Indexation de blocs extraits de pages Web en utilisant le rendu visuel

Nicolas Faessel

Laboratoire des Sciences de l'Information et des Systèmes (LSIS, UMR CNRS 6168)
Domaine universitaire Saint-Jérôme
Avenue Escadrille Normandie-Niemen
13397 Marseille Cedex 20

nicolas.faessel@lsis.org

RÉSUMÉ. *Cet article présente un modèle d'indexation de pages Web basé sur leur rendu visuel. Dans ce modèle, une page Web n'est plus considérée comme un tout, mais comme la combinaison d'un ensemble de blocs dont chacun porte sa sémantique propre. L'indexation d'une page Web est réalisée en deux étapes : (1) construction d'un arbre hiérarchique de blocs visuels, en s'appuyant sur la disposition visuelle des blocs de la page (2) indexation textuelle de chaque bloc par un vecteur de termes et tenant compte de l'importance de ces blocs et de l'indexation des blocs contenants, contenus ou voisins.*

ABSTRACT. *This paper presents a Web page indexation model. In this model, a Web page is not viewed as a whole, but as a combination of a set of blocks based on their visual rendering, where each bloc shares is own semantic. The indexation of a page Web is achieved in two steps : (1) construction of a hierarchical tree of visual blocks based on block visual layout in the Web page (2) textual indexation of each block by a term vector and taking into account blocks importance and indexation of neighbouring blocks (parent, children, siblings...).*

MOTS-CLÉS : *Recherche d'information, Indexation, Modèle Vectoriel, Segmentation de page Web*

KEYWORDS: *Information Retrieval, Indexation, Vector Space Model, Web Page Segmentation*

1. Introduction

Depuis son apparition, le Web a largement évolué en terme de contenus. Les premières pages Web étaient essentiellement composées de textes et de liens hypertextuels. Les pages Web actuelles ont une architecture structurelle et visuelle complexe et contiennent aussi bien du texte que des images, du son et de la vidéo. Les moteurs de recherche actuels sur le Web indexent le contenu textuel des pages Web en les considérant comme un tout. Or, ces pages contiennent en général plusieurs éléments d'information, qui ne sont pas forcément corrélés entre eux. De plus, certains de ces éléments ne sont pas pertinents pour l'utilisateur par rapport à une requête donnée, car ils ont plutôt un rôle de navigation, de formulaires de saisie ou de publicité. La prise en compte d'une page Web comme un ensemble d'unités sémantiques (*blocs*), pose le problème de sa décomposition en de telles unités. Une page Web est généralement représentée comme un document HTML, XHTML ou XML, combiné avec une feuille de style (CSS[1] par exemple), qui décrivent sa structuration logique ainsi que sa mise en page (CSS). La figure 1 montre un exemple de fichier HTML avec son rendu dans un navigateur. Le concepteur d'une page Web élabore ce document et cette feuille de style de manière à ce que l'utilisateur perçoive le sens d'une page lorsqu'il la visualise sur son navigateur. En conséquence, la recherche d'information dans ces pages doit prendre en compte à la fois la structure logique d'une page et son rendu visuel. Dans cet article, nous proposons donc un modèle de représentation des pages web sous la forme d'un arbre de blocs visuels ainsi qu'un modèle d'indexation de ces pages. Ce modèle d'indexation est une extension du modèle vectoriel (Salton *et al.*, 1975) qui tient compte de l'importance des blocs et de leurs relations de voisinage. Nous ne prenons pas en considération, pour le moment, les liens hypertextuels, ni l'aspect dynamique des pages. Cet article est organisé de la façon suivante. La section 2 présente le modèle de représentation d'une page Web tenant compte de son rendu visuel. La section 3 est consacrée à l'indexation d'une page ainsi représentée. Enfin, la section 4 conclut et dresse des perspectives.

2. Modèle de représentation visuelle d'une page Web

Cette section décrit le modèle de représentation d'une page qui prend en compte les blocs visuels de cette page considérés comme significatifs, leur importance et leurs liens de voisinage.

Une page Web est représentée par un document d et identifiée de manière unique par un URI (Uniform Resource Identifier[2]). d peut être représenté par un arbre du modèle objet de document (DOM[3]), contenant un ensemble de nœuds. La figure 2 donne un exemple de sous-arbre DOM, construit à partir du nœud *body*. Les nœuds sont représentés par des rectangles et les feuilles par des ovales. Chaque ressource

1. http ://www.w3.org/TR/REC-CSS2
2. http ://www.w3.org/Addressing/
3. http ://www.w3.org/DOM/

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Titre</title>
  <style type='text/css'>h2 {margin-top:5ex;}</style>
</head>
<body>
  <h1>Titre</h1>
  <h2>titre1</h2>
  <div>texte11 <span>texte12</span></div>
  <h2>titre2</h2>
  <div>texte2 <div>texte22</div></div>
</body>
</html>
```

Figure 1. *Un document HTML et son affichage dans un navigateur*

multimédia est indiquée par un nœud particulier au sein de cet arbre, et possède aussi un URI qui lui sert d'identifiant. Une fois le document affiché dans un navigateur, il est possible d'accéder aux propriétés visuelles de ses nœuds, comme par exemple les coordonnées de la zone rectangulaire d'affichage du nœud, la couleur de fond, la police utilisée, la taille du texte, la couleur du texte, ...

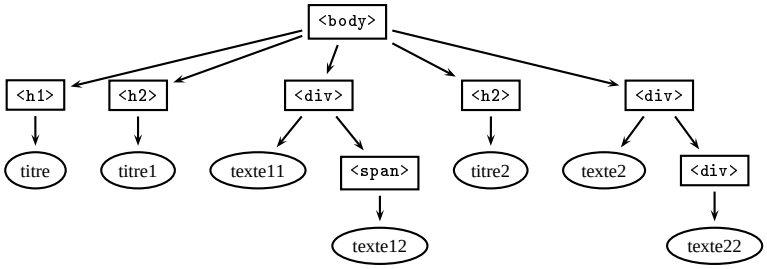

Figure 2. *L'arbre DOM du document HTML de la figure 1*

Nous proposons une nouvelle représentation des documents basée sur une hiérarchie visuelle de blocs, utilisant les propriétés visuelles. Le passage du DOM vers cette représentation s'effectue en plusieurs étapes :

1) Réduction de l'arbre DOM.

2) Construction d'une hiérarchie visuelle par agrégation, à partir des feuilles de l'arbre DOM réduit.

3) Choix d'un plan de coupe dans l'arbre hiérarchique, pour obtenir une granularité optimale des blocs.

2.1. *Réduction de l'arbre DOM*

Une fois la page affichée, son arbre DOM est réduit en groupant les feuilles visuelles F_d. Une feuille visuelle correspond à un noeud dont le rendu forme une unité visuelle atomique, comme le contenu d'un paragraphe ou une image. Pour rester le plus indépendant possible de la structure de l'arbre DOM et du langage propre au document analysé (HTML, XHTML, XML…), nous déterminons l'ensemble des feuilles visuelles en utilisant le flot de disposition des boites CSS [4]. Deux types de dispositions sont spécifiées :

– *block* : le nœud est disposé sous forme d'un bloc, créant une rupture de flot.

– *inline* : le nœud est placé sans rupture dans le flot qui le contient.

Une feuille visuelle correspond à la concaténation des nœuds *inline* contenus dans un nœud *block* jusqu'à ce qu'une rupture de flot soit rencontrée. Par exemple, dans la figure 3, la feuille *texte11 texte12* est une nouvelle feuille visuelle.

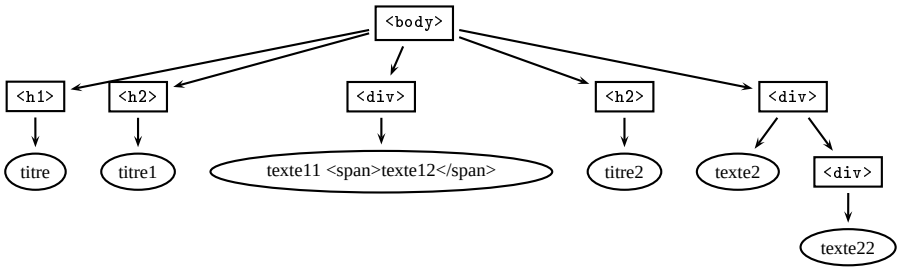

Figure 3. *L'arbre DOM réduit correspondant au modèle de la figure 2*

2.2. *Agrégation de blocs*

L'agrégation des feuilles visuelles en bloc au sein de l'arbre hiérarchique est réalisée à l'aide d'un algorithme de segmentation utilisant des critères visuels (espacement des feuilles, bordures ...). Plusieurs méthodes ont été proposées pour réaliser une telle segmentation, notamment :

– (Cai *et al.,* 2003) proposent un algorithme basé sur l'analyse de l'arbre DOM des documents HTML. Un ensemble d'heuristiques permet de trouver les séparateurs adéquats nécessaires à la segmentation de la page. Ainsi, un arbre hiérarchique de blocs est créé, et chaque bloc possède un degré de cohérence (DoC) représentant l'importance du bloc dans la page.

– (Simon *et al.,* 2005, Zou *et al.,* 2006) utilisent un algorithme de segmentation (Ha *et al.,* 1995) dérivé de l'algorithme récursif *X-Y cut* de (Nagy *et al.,* 1984). Ils l'appliquent à l'extraction de table dans les pages Web.

4. http ://www.w3.org/TR/REC-CSS2/visuren.html#normal-flow

L'algorithme X-Y cut, provenant de la communauté OCR, permet de représenter la structure visuelle d'un document en un arbre hiérarchique, en segmentant au niveau des espaces blancs du document. Cet algorithme, sensible aux bruits sur les documents numérisés, est très performant dans le cas de documents non bruités comme le sont les pages Web.

Notre approche s'inspire de (Zou *et al.*, 2006). Toutefois, notre approche est indépendante du langage HTML, et peut donc être utilisée pour des documents de différents formats du moment qu'ils peuvent s'afficher dans un navigateur. Pour initialiser le *X-Y cut*, nous utilisons l'ensemble F_d. L'ensemble F_d est déterminé en utilisant l'API du moteur de rendu de Mozilla (Gecko [5]) en java. Gecko a l'avantage d'être tolérant aux fichiers comportant des erreurs de syntaxe, et peut en corriger certaines, même si le fichier ne respecte pas forcément les standard du W3C.

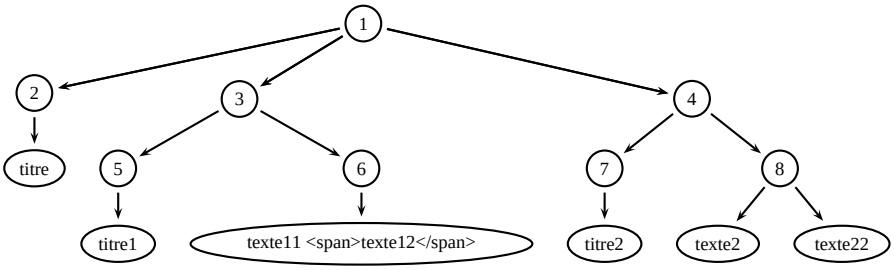

Figure 4. *Hiérarchie visuelle construite à partir des feuilles du DOM réduit*

A la fin de la segmentation, nous obtenons un arbre de blocs hiérarchique qui représente la proximité visuelle entre les blocs appartenant à d. Sur la figure 4, qui montre cet arbre, le nœud 8 montre que *texte21* et *texte22* sont visuellement proches. Le nœud 4 indique que *texte21* et *texte22* sont proches de *titre2*.

2.3. *Plan de coupe*

L'arbre hiérarchique, construit en analysant les espaces blancs entre chaque bloc, a pour feuilles l'ensemble de blocs. La hiérarchie sert à indiquer la proximité visuelle des blocs entre eux. Toutefois, les blocs n'ont peut être pas la granularité désirée. Certains blocs peuvent être agrégés à d'autres car, ensemble, ils forment une unité cohérente. Cette cohérence peut être déterminée, par exemple, en fonction de critères visuels atomiques (police, couleur, taille, ...). Pour ce faire, on définit un plan de coupe dans l'arbre hiérarchique, qui permet de fusionner les blocs entre eux. Nous n'avons pas encore choisi l'algorithme d'élagage.

Une fois le plan de coupe défini, nous obtenons un arbre de blocs hiérarchique définitif, qui donne les relations entre des blocs de granularité comparable.

5. http ://www.mozilla.org/newlayout/

3. Indexation

L'indexation des pages est effectuée à partir du texte contenu dans chaque bloc. L'indexation de ce texte produit un vecteur appelé *vecteur local du bloc*. Ce vecteur est ensuite renforcé par un coefficient d'importance du bloc et par injection de l'indexation des blocs voisins. Cette injection est particulièrement intéressante pour indexer les blocs à contenu multimédia (image ou une vidéo) à partir des textes qui y font référence.

3.1. *Index*

L'indexation textuelle d'un bloc est réalisée conformément au modèle vectoriel (Salton *et al.*, 1975). Elle se traduit par un vecteur de termes, appelé *vecteur local du bloc* dans lequel chaque terme est affecté d'un poids qui dépend de sa fréquence dans le texte du bloc (tf) et de sa fréquence inverse (idf). Dans le modèle vectoriel original la fréquence inverse (idf) est fonction du nombre de documents du corpus indexés par ce terme. Les travaux sur l'indexation des documents semi-structurés, XML principalement, ont montré que ce choix n'était pas le plus adapté et qu'il valait mieux que la fréquence inverse, alors désignée par ief, soit fonction du nombre d'éléments indexés par ce terme (Grabs *et al.*, 2002). Cette idée, appliquée aux blocs, a été reprise par (Debnath *et al.*, 2005, Jiang *et al.*, 2006). C'est le choix que nous avons fait : la fréquence inverse est fonction du nombre de blocs du corpus indexé par ce terme. De plus, comme dans (Bruno *et al.*, 2007), le modèle vectoriel est enrichi comme suit :

1) L'importance du bloc par rapport aux autres blocs de la page. En effet, tous les blocs n'ont pas la même importance visuelle au sein d'une page. Intuitivement, celle-ci est basée sur des critères visuels (le type, la taille et couleur de la police, la couleur de fond...) et des critères spatiaux (position dans la page, surface du bloc...).

2) L'indexation d'un bloc est influencée par l'indexation des blocs voisins dans la page. Un bloc est indexé par deux vecteurs :

- son vecteur local $\overrightarrow{b_i^{local}}$, qui décrit l'information textuelle contenue dans le bloc i. Le vecteur local est produit avec la pondération $tf * idf$.

- son vecteur global, qui représente l'information textuelle contenue dans les blocs qui lui sont associés par le coefficient de proximité β (Cf. section 3.2).

Le vecteur global est défini comme suit :

$$\overrightarrow{b_i} = \alpha_i \overrightarrow{b_i^{local}} + \sum_{j=1,n} \alpha_j \beta_{b_j,b_i} \overrightarrow{b_j} \quad [1]$$

Dans l'équation 1, le vecteur local est pondéré par l'importance α_i du bloc b_i au sein de la page. En général, α_i est déterminé par apprentissage (Song *et al.*, 2004, Liu *et al.*, 2006). Nous envisageons d'utiliser l'approche de (Liu *et al.*, 2006) pour déterminer α.

L'index de chaque bloc b_i est ainsi augmenté par les termes des blocs qui lui sont proches. La propagation des index d'un bloc à ses voisins, est représentée par le vecteur global du bloc, pondérée par la distance et l'importance des voisins.

3.2. *Proximité*

Le coefficient β_{b_j,b_i}, qui exprime la proximité visuelle entre le bloc b_j et le bloc b_i est exprimé en fonction de la distance de ces blocs dans l'arbre des blocs de cette page, et est défini comme suit :

$$\beta_{b_j,b_i} = \frac{\gamma}{(dist(b_j,k)+dist(b_i,k))^n} \quad [2]$$

où k est le plus proche ancêtre commun de b_i et b_j dans l'arbre de blocs hiérarchique , et $dist$ la fonction qui retourne le nombre d'arcs entre deux nœuds ancêtre/descendant.

Dans l'évaluation des propriétés de notre modèle, $\gamma = 4$, et $n = 2$ sont des valeurs possibles. Une autre approche envisagée est d'estimer β_{b_j,b_i} par apprentissage.

4. Conclusion et perspectives

Dans cet article, nous avons proposé un modèle de représentation et d'indexation de pages Web basé sur leur aspect visuel. Ce modèle est basé sur la décomposition d'une page Web en une hiérarchie de blocs. Cette décomposition est obtenue par la segmentation du rendu visuel de la page. Dans ce modèle, l'indexation d'un bloc dépend de son importance visuelle et de l'index de ces blocs voisins.

Nous avons réalisé une implémentation en Java, en utilisant le moteur de rendu Gecko, sur une résolution de 1024x768 et la librairie de recherche d'information Lucene[6]. En attendant de disposer de campagnes d'évaluation réellement adéquates à notre approche, nous testons notre modèle sur un corpus de 700 pages de journaux en ligne collectées pendant un mois et 5000 images, sur un jeu de 20 requêtes. La précision est évaluée manuellement par retour des utilisateurs.

De nombreuses perspectives d'utilisation du modèle et de l'indexation de blocs existent, comme par exemple l'indexation d'images sans annotation grâce au texte voisin, la recherche du meilleur point d'entrée dans une page par rapport à une recherche par mots-clé, la détection des blocs non sémantiques (pubs, blocs de navigations...).

Remerciements

Je tiens particulièrement à remercier Jacques Le Maitre et Emmanuel Bruno, mes directeurs de thèse, ainsi que Michel Scholl, pour leur soutien, leurs conseils avisés, et leurs multiples relectures.

6. http ://lucene.apache.org/

5. Bibliographie

Bruno E., Faessel N., Le Maitre J., « Indexation of Web Pages Based on their Visual Rendering », *Proceedings of the IADIS International Conference WWW/Internet 2007*, vol. 2, Vila Real, Portugal, p. 193-197, October, 2007.

Cai D., Yu S., Wen J.-R., Ma W.-Y., VIPS : A Vision-based Page Segmentation Algorithm, Technical report, Microsoft Research, 2003.

Debnath S., Mitra P., Pal N., Giles C. L., « Automatic identification of informative sections of Web pages », *Knowledge and Data Engineering, IEEE Transactions on*, vol. 17, n° 9, p. 1233-1246, 2005.

Grabs T., Schek H.-J., « ETH Zürich at INEX : Flexible Information Retrieval from XML with PowerDB-XML », *in* , N. Fuhr, , N. Gövert, , G. Kazai, , M. Lalmas (eds), *Proceedings of the First Workshop of the INitiative for the Evaluation of XML Retrieval (INEX)*, Schloss Dagstuhl, Germany, p. 141-148, December, 2002.

Ha J., Haralick R. M., Phillips I. T., « Recursive X-Y cut using bounding boxes of connected components », *ICDAR '95 : Proceedings of the Third International Conference on Document Analysis and Recognition (Volume 2)*, IEEE Computer Society, Washington, DC, USA, p. 952, 1995.

Jiang T., Tan A.-H., « Discovering Image-Text Associations for Cross-Media Web Information Fusion. », *in* , J. Fürnkranz, , T. Scheffer, , M. Spiliopoulou (eds), *PKDD*, vol. 4213 of *Lecture Notes in Computer Science*, Springer, p. 561-568, 2006.

Liu Y., Wang Q., Wang Q., Liu Y., Wei L., « An Adaptive Scoring Method for Block Importance Learning », *WI '06 : Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, IEEE Computer Society, Washington, DC, USA, p. 761-764, 2006.

Nagy G., Seth S., « Hierarchical Representation of Optically Scanned Documents », *ICPR84*, p. 347-349, 1984.

Salton G., Wong A., Yang C. S., « A vector space model for automatic indexing », *Commun. ACM*, vol. 18, n° 11, p. 613-620, 1975.

Simon K., Lausen G., « ViPER : augmenting automatic information extraction with visual perceptions », *CIKM '05 : Proceedings of the 14th ACM international conference on Information and knowledge management*, ACM Press, New York, NY, USA, p. 381-388, 2005.

Song R., Liu H., Wen J.-R., Ma W.-Y., « Learning important models for web page blocks based on layout and content analysis », *SIGKDD Explor. Newsl.*, vol. 6, n° 2, p. 14-23, 2004.

Zou J., Le D., Thoma G. R., « Combining DOM tree and geometric layout analysis for online medical journal article segmentation », *JCDL '06 : Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*, ACM Press, New York, NY, USA, p. 119-128, 2006.