre du projet annuel de notre troisième année de licence d'informatique à l'université de Caen, Jacques Vergne a proposé une étude sur le thème des moteurs de recherche : repenser leur problématique sans utiliser le concept de mot.

Ayant été créés par des anglophones, les moteurs de recherche traditionnels sont basés sur le concept de mot. L'utilisateur effectue une recherche d'un ou plusieurs mots, et recoit comme réponse un ensemble de documents. Ces résultats sont classés suivant un algorithme propre à chaque moteur.

Cette méthode fonctionne convenablement dans les langues dites à «mots». Cependant, sa limite à trouver un document pertinent est vite atteinte avec les langues qui pratiquent l'agrégation ou la déclinaison, mais aussi avec les langues à idéogrammes, comme le chinois, où la notion de mot est absente.

L'objectif de l'étude est donc de remettre en cause la validité de cette méthode et d'explorer de nouvelles techniques de recherche efficaces pour toutes les langues.

Pour réaliser notre étude, nous avons dans un premier temps, intégralement conçu et implémenté un moteur de recherche. La partie «moteur»a été réalisée en python et l'Interface Homme Machine en PHP. Une base de données MySQL permet de stocker les documents crawlés, les tables d'indexation et de mémoriser les requêtes effectuées.
Les méthodes implémentées sont la méthode par mots, par séquences répétées, par n-grammes et «hybride». Elle vous seront expliquées en détails dans la suite de ce document.

Vous pouvez retrouver l'IHM à l'adresse suivante :

`http ://pierre.crashdump.net/moteur_recherche/`

Deuxième partie

Implémentation d'un moteur de recherche

1 Le crawler pour collecter des documents

La première étape dans l'implémentation d'un moteur de recherche consiste à récupérer les documents qui seront inclus dans notre périmètre de recherche.

Pour y parvenir, nous avons mis en place ce qu'on appelle un **crawler**. Il nous permet de parcourir le web en simulant un comportement humain, de télécharger tous les documents qu'il rencontre, de les normaliser, les analyser puis de les sauvegarder.

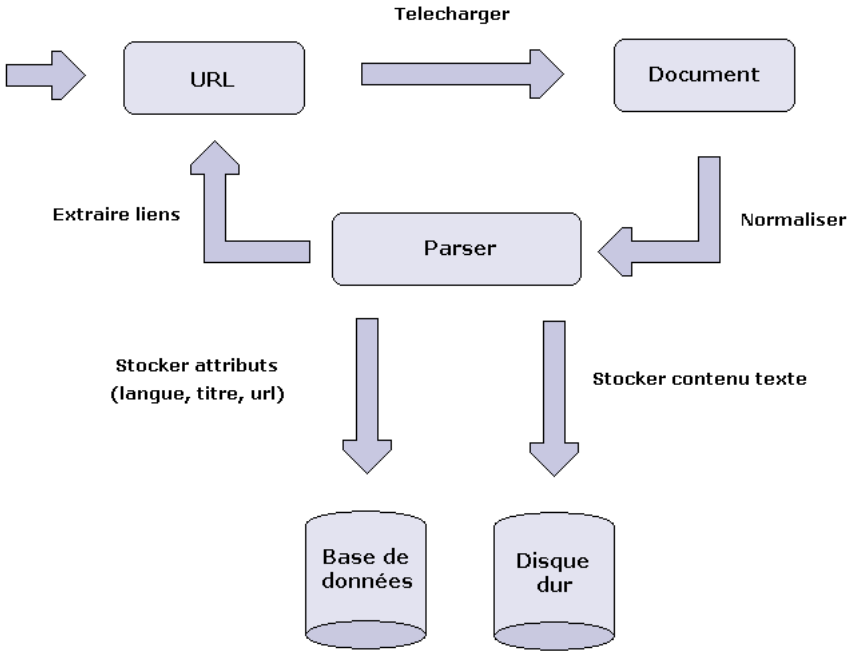

Fig. 1 – Schéma résumant le déroulement du crawler

Chaque document rencontré sera donc normalisé et encodé en UTF-8 (pour faire face à la multiplicité des langues) puis analysé pour en extraire :

- son titre
- son contenu textuel
- ses liens

Toutes les caractéristiques du document, c'est à dire son titre ainsi que son url, sa langue et son identifiant seront stockés en base de données.

Le contenu textuel du document sera lui sauvegardé sur le disque dur dans un fichier texte.

Et enfin, chaque lien sera ajouté à une file d'attente an attendant d'être lui-même visité par le crawler.

Remarque : Pour alimenter notre moteur de recherche, nous avons décidé de ne crawler que des articles de presse. Pour ce faire, nous avons ajouté un critère de sélection dans le choix des documents : Seules les url composées d'une série d'au moins 3 chiffres seront visitées, car un article de presse sur un site a forcément un identifiant qui se retrouve dans tous les cas dans son url.

Cela nous permet de nous abstraire dans la plupart des cas, des portails, des publicités etc... et donc de ne récupérer que des documents pertinents.

2 Indexation des documents

L'étape suivante est l'indexation des documents crawlés. Le but est de découper chaque document en tokens (suites de caractères) puis de les ajouter aux index stockés en base de données.

Nous avons mis en place 4 méthodes d'indexation que nous allons détaillé dans les parties suivantes.

2.1 Indexation par mots

Cette méthode est celle utilisée par les moteurs de recherche traditionnels. Le principe est simple, on remplace la ponctuation par des espaces puis on découpe le document en fonction de ces espaces.

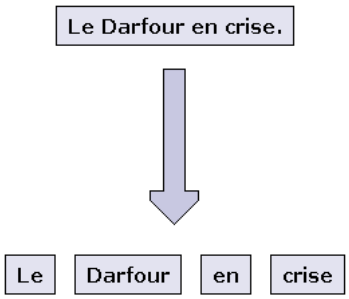

Fig. 2 – Exemple de découpage en mots

Le soucis avec cette méthode est qu'elle est inutilisable avec des langues à idéogrammes comme le chinois. Elle est également inefficace avec les mots composés que l'on peut trouver dans des langues comme l'allemand.

C'est sur cette problèmatique que repose tout l'intérêt de ce projet. Il a donc fallu réfléchir à plusieurs méthodes pouvant remplacer cette méthode traditionnelle d'indexation par mots.

2.2 Indexation par ngrammes

La première méthode de substitution est l'indexation par ngrammes. Le but est de découper chaque document en séquences de n caractères.

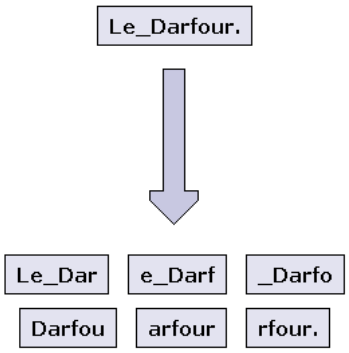

FIG. 3 – Exemple de découpage en 6grammes

Les tailles de ngrammes ont été choisies arbitrairement
- tokens de tailles 4, 5 et 6 pour les langues alphabétiques
- tokens de tailles 2, 3 et 4 pour les langues à idéogrammes

Remarque : Un choix arbitraire pour la taille des ngrammes n'est pas forcèment le plus pertinent. Une solution alternative pour chaque langue alphabétique aurait été de prendre plutôt la taille moyenne de ses mots. Même si nous avons pas eu le temps nécessaire pour évaluer cette alternative, cela reste une bonne piste à explorer.

2.3 Indexation par séquences répétées

Viens ensuite l'indexation par séquences répétées. Le principe est de découper toutes les séquences possibles de chaque document, puis de garder seulement celles qui apparaissent au moins x fois.

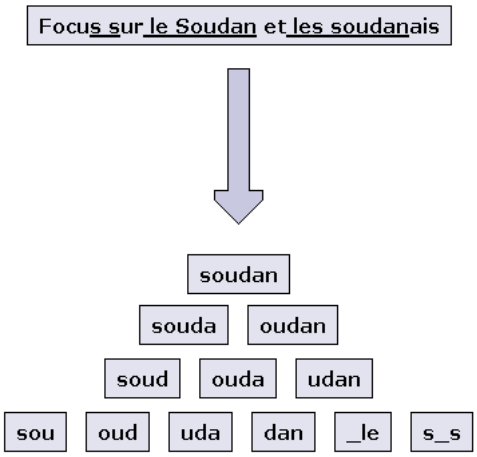

FIG. 4 – Exemple de découpage en séquences répétées avec $x = 2$

Notre index de séquences répétées a été alimenté avec le critère $x = 3$. Ce chiffre a été choisi suite à une série d'observations.

Il a également fallu imposer des tailles de séquences minimales et maximales. Car cela aurait été trop couteux au moment de l'indextion et de l'interrogation des index de prendre toutes les séquences possibles. Nous avons choisi de nous limiter aux séquences de 3 à 15 caractères pour les langues alphabétiques et de 2 à 10 caractères pour les langues à idéogrammes. Ces extrêmes restent raisonnables car les séquences en dehors de ces intervalles ne sont pas significatives.

Remarque : Plutôt que de choisir un critère x statique, il faudrait essayer de le faire varier en fonction de la taille du document (car un token a plus de chance d'apparaitre au moins 3 fois dans un document de 1000 caractères que dans un document de 100 caractères).

2.4 Indexation hybride

Voici la dernière méthode d'indexation. Nous l'avons nommé hybride car elle associe les indexations par mots et par ngrammes. C'est à dire que l'on fait d'abord un découpage par mots, puis un découpage par ngrammes de ces mots.

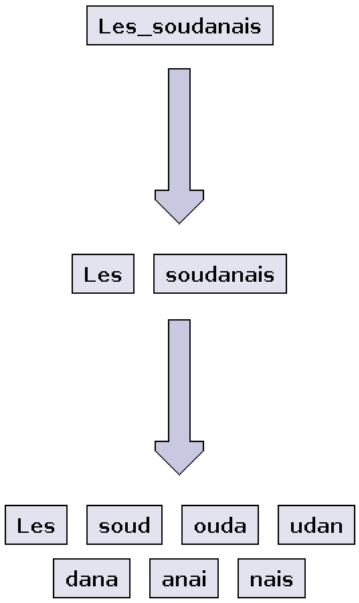

FIG. 5 – Exemple de découpage hybride

Nous avons choisi 4 comme taille de ngrammes par défaut, car elle est celle qui nous a semblé obtenir les meilleurs résultats suite à une courte série d'observations.

Mais pourquoi faire ce découpage en deux étapes ?

Nous avons pu remarquer que les ngrammes chevauchant deux mots pouvaient fausser les recherches. On peut le voir dans l'exemple précédent ou le 4grammes *es_s* n'est pas du tout significatif du terme recherché. C'est vrai qu'il est très courant de trouver un mot se terminant par *es* suivi d'un mot commencant par la lettre *s*.

Remarque : Il faudrait quand même faire une série de tests avec d'autres tailles de ngrammes pour obtenir la plus efficace, et pourquoi pas associer l'indexation par mots avec l'indexation par séquences répétées.

2.5 Structure des index

Comme nous l'avons expliqué dans la partie précédente, tous les index construits grâce aux différentes méthodes d'indexation sont stockés en base de données. Nous pouvons donc présenter leur structure avec l'exemple suivant :

Document 1 : le Soudan, le Darfour
Document 2 : le Soudan en crise

La structure de la table contenant l'index des mots sera sous la forme suivante :

Token	doc_occurences
le	1_2
Soudan	1_1
Darfour	1_1
le	2_1
Soudan	2_1
en	2_1
crise	2_1

On voit qu'à chaque token (les mots pour cet exemple) est associé le document ou il apparait ainsi que son nombre d'occurences.

2.6 Optimisation des index

Au moment d'interroger les index pour effectuer une recherche, le temps d'une requête est étroitement lié à la taille de l'index. Nous avons donc réfléchi à un moyen de réduire la taille des tables contenant ces index, ce qui nous permet d'optimiser le temps de recherche.

Pour se faire, nous avons décidé de fusionner toutes les lignes contenant le même token, et d'y rassembler tous les documents ou il apparait. On peut voir le résultat sur le schéma suivant :

Token	doc_occurences
le	1_2 2_1
Soudan	1_1 2_1
Darfour	1_1
en	2_1
crise	2_1

Cela nous permet bien d'optimiser notre moteur de recherche, comme nous le verrons dans la partie *Performances de l'application*

3 Interrogation des index

Pour interroger les index stockés en base de données, on fait tout d'abord un découpage de la requête.

Le découpage dépend de la méthode utilisée. Pour les recherches basées sur les méthodes hybride, par mots et par ngrammes, on applique le même découpage que pour l'indexation (voir la partie concernée). Mais pour une recherche basée sur les séquences répétées, on prend toutes les séquences possibles et pas seulement celles qui se répétent (car il est très rare que ce soit le cas dans une courte requête).

Puis on construit une requête SQL pour interroger la base de données. Cette requête applique un OU logique sur l'ensemble des tokens résultants du découpage.

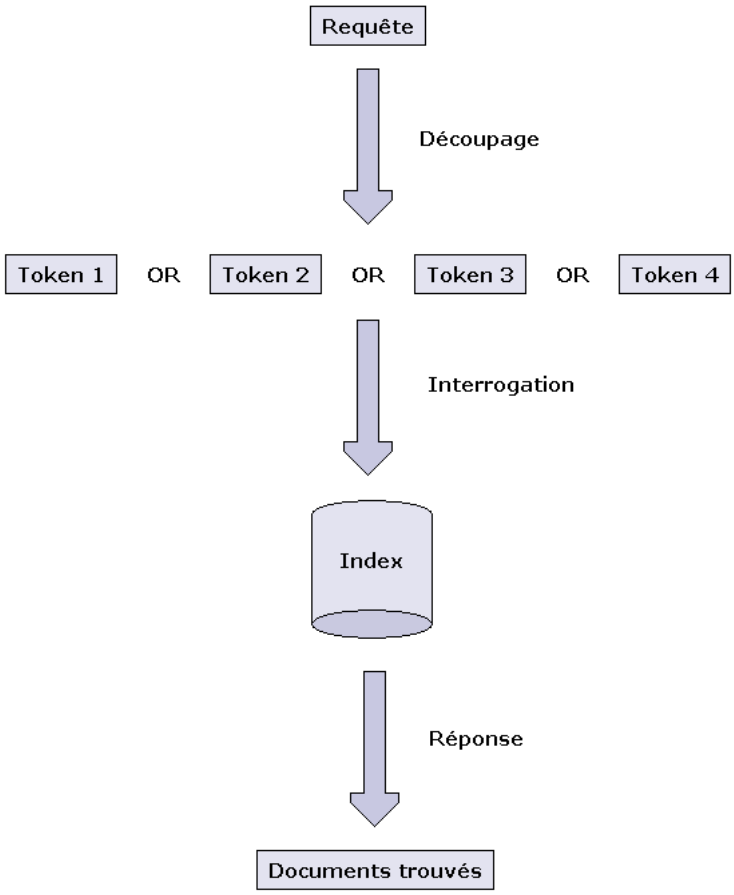

FIG. 6 – Interrogation des index stockés en base de données

On voit au final, que l'on obtient une série de documents qui répondent à la recherche effectuée par l'utilisateur. Mais cela n'est pas suffisant, car quand on sait qu'un moteur de recherche comme Google peut trouver des millions de documents suite à une requête. On conclue logiquement qu'un classement est nécessaire pour obtenir les documents les plus pertinents en tête de liste.

Nous allons voir les différents classements mis en place dans la partie suivante.

4 Classement des documents trouvés

4.1 Classement par la méthode classique

On fait un classement des documents en deux étapes. D'abord en fonction du nombre de tokens **différents** trouvés dans chacun d'eux. Puis quand il y a égalité, on classe ces documents en fonction du **nombre total** de tokens trouvés.

Pour être plus explicite, nous allons prendre un exemple concret avec un classement de trois documents suite à une recherche par mots :

- **Recherche :** match contre l'Argentine

- **Document 1 :** L'argentine est un pays d'Amérique du sud
- **Document 2 :** Le match face à l'Argentine sera un match intéressant
- **Document 3 :** Le match face à l'Argentine sera disputé

On obtiendra un premier classement :

- **Document 2** avec 2 points (match et Argentine)
- **Document 3** avec 2 points (match et Argentine)
- **Document 1** avec 1 point (Argentine)

On voit qu'il y a égalité entre les documents 2 et 3 donc on les classe maintenant en fonction du nombre total de tokens trouvés, pour avoir finalement :

- **Document 2** avec 3 points (3 tokens trouves)
- **Document 3** avec 2 points (2 tokens trouves)
- **Document 1** (classement précédent)

4.2 Classement par la méthode de Grefenstette

Ce type de classement est basé sur les recherches sur l'identification des langues de l'allemand Grefenstette.

Il s'agit dans un premier temps d'attribuer un score à chaque token résultant du découpage de la requête. Ce score correspond à la division du nombre d'occurences dans le document divisé par le nombre total de tokens composant ce même document.

Reprenons l'exemple précédent :

- **Document 1 :** On aura 1/7 pour le mot *Argentine* car il est présent une fois dans un document de 7 caractères. Puis 10^{-6} pour les mots *match* et *contre* qui n'apparaissent pas dans le document.

- **Document 2 :** Le mot *Argentine* aura 2/8 car il apparait 2 fois dans un document de 8 caractères. Le score du mot *match* sera lui de 1/8 et finalement, on aura 10^{-6} pour le mot *contre*.

- **Document 3 :** Le score des mots *match* et *Argentine* sera de 1/6 et le score du mot *contre* sera de 10^{-6}.

Remarque : Pour l'exemple précédent, on ne considère pas les mots composés d'un unique caractère.

Remarque : On voit que l'on choisi un score par défaut pour tout token n'apparaissant pas dans le document. Ce score correspond au nombre de chiffres significatifs (que l'on a pris à 6 pour notre application).

L'astuce consiste finalement à faire le produit de tout ces nombres pour avoir un score pour chaque document. Le classement se fera à partir de ce score et sera le suivant pour notre exemple :

- **Document 2 :** Il aura comme score $2/8 * 1/8 * 10^{-6} = 3.125 * 10^{-8}$
- **Document 3 :** Son score sera de $1/6 * 1/6 * 10^{-6} = 2.777778 * 10^{-8}$
- **Document 1 :** Finalement, le score de ce document sera $1/7 * 10^{-6} * 10^{-6} = 1.428571 * 10^{-13}$

On voit que le classement reste le même qu'avec la méthode précédente, mais cela peut varier avec d'autres exemples. La différence majeure entre les deux classements présentés est que ce dernier prend en compte la taille du document.

Troisième partie

Performances de l'application

5 Temps de collecte et d'indexation

On voit sur le diagramme suivant la durée du processus complet, de la collecte (crawling) de documents à la réduction des index.

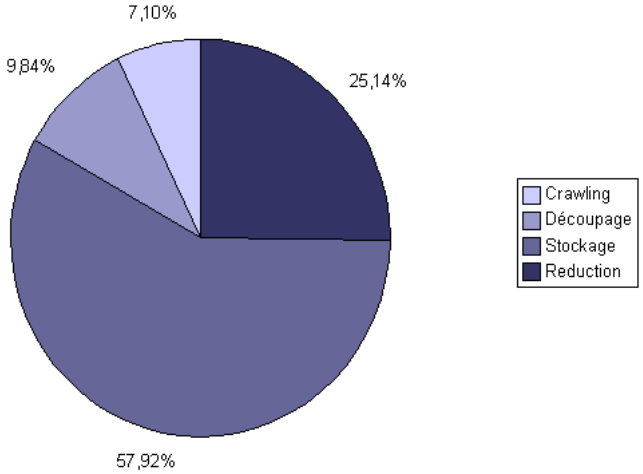

Fig. 7 – Collecte et indexation de 3000 documents en 23 heures et 30 minutes

On remarque que la collecte est beaucoup plus rapide que l'indexation des documents. Dans cette indexation, on distingue les étapes de découpage, de stockage en base de données et enfin de réduction des index. Cette dernière n'est pas indispensable mais très utile pour accélerer l'interrogation.

6 Taille des index

Sur ce diagramme, on peut voir la taille des index en nombre d'enregistrements avant et après réduction.

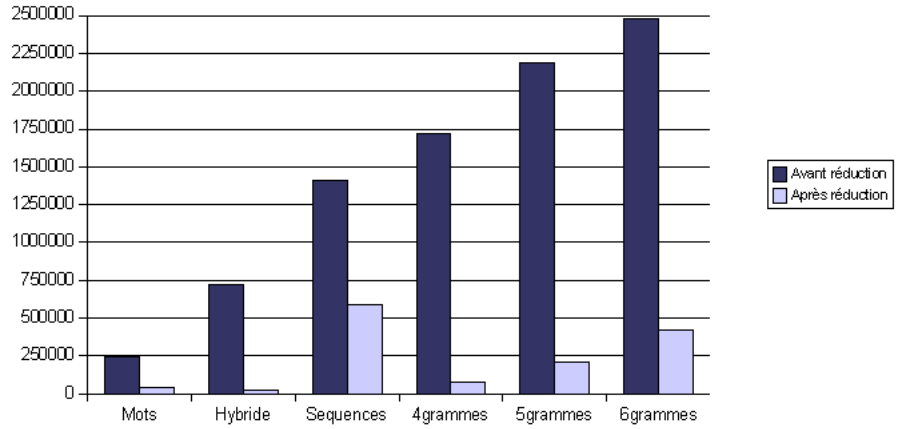

FIG. 8 – Taille des index en nombre d'enregistrements

On remarque sur ce diagramme que l'index des ngrammes augmente linéairement avec la taille du ngramme, ce qui est logique puisque plus une séquence est courte, plus elle a de chance de se répéter dans un même document. On aura donc des tokens avec plus d'occurences, mais au final moins de lignes dans la table contenant l'index.

Pour les séquences répétées, comme toutes les séquences sont possibles (dans les limites de l'intervalle) mais qu'elles doivent avoir un certain nombre d'occurences pour être stocké, l'index reste de taille raisonnable.

Quand à la méthode hybride, comme on filtre seulement les 4grammes qui se trouvent au sein d'un mot, on obtient forcément un index plus petit que celui des 4grammes et plus grand que celui des mots.

Finalement, pour l'index des mots, on constate simplement qu'il reste le plus petit en nombre d'enregistrements. Sûrement parcequ'un mot se répète plus souvent qu'une série de caractères, mais aussi parceque sa taille moyenne reste assez élevée.

Quand on observe la réduction des index, on constate que le nombre d'enregistrements diminue fortement. On peut maintenant présenter un diagramme présentant le taux de réduction pour chaque méthode.

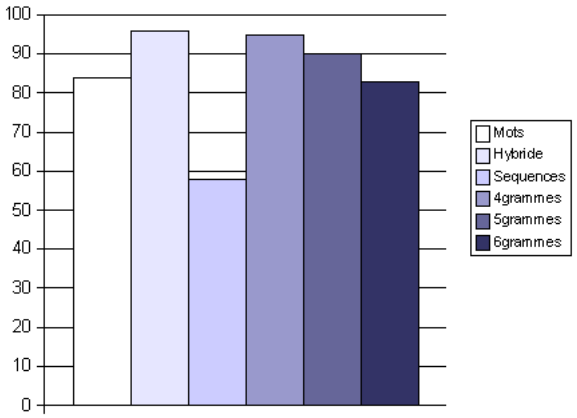

Fig. 9 – Taux de réduction (en %)

Comme précédemment, on voit qu'il y a une évolution linéaire du taux de réduction pour les ngrammes. Pour la même raison, un 4grammes a plus de chances de se retrouver à travers plusieurs documents qu'un 6grammes, donc le taux de réduction augmente logiquement.

Le taux de réduction des séquences est forcément le moins élevé, tout simplement parceque la probabilité de retrouver une séquence de 15 caractères dans plusieurs documents est très faible.

Pour les hybrides, le taux de réduction dépend de la taille de ngrammes choisie. Dans ce cas, comme on a choisi 4 comme taille, on voit que le taux de réduction se rapproche fortement de celui des 4grammes.

Et enfin, un mot a beaucoup de chances de se retrouver dans différents documents, ce qui explique un taux de réduction relativement fort.

7 Temps d'interrogation des index

Sur le diagramme suivant, nous présentons le temps moyen de toutes nos requêtes effectuées.

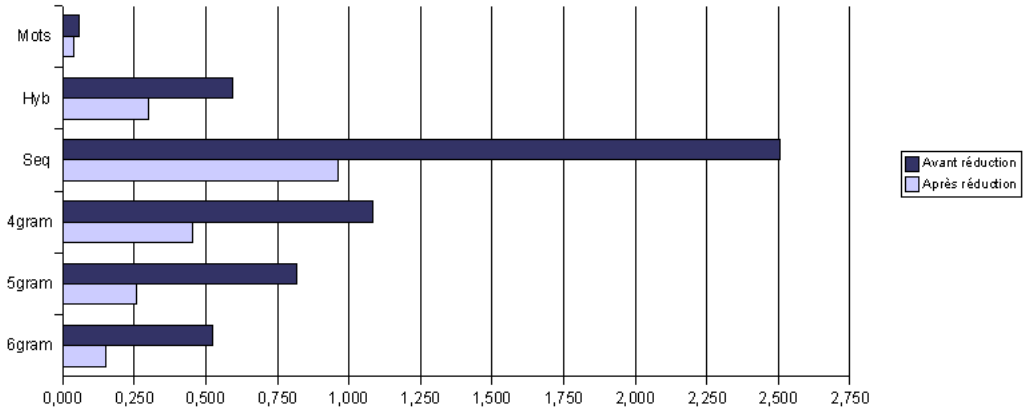

FIG. 10 – Temps moyen de nos requêtes

Deux facteurs expliquent le temps d'interrogation des index :
– La taille des index
– Le nombre de tokens à rechercher

La recherche par mots est la moins couteuse en temps, tout simplement parcequ'elle associe un index petit et un faible nombres de mots (en général) à rechercher dans la base de données.

Au contraire, l'interrogation par séquences répétées associe elle un index très grand et un grand nombre de tokens à rechercher (découpage de la requête pour obtenir toutes les séquences possibles).

Entre les deux, on trouve un temps moyen pour interroger les index de ngrammes et qui encore une fois, diminue linéairement avec la taille du ngramme. Cela s'explique surtout par le nombre de tokens à rechercher qui diminue quand la taille du ngramme augmente.

Quand aux hybrides, le temps de recherche est également fortement influencé par le nombre de tokens à rechercher. Si la requête est composé de mots courts, dans ce cas cela revient à interroger avec la méthode par mots, par contre, si l'on fait une recherche composée de mots longs, on se retrouvera avec beaucoup de 4grammes à chercher dans l'index.

Enfin, on constate que ce diagramme confirme bien le gain de temps apporté par la réduction des index.

Voici un dernier diagramme qui présente un intérêt secondaire de la réduction des index. Nous avons recherché dix fois une expression courante et dix fois une expression rare, pour obtenir une moyenne de temps, et voici ce que l'on obtient :

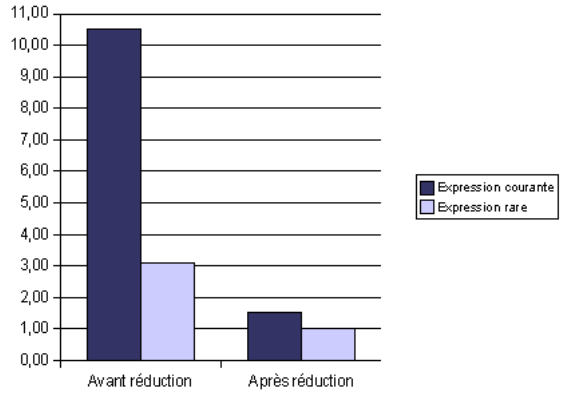

FIG. 11 – Recherche d'une expression courante et d'une rare

On voit que la différence de temps entre une recherche courante et une expression rare est beaucoup plus petite après réduction. Cela s'explique simplement qu'après réduction, un token, qu'il soit rare ou courant, a le même nombre d'enregistrement dans la base de données... Un seul et unique. Et comme une opération de découpage d'une chaîne (celle contenant tous les documents ou apparait le token) est moins couteuse qu'une opération de recherche en base de données de plusieurs fois le même token, cela se répercute directement sur le temps de recherche.

Pour conclure, en plus de diminuer le temps de recherche, la réduction des index réduit également l'influence des expressions courantes.

Quatrième partie
Expérimentation des différentes méthodes

8 Protocole expérimental

Un moteur de recherche est un outil de l'internet, et a donc une dimension universelle. C'est pourquoi le moteur de recherche ne doit pas se heurter à la barrière de la langue. Le nombre de langues a été limité à deux, le français et l'allemand. Elles ont été choisies pour leurs caractéristiques spécifiques.

1000 articles de presse par langue ont été collectés et ajoutés à notre périmètre de recherche.

Nous avons travaillé avec les quatres méthodes d'indexation et de recherche explicitées auparavant ainsi qu'avec les deux classements mis en place.

Pour évaluer ces différentes méthodes et l'influence des types de classements, on prendra la pertinence comme seul critère d'efficacité.

8.1 Le choix des langues

Le Français est une langue romane originaire de la France. Elle a été la première langue que nous avons choisi. En effet, c'est notre langue maternelle, et de ce fait elle constitue un point de repère important dans notre étude. L'analyse sera facilitée car nous maîtrisons son étymologie, sa grammaire et sa sémantique.

Étymologie
La majorité du fond lexical français provient du latin (en tant que langue-mère) ou bien est construit à partir des racines gréco-latines. De nombreux termes possèdent un synonyme, l'un venant de la racine latine ancienne, l'autre étant populaire. Ces doublets sont surtout présents avec un nom (populaire) et l'adjectif dérivé (savant) : mère/maternel, frère/fraternel, cheveu/capillaire, foi/fidèle, froid/frigide, œil/oculaire, sûreté/sécurité, ...
Le francique, en tant que superstrat, a laissé quelques mots importants et les emprunts sont nombreux : d'abord à l'anglais, puis à l'italien, aux autres langues romanes, aux langues germaniques tels que l'allemand ou le néerlandais, ...

Grammaire
Une des caractéristiques de la grammaire française vis à vis de nombreuses langues vivantes est la richesse de ses temps et modes. On peut toutefois noter que cette richesse tend à se réduire à l'oral.

Sémantique
La langue française a une sémantique très riche. Elle se prête à des jeux de mots, des traits d'esprits, des devinettes, des contrepèteries...
Cette caractéristique est importante pour le traitement de l'information dans les bases de données.

L'Allemand est une langue intéressante pour l'étude de notre moteur de recherche. Son écriture et sa grammaire sont différentes de celle du français. Elle nous a été beaucoup plus difficile à étudier que le Français car nous ne la maitrisons pas.

Ecriture

L'allemand s'écrit avec les 26 lettres de l'alphabet latin, trois voyelles surmontées d'un Umlaut (sorte de tréma) ä, ö et ü, et un symbole graphique spécial ß, eszett (fusion de « s » et de « z », ou de deux « s ») ou scharfes S, utilisé en lieu et place de «ss»dans certains cas (principalement après une voyelle longue ou une diphtongue[1]).

Grammaire

L'allemand est une langue flexionnelle comportant des conjugaisons et des déclinaisons

Le principe de la conjugaison allemande est assez proche du principe de la conjugaison française. Les différences notables sont : l'existence du subjonctif, la différence entre passif-action (Das Haus wird gebaut, la maison est [en train d'être] construite) et passif-état (Das Haus ist gebaut, la maison est construite [elle est finie]).

La déclinaison allemande comporte quatre cas, le nominatif, l'accusatif, le datif et le génitif, auxquels s'ajoutent trois genres, le masculin, le féminin et le neutre ainsi que deux nombres, le singulier et le pluriel.

Les déclinaisons sont employées :

- pour indiquer la fonction d'un groupe nominal : sujet, objet, complément d'attribution, ou complément du nom,
- après certaines prépositions pour indiquer s'il y a changement de lieu ou non,
- en fonction du cas exigé par les prépositions ainsi que par les verbes, les adjectifs ou les noms suivis d'un complément.

L'allemand a pour particularité syntaxique principale de placer des éléments importants, soit en première position dans la phrase, soit en dernière position.

[1] Une diphtongue, par opposition à la monophtongue, est un son voyelle constitué de deux voyelles : la réalisation se fait en passant de l'une à l'autre. Dans certaines langues, dont l'anglais, on trouve également des triphtongues dont la qualité varie deux fois.

8.2 La pertinence, unique critère d'évaluation

La facilité avec laquelle un utilisateur va trouver un document recherché, permet de juger de la qualité d'un moteur de recherche. Des requêtes différentes doivent pouvoir être utilisées pour trouver un même document en fonction des individus. Ce document ciblé par l'utilisateur doit en plus se retrouver parmi les premiers résultats, au risque de se retrouver noyé dans la masse.

Le bruit[2] et le silence[3] n'ont pas été étudiés car ils se font plus ou moins ressentir de manière indirecte dans la pertinence.

La logique nous fait penser que plus le bruit est important moins la pertinence sera bonne. Dans notre étude, ce n'est pas tout à fait le cas car on observe seulement les premiers résultats (comme à tendance à le faire un utilisateur classique). Etant donné la quantité des documents disponibles sur l'internet, il est normal d'obtenir un bruit élevé. Cependant, ce qui va importer, c'est la qualité du classement des résultats. En effet, si tous les documents «bruyants» se situent vers la fin des résultats, le bruit ne devient plus nocif et l'utilisateur ne sera pas perturbé.

Le silence est lui plus problématique. Si une page web qui pourrait convenir à l'utilisateur ne figure pas parmis les résultats alors la pertinence est moins bonne. Ou alors si cette page n'est pas dans les premiers résultats, on peut parler de silence car le document est noyé dans la masse. Le silence fait donc chuter la pertinence.

La difficulté à chiffrer le bruit et le silence nous a incité à ne pas les étudier. En effet, chiffrer de manière objective le bruit et le silence n'est pas possible. Pour les mesurer, seule la manière comparative est envisageable. C'est à dire qu'il faut prendre une méthode de référence et comparer toutes les autres méthodes à celle-ci. Le silence correspondrait aux résultats qui sont présents dans la méthode de référence mais absent dans la méthode étudiée et vice et versa pour le bruit.

Ce type d'étude avait été commencé mais il a été stoppé. En effet, le bruit et le silence comparatifs n'ont rien à voir avec ceux théoriques. Il vont témoigner de la différence des résultats par rapport à la méthode de référence. Ce genre d'étude aurait pu être intéressant si les méthodes avaient été comparées deux à deux entre elles. Par exemple, deux méthodes différentes considérées aussi pertinentes l'une comme l'autre peuvent avoir des résultats complètement différents. Ainsi, la différence des résultats entre deux méthodes pour une même requête aurait été observée.

8.3 Chiffrement de nos requêtes

Avant de vérifier nos méthodes, nous avons défini pour chaque langue dix requêtes différentes. A ces requêtes sont fixés des résultats attendus. C'est l'absence ou la présence de ces résultats qui jugeront de la pertinence.

La pertinence d'une méthode sera évaluée sur ces dix requêtes. Pour chaque requête, seulement les dix premiers résultats sont analysés. Si le résultat était attendu alors la note de la requête en cours d'évaluation augmente d'un point[4]. Si une requête comporte moins de dix résultats[5] les résultats manquants équivalent à des résultats non-pertinents. Une requête obtient donc une note comprise entre 0 et 10.

Finalement, la note de la méthode est établie en effectuant la moyenne des notes des dix requêtes.

[2] le bruit est la présence d'un document inutile pour l'utilisateur dans les résultats affichés par le moteur de recherche suite à une requête

[3] le silence est l'absence d'un document attendu par l'utilisateur dans les résultats affichés par le moteur de recherche suite à une requête

[4] La note initiale d'une requête est 0

[5] Les requêtes ont été choisies dans l'idée d'avoir plus d'une dizaine de résultats

8.4 Les problèmes posés

L'analyse de la pertinence est assez subjective car les requêtes attendues ont été fixées arbitrairement. Tout le monde ne recherche pas une page web de la même façon. Une solution possible serait d'effectuer un sondage auprès d'un public diversifié mais nous n'avons malheureusement pas eu le temps de le faire car sa mise en place n'est pas évidente et demande un investissement important de la personne sondée.

Une recherche peut être menée de différentes façons. Soit un document particulier est recherché, par exemple l'utilisateur veut retrouver une page qu'il avait déjà visité, soit l'utilisateur a besoin de plusieurs sources d'informations, par exemple il souhaite trouver et comparer plusieurs chambres d'hôtes pour son prochain voyage.

Le protocole d'étude que nous avons mis en place est adapté au deuxième cas car la plupart du temps, quand un utilisateur fait une recherche, il s'attend a trouver plusieurs documents intéressants. Une autre manière d'étudier la pertinence des méthodes aurait été de prendre un document crawlé et de taper une requête adaptée pour retrouver ce document. Le système de notation est bien moins coûteux que celui de notre protocole mais moins précis.

Dans cette partie, il serait intéressant d'observer quelques exemples précis de résultats qui caractérisent la méthode de recherche.

9 Etude de la pertinence

9.1 En francais

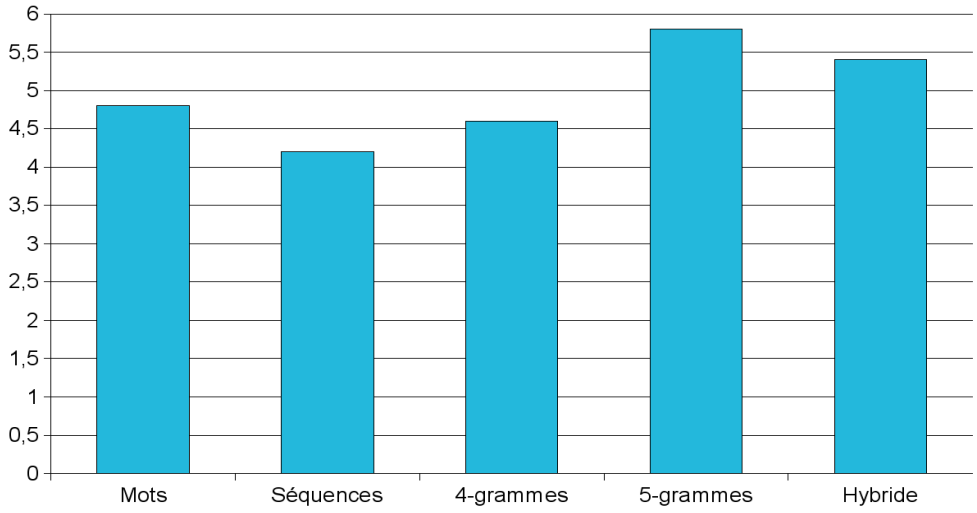

FIG. 12 – Pertinence avec un classement classique - Français

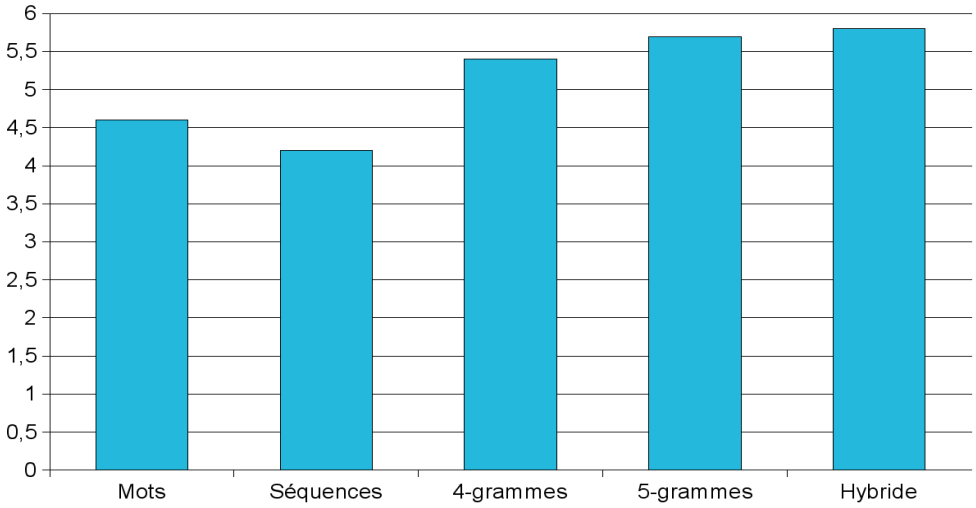

FIG. 13 – Pertinence avec un classement Grefenstette - Français

Ces deux histogrammes en bâtons représentent les notes de la pertinence des différentes méthodes pour un type de classement. L'axe des ordonnées indique la valeur de la note et chaque bâton est associé à une méthode.

L'histogramme de la figure 12 étudie la pertinence avec un classement classique. Il montre que la méthode par séquences semble la moins pertinente, ensuite la méthode par 4-grammes, puis par mots, hybride et la méthode la plus efficace est celle par 5-grammes. L'écart entre les différentes méthodes est faible. La note minimum est de 4.2 et la maximum de 5.8.

Avec un classement grefenstette (figure 13), l'ordre de pertinence est différent, par ordre croissant on a : séquences, mots, 4-grammes, 5-grammes, hybride. L'écart entre les différentes méthodes est également très faible. La note minimum est de 4.2 (méthode par séquence) et la maximum de 5.8 (méthode hybride).

Dans les deux classements, le minimum et le maximum sont identiques. En revanche, le classement change la pertinence d'une méthode. Certaines deviennent plus pertinentes comme les méthodes par mots, 4-grammes et hybride lors du passage du classement classique au classement grefenstette, tandis que la pertinence de la méthode par 5-grammes baisse. De façon générale le classement grefenstette tend à augmenter sensiblement la pertinence des méthodes. La méthode par 5-grammes ne baisse que très peu, elle passe de 5,8 à 5,7. La pertinence par 4-grammes est celle qui a le mieux progressé en passant au classement grefenstette.

Les deux méthodes qui ressortent comme étant les plus pertinentes sont la méthode par 5-grammes (moyenne de 5,75) puis l'hybride (moyenne de 5,6) quelque soit le type de classement. Les moins pertinentes sont la méthode par séquences avec une moyenne de 4,2 puis par mot avec une moyenne de 4,7. La meilleure note est de 5.8, elle est obtenue à la fois dans la méthode par 5-grammes avec un classement classique et dans la méthode hybride avec un classement grefenstette.

Les couples retenus comme les plus pertinents sont d'abord hybride-grefenstette et 5-grammes-grefenstette à égalité avec le couple 5-grammes-classique.

9.2 En Allemand

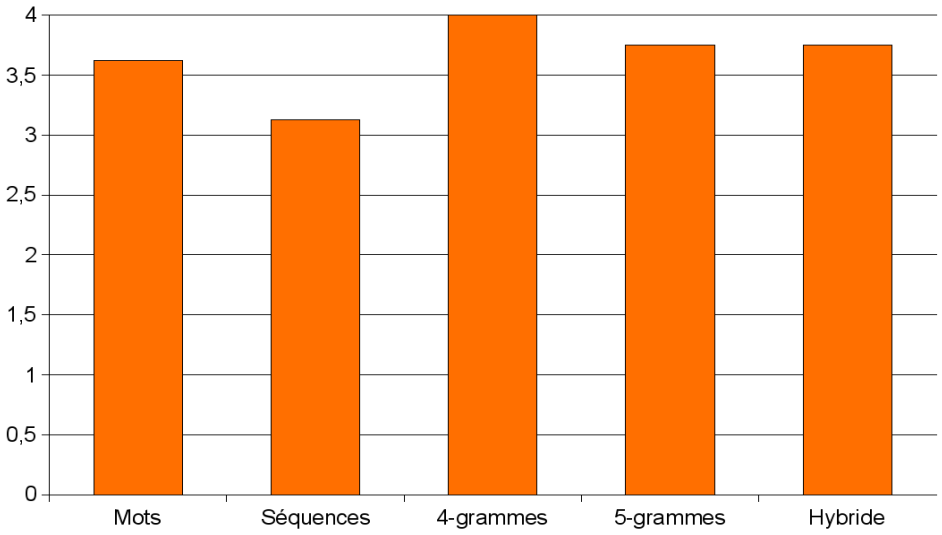

FIG. 14 – Pertinence avec un classement classique - Allemand

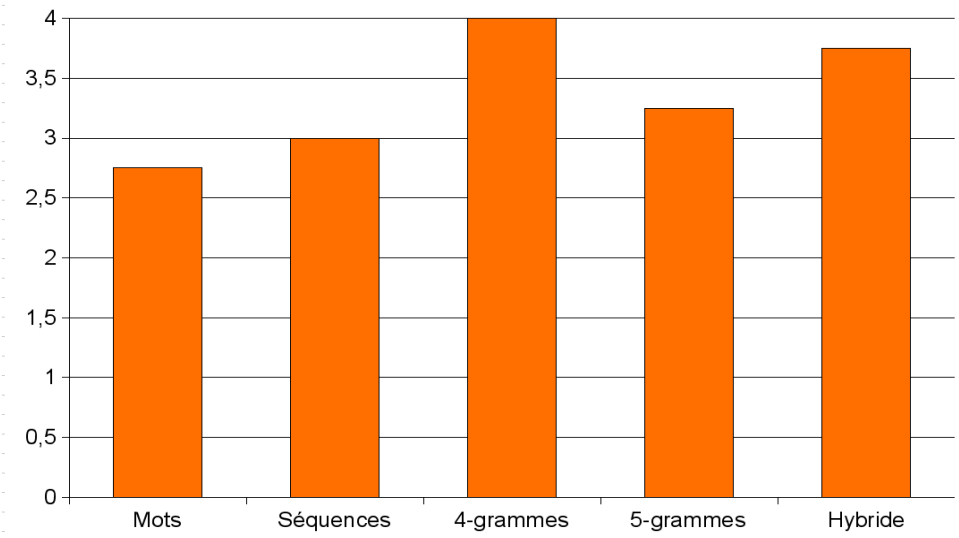

FIG. 15 – Pertinence avec un classement Grefenstette - Allemand

Les histogrammes en bâtons des figures 14 et 15 sont organisés de la même façon que pour le français.

Dans la figure 14, la pertinence avec un classement classique est étudiée. La pertinence des méthodes par ordre croissant est : séquences, mots, 5-grammes, hybride et 4-grammes. L'écart entre les résultats des différentes méthodes est très faible. Les notes vont de 3,13 pour la méthode par séquences à 4,0 pour la méthode par 4-grammes. La moyenne est de 3,65.

Dans la figure 15, on étudie la pertinence avec un classement grefenstette. L'écart entre les notes est plus important qu'auparavant. La méthode la plus pertinente est la méthode par 4-grammes (note : 4,0) puis la méthode hybride, 5-grammes et la moins pertinente la méthode par mots (note : 2,75). La moyenne est de 3,35.

Les deux types de classement ont une forte influence sur les résultats qui ont tendance à baisser en passant au classement grefenstette et particulièrement pour la méthode par mots. En revanche, les résultats de la méthode hybride et par 4-grammes n'ont strictement pas évolué.

En tenant compte des deux classements, la méthode la moins pertinente est la méthode par séquences suivie de la méthode par mots ; les méthodes par 4-grammes et hybride sont les plus pertinentes. Le meilleur couple méthode–classement est la méthode par 4-grammes avec une note de 4,0 puis hybride avec une note de 3,75 et il y a une curieuse indifférence entre le classement classique et grefenstette.

Le classement par grefenstette est donc globalement moins bon que le classement classique car il n'augmente pas la pertinence quelque soit la méthode utilisée. Mais sur les méthodes qui ressortent comme les plus pertinentes, le classement n'a pas d'effet. Donc la méthode par 4-grammes avec n'importe quel classement est la meilleure. Les couples avec la méthode hybride sont également très satisfaisants.

9.3 Bilan

Les chiffres montrent clairement un comportement différent en fonction des méthodes utilisées.

La méthode par mots est la seule des méthodes qui a retourné moins de 10 documents pour certaines requêtes. Elle engendre plus facilement du silence car elle est beaucoup plus stricte. Faire une recherche par mots oblige l'utilisateur à fournir les termes exacts, au risque de ne pas récupérer les documents ciblés.

L'utilisateur a une forte influence sur sa recherche avec cette méthode. Elle lui demande une compréhension assez élevée sur le fonctionnement d'un moteur de recherche car la requête agit plus «directement»qu'avec les autres méthodes.

Il suffit par exemple que l'utilisateur ait mal orthographié un mot pour que la recherche n'aboutisse pas. Par exemple, un utilisateur tape «apartement»au lieu de «appartement», aucun résultat n'est trouvé avec la méthode par mots tandis que toutes les autres méthodes trouvent un résultat très pertinent. De la même façon, pour le français, si l'utilisateur met ou non au pluriel, l'ajout du «s»aura des répercussions très importantes.

De plus, l'allemand est une langue qui fonctionne avec des déclinaisons et des agrégations[6]. L'utilisateur va donc décliner les mots de sa requête comme il pense être le plus adapté. Mais si ce mot est présent dans un document sous une autre déclinaison, la méthode par mots ne le trouve pas. L'agrégation conduit au même genre de problème. Il y a donc plusieurs facteurs non-négligeables qui sont source de silence avec la méthode par mots.

On perd également toutes notions de recherche à thème. Par exemple, avec la requête «conflit Palestine», les occurrences sur le thème de la Palestine sont importantes. Par exemple, le terme «conflit palestinien»est ignoré avec la méthode par mots. La méthode obtient une moyenne de 2 sur les deux classements, alors que les autres méthodes ont toutes des notes bien supérieures (autour de 6) pour cet exemple. Sur notre étude, on constate effectivement que la méthode par mots bien que correcte est loin d'être la plus satisfaisante.

Quelque soit la langue, les résultats obtenus avec la méthode par séquences sont souvent bien différents des résultats trouvés avec les autres méthodes. Parfois, cette méthode arrive à «remonter»des documents pertinents qui ne sont pas perçus par les autres méthodes. Mais, ce phénomène est assez aléatoire et les documents pertinents qui sont trouvés par les autres méthodes sont assez souvent perdu dans le bruit que provoque cette méthode. Au cours de notre étude, c'est la méthode qui s'est révélée la moins efficace malgré certains atouts.

Les méthodes par n-grammes se sont révélées plutôt efficaces. Elles sont toujours dans les meilleurs résultats. En recherchant «windsurf», on trouve le document 1 003 qui évoque le surf et le kite-surf. Même si le document n'est pas réellement pertinent, cette requête nous permet de constater de l'efficacité de la méthode par ngrammes. De même, en allemand, avec la requête «Google kauft»(google achète), on retrouve le verbe conjugué ou transcrit en mot (achat). Elle semble bien faire le compromis entre la recherche par mots et la recherche par séquences. Le grammage est sensible à la qualité des résultats. Indifféremment du classement, la méthode par 5grammes est plus efficace en français alors que c'est la méthode par 4grammes qui l'emporte en allemand. Le gramme choisit doit donc être fonction de la langue (peut être en fonction de la taille moyenne de ses mots).

La pertinence de la méthode hybride est proche de la méthode par ngrammes. Dans les deux langues, les deux meilleurs méthodes sont toujours l'une des deux, la méthode par ngrammes ou la méthode hybride. Dans le détail, la méthode hybride se rapproche de la méthode par ngrammes mais en moins aléatoire. C'est la méthode la plus fiable.

Pour le francais on a vu que la méthode par 5grammes était plus efficace que la méthode par 4grammes. On pourrait donc essayer pour cette langue de modifier la méthode hybride en associant la méthode par mots et celle par 5grammes (plutôt que 4grammes actuellement).

[6] plusieurs mots sont collés pour n'en former qu'un

Le classement joue un rôle important dans les résultats. L'exemple le plus flagrant est celui de «google kauft»avec la méthode par 5-grammes. Avec le classement classique on trouve 5 documents pertinents et avec le classement grefenstette on n'en trouve aucun. Cependant cet exemple n'est pas forcément révélateur car pour les méthodes les plus pertinentes, le classement par grefenstette est au moins aussi efficace que le classement classique. Si on regarde toutes les méthodes, en français le classement par grefenstette apporte de meilleurs résultats alors qu'en allemand c'est l'inverse. Il est donc difficile d'aboutir à des conclusions pour le type de classement le plus abouti.

Cinquième partie
Conclusion

Actuellement, les moteurs de recherches utilisent pour la plupart la méthode par mots. L'objectif de notre étude était de voir si les méthodes actuelles sont les meilleures et les plus universelles. Dans le bilan de notre étude, il est clair que la méthode par mots n'est pas la plus efficace et qu'elle n'est pas adaptée à toutes les langues même à une langue alphabétique comme l'allemand. C'est la méthode la moins pertinente en allemand. L'étude remet donc fortement en question les méthodes utilisées actuellement et leurs côtés primitifs.

Et qu'en est-il des langues à idéogrammes ? Nous avons préféré nous concentrer sur les langues alphabétiques, mais tout porte à croire que ces méthodes utilisant le grain caractère plutôt que le grain mot sont indispensables pour ce type de langue.

Le classement par grefenstette semble légèrement plus intéressant mais il n'a pas d'effets miraculeux sur les méthodes. On aurait pu croire à un changement radical des résultats sur au moins une méthode mais ce n'est pas le cas, du moins pas sur les langues étudiées.

L'étude, que nous avons mené, est assez ambiguë. Le nombre de documents crawlés ont limité les possibilités de requêtes. La méconnaissance des langues étrangères a été une barrière et pose des limites. La quantité faramineuse des documents disponibles sur l'internet et les diverses possibilités de recherche remettent en cause notre étude. La fiabilité de notre échantillon est donc remis en question devant l'immensité de la toile. L'étude soulève des questions qui restent en suspens et doit donc être approfondie avec des moyens plus conséquents.