Traiter les documents XML avec les « contextes de lecture »

Xavier Tannier

Centre de recherche Xerox de Grenoble
6, chemin de Maupertuis
38240 Meylan
Xavier.Tannier@xrce.xerox.com[1]

RÉSUMÉ. *Le langage XML autorise, par sa souplesse de structuration, des manipulations du contenu qui créent parfois des ruptures arbitraires dans le flot naturel du texte. Ces caractéristiques soulèvent des difficultés lorsque l'on souhaite mettre en œuvre des techniques d'analyse automatique du contenu des documents XML. Cet article présente cette problématique et y répond, sur le plan théorique, avec l'introduction du concept de contexte de lecture, puis sur le plan pratique, avec une classification automatique des balises XML et la présentation d'un outil générique de gestion des contenus XML.*

ABSTRACT. *Some tags used in XML documents create arbitrary breaks in the natural flow of the text. This flexibility may raise some difficulties for some techniques of document engineering. This article presents this issue and proposes answers, theoretically first, with the introduction of a new concept of reading context, and in practice afterwards, with an automatic classification of tags and the presentation of a generic tool for XML content handling.*

MOTS-CLÉS : *XML, contexte de lecture, proximité logique, classes de balises, traitement automatique de la langue.*

KEYWORDS: *XML, reading context, logical proximity, tag classes, natural language processing.*

1. Ce travail a été réalisé alors que l'auteur était à l'École des Mines de Saint-Étienne (département RIM), 158, Cours Fauriel, 42023 Saint-Étienne.

1. Introduction

XML (W3C, 2004)) est un langage dit semi-structuré qui permet de représenter des données textuelles en les enrichissant avec des annotations structurelles et sémantiques. Ces annotations sont introduites par des balises, indépendamment de la présentation du texte. La facilité et la souplesse du langage XML en ont fait un standard en matière d'échange et de stockage d'information, notamment au travers d'Internet.

Ici nous considérons le document XML selon l'approche dite « document ». Dans cette approche, le document est considéré sous sa forme traditionnelle, et le balisage sert à fournir des informations concernant la structure logique et/ou la forme du texte. Cette vision des choses est adaptée aux textes destinés à des humains, comme les manuels, les livres, les articles ou les pages Web statiques (voir la figure 1). Elle est opposée à l'approche « données », plus utilisée par des applications de bases de données (horaires de vols, catalogues, etc.) (Fuhr et Großjohann, 2001).

L'ajout d'une structure souple aux textes pose un certain nombre de problèmes lorsqu'il s'agit d'en faire une analyse automatique, de quelque type que ce soit. Dans cet article, nous détaillons certaines de ces difficultés (section 2), et proposons de les regrouper grâce à la définition du concept de « contexte de lecture » (3.1). Nous revenons ensuite sur une catégorisation existante des balises (3.2), et montrons qu'elle s'adapte bien à ce nouveau concept (3.3). Nous mettons en place une technique de détermination automatique de la catégorie des balises (section 4). Enfin, nous présentons XGTagger (section 5), une interface générique de traitement des contenus XML s'appuyant sur toutes les notions vues précédemment.

```
<livre type="roman">
    <titre>Le tour du monde en 80 jours</titre>
    <auteur>Jules Verne</auteur>
    <éditeur>Hatzel</éditeur>
    <chapitre n="I">
        <titre_chapitre>
            Dans lequel Phileas Fogg et Passepartout s'acceptent réciproquement, l'un
            comme maître, l'autre comme domestique
        </titre_chapitre>
        En l'année 1872, la maison portant le numéro 7 de Saville-row, Burlington Gardens – maison dans laquelle Sheridan mourut en 1814 – était habitée par Phileas Fogg, esq., l'un des membres les plus singuliers et les plus remarqués du Reform-Club de Londres, bien qu'il semblât prendre à tâche de ne rien faire qui pût attirer l'attention. [...]
    </chapitre>
</livre>
```

Figure 1 – Représentation en XML d'un roman.

2. Problématique

Voici quelques indications de terminologie importantes pour comprendre la suite de l'article. Dans l'exemple de la figure 1 :

– « livre », « titre » sont des *noms de balise* ;

– `<titre>` et `</titre>` sont des *balises* (resp. balises de début et de fin) ;

– Le texte « Le tour du monde en 80 jours » est un *contenu*.

– Les balises de début et de fin ainsi que leur contenu (situé entre elles) constituent un *élément* XML.

– *type* est un *attribut* de l'élément ‘`livre`’ ayant pour *valeur « roman »*.

– Enfin, la DTD est un document définissant les balises pouvant être utilisées dans le fichier XML, ainsi que leur structure (imbrication, nombre, séquences, etc.).

Le balisage semi-structuré de type XML permet de prendre certaines libertés avec le texte lui-même. Ainsi, un groupe de mots (ou même un seul mot) peut être coupé par une balise ; des éléments contextuellement très éloignés peuvent être physiquement proches dans le document XML, et inversement. C'est la sémantique particulière des balises, attribuée par un expert humain, qui autorise ces manipulations tout en conservant le sens du texte initial. À l'opposé, pour les processeurs XML, les balises sont toutes équivalentes, et surtout totalement vides de sens.

Ces caractéristiques propres aux documents semi-structurés ne posent pas uniquement des problèmes de « haut niveau » (comme l'extraction d'information ou les relations sémantiques entre balises, dont les enjeux sont l'objet d'initiatives telles que XML Schema (W3C, 2001b) ou le Semantic Web (W3C, 2001a)), mais aussi des difficultés plus élémentaires, que nous présentons dans cette section.

Les différentes facettes du problème que nous abordons sont parfois décrites à l'aide d'exemples inventés ou adaptés d'exemples réels. Le but est de présenter de façon concise et claire les aspects que nous voulons mettre en évidence. La contrepartie est que l'illustration peut paraître artificielle, mais il nous semble que l'utilisation d'exemples réels aurait nui à la facilité de lecture et à la concision de l'exposé.

2.1. *Rattachement des mots*

De nombreux phénomènes (indications de forme, notes, mises à jour, transcriptions, insertion de champs de métadonnées) provoquent certaines anomalies dans la continuité physique du texte, et empêchent de distinguer correctement les mots.

Ainsi, dans l'exemple 1.a ci-dessous, les mots *‘Petit’* et *‘Prince’* sont coupés par des balises ‘`pc`’ (petites capitales). Pour retrouver les véritables mots, il est nécessaire de supprimer totalement les balises. Mais si l'on applique cette méthode aux exemples 1.b et 1.c, les suites de caractères *‘Antoinede’* et *‘1943Son’* seront reconnus

comme des mots, au lieu de *'Antoine'*, *'de'*, *'1943'* et *'Son'*. Ici les balises 'prenom', 'nom' et 'note' doivent être remplacées par un blanc.

(1) a. <titre>Le P<pc>etit</pc> P<pc>rince</pc></titre>

b. <auteur>
<prenom>Antoine</prenom><nom>de St-Exupéry</nom>
</auteur>

c. Livre publié en 1943<note>Son auteur est mort l'année suivante.</note> par Gallimard.

Ces particularités ont de l'importance en ingénierie des documents (pour l'indexation par exemple). Les systèmes que nous connaissons contournent ce problème en listant les balises de forme (comme 1pc' dans notre exemple) et en les supprimant avant toute autre opération. Les autres types de balises sont alors tous considérés comme des coupures dans le texte, et des espaces sont ajoutés pour éviter tout rattachement malencontreux. Cette méthode conduit à une perte d'information (les balises de forme, qui représentent pourtant une indication importante) et ne rétablit pas l'ordre des mots (dans le cas des notes, par exemple, qui sont d'utilisation très courante). Ce dernier aspect pose la question de la proximité réelle des termes dans le document XML.

2.2. *Proximité logique*

Il est nécessaire de distinguer la proximité *physique* de ce que nous pouvons appeler la proximité *logique* de deux termes dans un document XML. Tandis que la proximité physique est définie par la position des termes dans le fichier, la proximité logique dépend de leur organisation dans la structure. Ainsi, dans l'exemple 2, les mots « *élections* » et « *aux États-Unis* » sont physiquement consécutifs, mais logiquement éloignés (la section 3 apportera une définition formelle de la proximité logique).

(2) <infos>
<item>Dernier sondage, à quatre jours des élections</item>
<item>Aux États-Unis, une fausse alerte provoque la panique...</item>
</infos>

(3) a. <par>
Les élections aux <gras>États-Unis</gras> sont prévues pour 2008.
</par>

b. <titre>
Les commentaires de Noam Chomsky concernant les <italique>élections</italique> aux <souligne>États-Unis</souligne>.
</titre>

(4) a. <transcription_orale>
Les nouvelles ne parlent plus que des élections aux États-<commentaire>Une porte claque.</commentaire>Unis.
</transcription_orale>

b. <paragraphe>
En 2004, les élections<nbp>Nous parlons bien sûr des élections présidentielles</nbp> aux États-Unis furent moins controversées qu'en 2000.
</paragraphe>

c. <résumé>
Cet article traite du voyage du président roumain<nbp>Traian Basescu, qui a remporté de justesse les dernières élections</nbp> aux États-Unis.
</résumé>

Cette distinction est notamment intéressante en recherche d'information. Supposons que l'on recherche des renseignements sur les élections aux États-Unis. L'ensemble d'exemples ci-dessus montre que la proximité physique des termes « élections » et « États-Unis » ne garantit pas la pertinence de ce bigramme, plutôt liée à la proximité logique. Ainsi les exemples 2 et 4.c ne sont pas pertinents pour notre recherche, malgré la proximité physique des termes de la requête, alors que 3.a et 3.b, dans une situation comparable (mots séparés par des balises), sont pertinents. Enfin 4.a et 4.b concernent bien le sujet recherché, malgré les éléments XML insérés.

Si la recherche de motifs est concernée, tout traitement syntaxique des textes l'est bien entendu également. Notons que cette idée de proximité est également applicable aux listes, ce qui est particulièrement intéressant. Les listes comprennent une introduction, plusieurs éléments et parfois une conclusion (Aït-Mokhtar et al., 2003), et il existe la même proximité logique entre l'introduction et chaque élément de la liste, et entre chaque élément et la conclusion. Ainsi, dans l'exemple suivant, les mots *« lipides »* et *« protéines »* sont *physiquement* éloignés du terme *« macronutriments »* (numérotation du bas), mais si l'on prend en compte la proximité *logique* (numérotation du haut), les trois constituants (*« glucides »*, *« lipides »* et *« protéines »*) sont chacun à la même distance de l'introduction.

(5) Les $^{1}_{1}$ macronutriments $^{2}_{2}$ sont $^{3}_{3}$ constitués $^{4}_{4}$ par $^{5}_{5}$:
<liste>
<item>les $^{6}_{6}$ glucides $^{7}_{7}$, appelés plus communément sucres, séparés en glucides simples et glucides complexes.</item>
<item>les $^{6}_{21}$ lipides $^{7}_{22}$, constituants majeurs des matières grasses, qui englobent les acides gras saturés, mono-insaturés et polyinsaturés.</item>
<item>les $^{6}_{36}$ protéines $^{7}_{37}$, constituées d'acides aminés souvent indispensables car non synthétisables par l'organisme ...</item>
</liste>

2.3. *Traitement automatique du langage*

Les documents XML sont, tout comme les documents plats (c'est-à-dire non structurés), un terrain d'étude intéressant pour les chercheurs en traitement automatique des langues. Mais dans le cas de XML, un problème supplémentaire réside dans la difficulté de préserver le fil de la lecture de l'être humain, indépendamment des éléments structurels, ce qui est bien sûr indispensable en traitement de la langue.

C'est notamment nécessaire pour effectuer une analyse morphosyntaxique automatique correcte. En effet, les logiciels d'étiquetage morphosyntaxique déterminent la catégorie grammaticale d'un mot ambigu en utilisant le contexte dans lequel il est utilisé (les termes l'entourant). Le problème se pose également pour des analyses plus avancées en termes de syntaxe et/ou de sémantique. Les exemples ci-dessus peuvent être lus « à l'envers » pour illustrer cela. Ainsi, dans l'exemple 4, toute analyse linguistique devra bien évidemment considérer comme un tout non dissociable les phrases *« Les nouvelles ne parlent plus que des élections aux États-Unis »* (4.a), *« En 2004, les élections aux États-Unis furent moins controversées qu'en 2000. »* (4.b) et *« Cet article parle du prochain voyage du président roumain aux États-Unis »* (4.c), et cela malgré les éléments qui les interrompent. Ici un traitement approprié des balises est nécessaire pour retrouver le texte « initial » (celui qui est destiné à être lu par l'être humain).

3. Le contexte de lecture et la classification des balises

3.1. *Le contexte de lecture*

Il nous semble possible de rendre compte de l'ensemble des caractéristiques présentées ci-dessus en définissant une notion que nous appelons « contexte de lecture ». Un *contexte de lecture* est une petite partie de texte, syntaxiquement et sémantiquement indépendante et homogène, qu'une personne peut lire d'une seule traite, sans interruption. Dans un document XML en particulier, les contextes de lecture ne respectent pas forcément la linéarité du texte, et dépendent fortement des éléments structurels mis en place. Par exemple :

– Les balises '`item`' de l'exemple 2 provoquent un changement de contexte de lecture, car leurs contenus sont distincts et syntaxiquement incompatibles.

– Les balises '`pc`' (petites capitales) de l'exemple 1.a apparaissent à l'intérieur d'un contexte de lecture (*« Le Petit Prince »*), ne l'interrompent pas et ne le modifient pas.

– L'élément '`nbp`', dans l'exemple 4.b, forme un nouveau contexte de lecture qui s'insère dans un contexte existant, celui-ci reprenant par la suite.

– Enfin, les listes sont encore plus particulières. Il est possible de les interpréter de façons différentes, selon l'application souhaitée. On peut par exemple estimer que chaque élément d'une liste représente un contexte de lecture indépendant, et que l'introduction et la conclusion sont communes à tous (on aurait donc trois contextes différents dans l'exemple 5 : *« Les macronutriments sont constitués par les glucides, appelés plus communément sucres … »*, *« Les macronutriments sont constitués par les lipides, constituants majeurs des matières grasses … »* et *« Les macronutriments sont constitués par les protéines, constituées d'acides aminés … »*). Mais cela pose des problèmes, notamment de ponctuation, que nous ne souhaitons pas aborder ici.

Ainsi, chacun des problèmes exprimés dans la section 2 peut se définir ou s'énon-

cer en utilisant le concept de **contexte de lecture** :

– **rattachement des mots.** Deux chaînes de caractères forment un mot[1] si elles sont accolées sans espace dans le même contexte de lecture.

– **proximité logique.** Deux mots sont logiquement consécutifs s'ils sont consécutifs dans le *même* contexte de lecture.

– **traitement automatique du langage.** Une analyse morphosyntaxique, syntaxique, sémantique ou autres doit considérer les contextes de lecture, et non pas le texte dans l'ordre d'apparition dans le document.

3.2. *Types de balises XML*

Les sections précédentes nous montrent que les balises XML ont des influences variées sur les contextes de lecture. Pour traiter de façon automatique le contenu d'un document XML, il semble donc nécessaire de distinguer plusieurs types de balises (Renear et al., 2002). Goldfarb (1981) faisait déjà la distinction entre les approches « procédurale » (instructions de formatage) et « descriptive » (composants logiques, comme les paragraphes, les chapitres ou les titres). Cette distinction est devenue consensuelle avec SGML (Goldfarb, 1991) puis XML. Par ailleurs, la TEI (Text Encoding Initiative (Ide et Veronis, 1995)) a mis en place une intéressante répartition entre différents niveaux d'inclusion : « chunk », « phrase-level » et « inter-level ». La définition de ces niveaux est purement syntaxique et aucune distinction sémantique n'est faite.

Une division des balises a également été proposée par Lini et al. (2001), dans le but d'identifier différentes catégories qu'il serait important de distinguer dans le cadre de la recherche dans des documents XML. L'idée de départ était de permettre des traitements différents pendant une recherche de motif (de séquence de caractères) (Lini et al., 2001; Colazzo et al., 2002). Les différentes classes de balises sont les suivantes :

– les *balises « dures »* (*« hard » tags*) sont les plus fréquentes. Elles interrompent la « linéarité » d'un texte et contribuent généralement à la structuration du document. Des exemples de balises dures sont les titres, les chapitres, les paragraphes.

(6) … <titre>Text Retrieval System</titre>
<auteur>Michele Paoli</auteur> …

– Les *balises « transparentes »* (*« soft » tags*) identifient des parties significatives du texte, comme les textes cités, les effets de forme ou les corrections, mais sont « transparentes » lorsqu'on lit le texte.

(7) Les ministres de l'<gras>Union Européenne</gras> se sont réunis …

– Les *balises de « saut »* (*« jump » tags*) sont utilisées pour représenter des éléments particuliers comme les notes de marges, les références bibliographiques ou des définitions. Elles sont détachées du texte les entourant, comme la `'note'` suivante :

1. La définition du mot se doit bien entendu d'être plus précise, mais cette question ne nous préoccupe pas dans ce contexte.

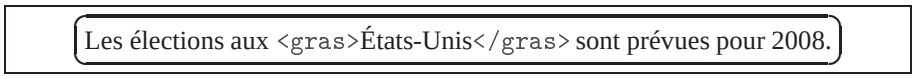


Figure 2 – Contextes de lecture et balises transparentes

(8) La mort de Mozart, dans l'indifférence générale<note>Son corps sera jeté à la fosse commune, sans même une croix.</note>, l'empêcha de terminer son requiem.

N.B. : Des éléments comme les tableaux ou les listes, qui ne sont pas abordés par les auteurs, semblent particuliers (Douglas et al., 1995; Aït-Mokhtar et al., 2003). Par ailleurs, les formules mathématiques, chimiques ou autres peuvent être utilisées dans le texte (éléments transparents) ou dans un contexte à part (éléments durs).

3.3. *Contextes de lecture et types de balises*

Nous reprenons ici à notre compte les trois classes différenciées par Lini et al. (2001). Pour chacune d'entre elles, nous proposons deux nouvelles définitions. La première est exprimée en fonction des contextes de lecture ; la seconde est une définition constructive adoptant une vision plus linguistique. Nous montrerons qu'elle est également plus sujette à des ambiguïtés diverses. Nous la proposons en vue de la catégorisation automatique des balises présentée à la section 4.

Soient $\begin{cases} n & \text{le nom de la balise dont nous souhaitons déterminer la classe} \\ e_n & \text{un élément XML de nom } n \\ p(e_n) & \text{l'élément parent de } e_n \text{ } (p(e_n) \text{ contient } e_n) \end{cases}$

3.3.1. *Balises transparentes*

Définition $\mathcal{D}_1^T(n)$: *Une balise transparente s'inscrit dans le contexte de lecture courant sans le modifier ni l'interrompre (figure 2).*

Définition $\mathcal{D}_2^T(n)$: *L'élément* e_n *est un élément transparent s'il est possible de supprimer le balisage et d'obtenir un texte intelligible dans* $p(e_n)$.

L'application de cette dernière définition à nos exemples, repris des illustrations de la proximité logique (section 2.2), conduit aux textes suivants. Ainsi, seules les balises `'gras'`, `'italique'`, `'souligne'` et `'sc'` sont des balises transparentes (cas 3').

(2') * Dernier sondage, à quatre jours des électionsAux États-Unis, une fausse alerte provoque la panique dans un avion

(3') a'. Les élections aux États-Unis sont prévues pour 2008.

b'. Les commentaires de Noam Chomsky concernant les élections aux États-Unis.

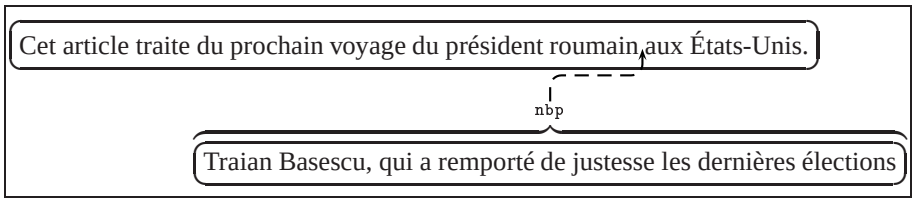


Figure 3 – Contextes de lecture et balises de saut

(4') a'. * Les nouvelles ne parlent plus que des élections aux États-Une porte claque.Unis.

b'. * En 2004, les électionsNous parlons bien sûr des élections présidentielles aux États-Unis furent moins controversées qu'en 2000.

c'. * Cet article traite du prochain voyage du président roumainTraian Basescu, qui a remporté de justesse les dernières élections aux États-Unis.

3.3.2. *Balises de saut*

Définition $\mathcal{D}_1^S(n)$: *Une balise de saut est insérée à l'intérieur d'un contexte de lecture (figure 3).*

Définition $\mathcal{D}_2^S(n)$: *L'élément* e_n *est un élément de saut s'il est possible de supprimer l'élément entier (balisage + contenu) et d'obtenir autour un texte intelligible dans* $p(e_n)$.

Dans les exemples qui suivent, la seconde définition ($\mathcal{D}_2^S(n)$) montre que `'commentaire'` et `'nbp'` (note de bas de page) sont des éléments de saut (cas 4").

(3") a". * Les élections aux sont prévues pour 2008.

b". * Les commentaires de Noam Chomsky concernant les aux .

(4") a". Les nouvelles ne parlent plus que des élections aux États-Unis.

b". En 2004, les élections aux États-Unis furent moins controversées qu'en 2000.

c". Cet article traite du prochain voyage du président roumain aux États-Unis.

3.3.3. *Balises dures*

Définition $\mathcal{D}_1^D(n)$: *Une balise dure provoque un changement de contexte de lecture (figure 4).*

Définition $\mathcal{D}_2^D(n)$: Une balise dure a des caractéristiques spécifiques. Pourtant, pour que nos trois classes forment sûrement une partition, et par souci de simplicité, nous proposons : *Une balise dure n'est ni une balise de saut ni une balise transparente.*

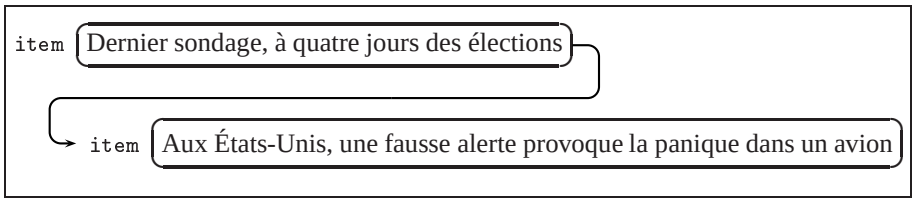


Figure 4 – Contextes de lecture et balises dures

3.3.4. *Commentaires*

Il est évident que les définitions constructives $\mathcal{D}_2$ ne peuvent avoir une application isolée ; on ne peut considérer chaque élément séparément, pour les raisons suivantes :

– Si l'élément parent $p(e_n)$ n'a pas de contenu mixte (c'est-à-dire s'il ne contient aucune donnée textuelle, mais uniquement d'autres éléments XML), il sera impossible de conclure, comme le montre l'exemple suivant pour les `'italiques'` :

(9) <titre>
<italiques>Les technologies de l'Espace</italiques>
</titre>

– Si la portion de texte considérée contient d'autres éléments que du texte et des éléments de type n, il est impossible de reconstituer avec certitude le texte réel. En effet, nous ignorons en théorie la classe des autres balises :

(10) <titre>
Les <gras>voyages humains</gras> vers <ital>Mars</ital> pourraient avoir lieu plus tôt que prévu.
</titre>

Dans cet exemple, en s'intéressant à la balise de type `'ital'`, la classe de `'gras'` est inconnue. On ignore donc quelle partie de texte doit être conservée pour l'analyse.

– Dans certains cas, les deux définitions $\mathcal{D}_2^t$ et $\mathcal{D}_2^s$ peuvent s'appliquer à un même élément, et ainsi conduire à une ambiguïté concernant le type (transparent ou de saut) d'une balise :

(11) Napoléon Bonaparte <note>qui naquit en Corse en 1769</note> mourut à l'âge de 52 ans après avoir marqué le monde de son empreinte.

(12) Le professeur <bold>Stephen Hawkins</bold> travaille sur les lois qui gouvernent l'Univers.

Nous montrerons dans la section suivante comment ces difficultés peuvent être contournées.

4. Classification automatique

Pour traiter de façon spéficique certaines classes de balises, quelle que soit la classification (balises dures, transparentes, de saut (Colazzo et al., 2002), balises d'emphase (van Zwol et al., 2005), balises équivalentes (Abiteboul et al., 2005)), la liste des balises appartenant à telle ou telle classe est toujours obtenue manuellement (par l' « intuition ») ou en utilisant le nom des balises ou encore les commentaires présents dans la DTD (Abiteboul et al., 2005). Mais l'intuition prend du temps et a ses limites, les noms de balises sont rarement de « vrais » mots et les commentaires, leur clarté, leur disposition et surtout leur présence, sont très dépendants de la personne qui les a écrits, ce qui en fait des indices bien trop approximatifs.

Nous proposons ici une méthode pour séparer automatiquement les balises entre les trois classes que nous avons décrites précédemment (balises dures, transparentes, de saut). Cette méthode s'affranchit des contraintes liées à la DTD, puisqu'elle ne l'utilise pas du tout. Elle repose sur les caractéristiques des balises en termes de comportement des contextes de lecture et sur une procédure basée sur une analyse syntaxique du texte.

4.1. *Description de l'approche*

4.1.1. *Approche globale et nouvelles définitions*

Nous utilisons les définitions $\mathcal{D}_2$ introduites ci-dessus. Nous remplaçons le terme *« texte intelligible »*, suffisant pour un être humain, mais bien sûr trop imprécis pour un algorithme automatique, par *« texte syntaxiquement correct »*. Ainsi :

Définition $\mathcal{D}_{2'}^{T}(n)$: *L'élément* e_n *est un élément transparent s'il est possible de supprimer le balisage et d'obtenir un texte syntaxiquement correct dans* $p(e_n)$.

Définition $\mathcal{D}_{2'}^{S}(n)$: *L'élément* e_n *est un élément de saut s'il est possible de supprimer l'élément entier (balisage + contenu) et d'obtenir autour un texte syntaxiquement correct dans* $p(e_n)$.

Définition $\mathcal{D}_{2'}^{D}(n)$: La définition ne change pas par rapport à $\mathcal{D}_{2}^{D}$: *Une balise dure n'est ni une balise de saut ni une balise transparente.*

Nous avons décrit plus haut les différentes raisons pour lesquelles ces définitions n'étaient pas applicables efficacement à chaque balise rencontrée séparément. À ces arguments s'en ajoute un nouveau concernant l'analyse syntaxique assistée par ordinateur. Les analyses erronées restent légion, quel que soit le soin apporté à la conception de la grammaire ① [2]. Enfin, il est fréquent que des passages de texte ne respectent pas la grammaire d'une langue déterminée, mais soient composés d'abréviations, de fonctions mathématiques, ou d'autres formes de chaînes de caractères qu'un analyseur ne peut reconnaître de façon générique ②.

2. Les numéros entourés de cette section correspondent aux numéros de la figure 5.

Toutes ces raisons nous incitent donc à adopter une approche plus globale des types de balises. En effet, une classe (transparent, dur ou de saut) est attribuée à un type (un nom) de balise. Les cas où des balises d'un même type peuvent être utilisées de deux manières différentes sont extrêmement rares et souvent dus à des abus de la part des auteurs. Ainsi, nous pouvons mettre en place, en complément de l'analyse syntaxique, une analyse statistique des résultats obtenus. En effet, même si les éléments ne comprenant aucun contenu mixte sont écartés, même si ceux comprenant plusieurs types de balises ne sont pas traités, même si quelques cas provoquent une ambiguïté ③, et même si des phrases correctes sont négligées par l'analyseur ①, notre hypothèse est qu'un nombre suffisant de cas sera traité de façon appropriée pour aboutir à des résultats significatifs (au dessus d'un seuil s ④), et ainsi conclure sans équivoque sur la classe des noms de balises.

De plus, nous estimons que la probabilité pour que l'application des deux définitions $\mathcal{D}_{2'}^{T}$ et $\mathcal{D}_{2'}^{S}$ sur la même balise conduisent toutes deux à des phrases correctes (phénomène illustré par les exemples de Napoléon – 11 – et Stephen Hawkins – 12) est relativement faible ③, même si elle peut dépendre des habitudes d'écriture des auteurs.

La conclusion de ces réflexions est que les cas pour lesquels le système conclura *à tort* qu'une classe *s'applique* à une balise donnée devraient être très minoritaires ⑥, à condition d'avoir une analyse syntaxique qui ne valide pas trop de constructions incorrectes ⑤. De cette façon, même s'il arrive souvent que le système conclue, toujours à tort, qu'une définition *ne s'applique pas* ⑦, la différence entre ces deux cas sera suffisamment importante pour permettre de parvenir à se prononcer de façon pertinente ④.

Tout cela est illustré à la figure 5.

4.1.2. *Analyse syntaxique*

Ici l'analyse syntaxique a pour seul but de regrouper des suites de mots. Aucun traitement autre que la simple reconnaissance des structures n'est nécessaire. Nous nous contentons donc de mettre en place une analyse morphosyntaxique avec le logiciel TreeTagger (Schmid, 1994) suivie d'une analyse superficielle avec le logiciel *Cass* (Abney, 1996), fonctionnant sur la base d'une cascade d'automates à états finis, et dont nous conservons les règles suggérées par l'auteur[3]. Le travail présenté concerne des textes en langue anglaise.

Le problème des ambiguïtés syntaxiques, non gérées par *Cass*, ne nous préoccupe pas. Des rattachements sémantiquement erronés nous importent peu, nous souhaitons simplement prouver qu'un énoncé respecte une grammaire donnée ou pas. De plus, les collections de documents pouvant être très volumineuses, l'efficacité en termes de temps de calcul est importante, et le logiciel d'Abney affiche de très bons résultats à ce niveau. Enfin, nos contraintes de précision exprimées plus haut trouvent une réponse

3. À l'exception d'un niveau nommé `nmess`, qui provoque un certain nombre d'analyses erronées, ce que nous voulons éviter à tout prix.

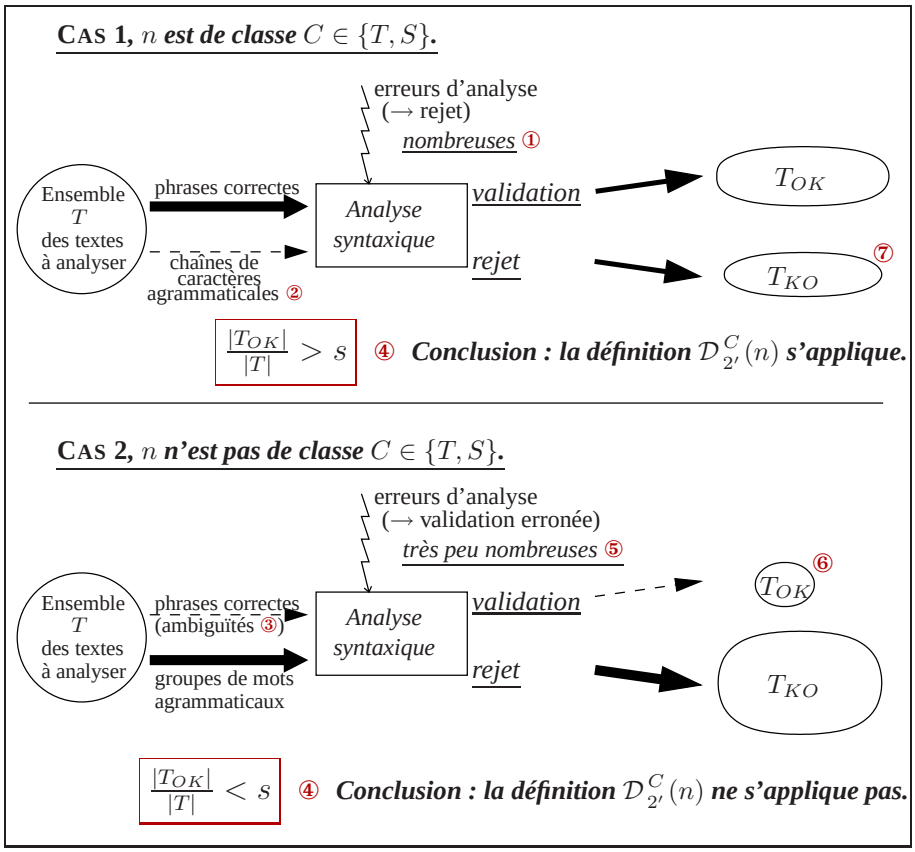


Figure 5 – Vers une application automatique des définitions $\mathcal{D}_{2'}$.

adaptée dans l'application d'une analyse superficielle. En effet, l'analyse de *Cass* ne concerne que des constituants basiques et a peu de chances de valider des énoncés incorrects.

4.1.3. *Algorithme de détermination automatique*

Pour déterminer si un type de balise n est de classe C ($C \in \{T, S\}$ – transparente ou de saut), on applique l'algorithme suivant :

1) *Collecte des éléments.* À partir de la collection complète, tous les éléments parents d'au moins une balise de type n sont sélectionnés ($p(e_n)$).

2) *Filtrage.* Seuls les éléments auxquels il est possible d'appliquer la définition $\mathcal{D}^{C}_{2'}$ sont conservés :

- $p(e_n)$ doit contenir une ou plusieurs parties textuelles, et pas seulement d'autres nœuds ;

- $p(e_n)$ ne doit pas contenir d'éléments autres que du texte et des éléments n.

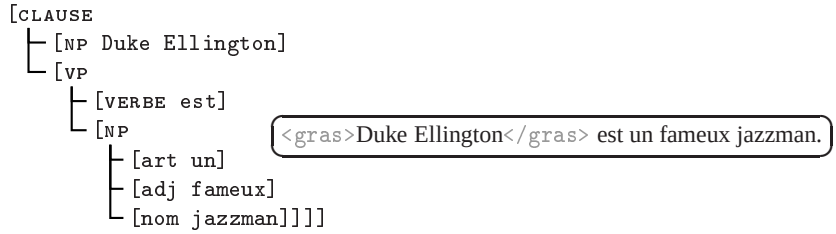
3) *Sélection des portions de texte à analyser.* Pour chaque élément parent $p(e_n)$, le texte adéquat est sélectionné en fonction de la définition concernant la classe recherchée ($\mathcal{D}^C_{2'}(n)$). Pour la transparence ($C = T$), on enlève le balisage et on conserve tout le texte ; pour le saut ($C = S$), on supprime les éléments de type n.
Dans le cas où le texte contient plusieurs phrases (*i.e.* si une ponctuation forte apparaît dans le texte), on réduit la portion de texte à la phrase contenant l'élément étudié :

(13) <nouvelles>
Sports – Tennis : la Belge <italiques>Kim Clijsters</italiques> est forfait pour l'Open d'Australie. Elle est toujours en convalescence après sa blessure.
</nouvelles>
⇝ la Belge Kim Clijsters est forfait pour l'Open d'Australie.

4) *Analyse syntaxique.* On effectue une analyse syntaxique de chacun des textes obtenus. Les critères de validation sont différents selon la classe :

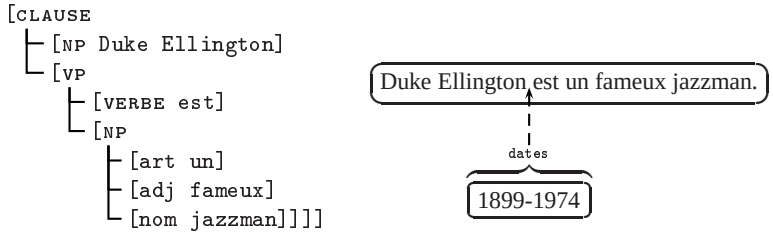
- Pour la transparence ($C = T$), on considère que l'analyse est valide si un même regroupement syntaxique englobe le contenu d'un élément e_n et du texte situé autour de cet élément (rappelons que si ces exemples sont en français, le travail s'est effectué sur de l'anglais) :

(14) <gras>Duke Ellington</gras> est un fameux jazzman.



- Pour le saut ($C = S$), l'analyse est valide si un même regroupement syntaxique englobe du texte situé de part et d'autre de l'emplacement de l'élément e_n.

(15) Duke Ellington <dates>1899-1974</dates> est un fameux jazzman.



5) *Validation statistique.* Si la proportion de textes validés par l'analyseur syntaxique est supérieure à un seuil s, alors on considère que le type n représente des balises de la classe C.

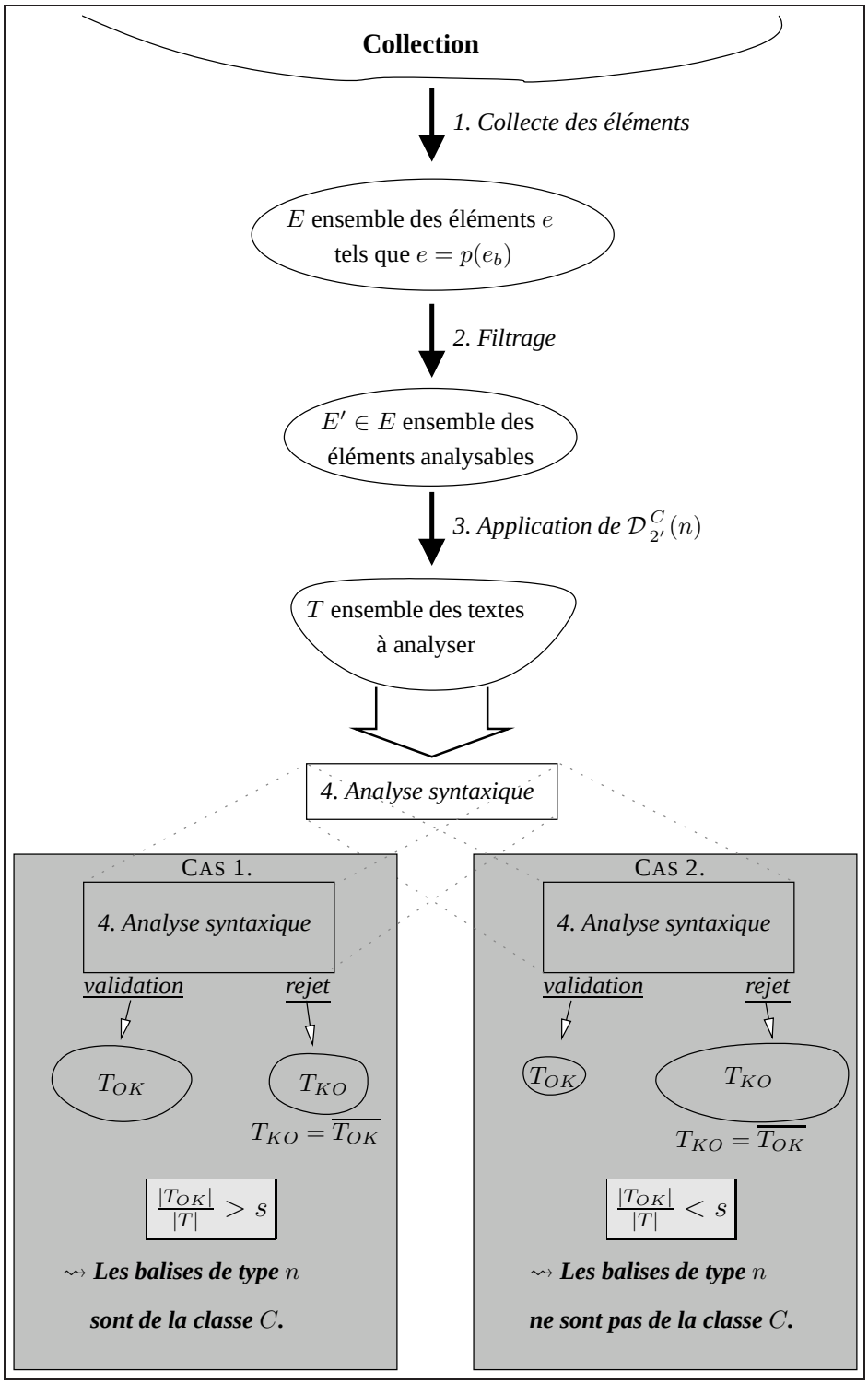


Figure 6 – Attribution d'une classe C à un type de balise n.

Cette procédure est illustrée par la figure 6. En pratique, pour chaque type n, on effectue d'abord l'analyse pour la classe des balises transparentes. Si on parvient à la conclusion que n fait partie de cette classe, on arrête l'analyse[4]. Dans le cas contraire, on applique l'algorithme pour les balises de saut. À défaut, on conclut que les balises n sont des balises dures.

4.1.4. *Expérimentations*

Les expérimentations que nous avons menées concernent la langue anglaise. La collection utilisée est celle d'INEX 2004 (Malik et al., 2005), comprenant 12 000 documents scientifiques en XML. La structure de ce corpus représente aussi bien du balisage logique, comme les sections, les paragraphes, les titres, que des balises de présentation. Au total, 192 types d'éléments différents sont présents dans la DTD.

4.2. *Résultats*

4.2.1. *Commentaires préliminaires*

4.2.1.1. Seuil.

Comme nous l'avons expliqué, nous estimons que les chances pour que notre analyse syntaxique valide des phrases pour des balises ne répondant pas en réalité à la définition sont très faibles. En revanche, les raisons pour lesquelles l'analyseur peut rejeter des phrases correspondant à la définition sont nombreuses. C'est pourquoi nous avons choisi d'utiliser un seuil s qui peut sembler très bas : 20 %[5]. Nous verrons qu'il suffit amplement lorsque la collection est suffisamment grande.

4.2.1.2. Balises dures contre le reste du monde.

Nous avons montré qu'un élément devait avoir un contenu mixte (texte + élément e_n) pour pouvoir être analysé. Pour de nombreux types de balises, cette configuration n'arrive jamais dans la collection. C'est le cas des balises logiques comme `'bdy'` (*body*, le corps du texte), `'sec'` (section), `'p'` (paragraphe), `'bib'` (bibliographie), `'bm'` (*back matter*, la partie finale), mais aussi `'au'` (auteur), `'atl'` (titre), `' abs'` (*abstract*, le résumé) et les éléments comme les listes ou les cellules de tableaux. Conformément à l'algorithme mis en place, ces balises sont donc classées comme *dures*, ce qui semble bien correspondre à la réalité.

4.2.1.3. Notations.

Les notations suivantes sont utilisées dans l'exposé de nos résultats :

– N = nombre total d'apparitions d'éléments de type n dans la collection ;

4. La raison de cet ordre est que les balises transparentes sont plus nombreuses que les balises de saut.
5. Ce seuil a été fixé après examen des résultats obtenus sur une petite partie du corpus et a confirmé sa validité par la suite.

– $p_f(C, n)$ = pourcentage de ces éléments conservé après le filtrage (C représentant la classe étudiée ($C \in \{T, S\}$) ;

– $s_f(C, n)$ = pourcentage des éléments conservés dont l'analyse syntaxique valide la définition $\mathcal{D}_{2'}^{C}$;

– $s_t(C, n)$ = rapport entre le nombre de balises validées et le nombre total de balises dans la collection ($s_t = s_f \times p_f$).

Étant donné le nombre élevé (192) des différents types de balises contenus dans la collection, il serait trop long de détailler ici les résultats pour chacun d'entre eux. Nous nous focalisons ici sur les résultats intéressants. Comme nous l'avons dit, les éléments structurants comme les paragraphes, les sections, les listes, les tableaux, les figures, et les données comme les auteurs ou les titres, ne franchissent pas la barrière du filtrage et sont directement classés dans les éléments durs.

4.2.2. *Balises transparentes*

Le tableau 1 donne des résultats pour l'algorithme de recherche des balises *transparentes*. Dans la première colonne, avec le nom de la balise, figure entre parenthèses la « sémantique » donnée par les commentaires de la DTD. La colonne la plus importante est donnée en caractères gras ($s_f(T, n)$), elle indique le taux des éléments traités (après filtrage) dont le système a validé la correspondance avec la définition. Pour les balises d'emphase[6], la plupart des scores se situent entre 40 et 70 % d'analyses syntaxiques correctes après filtrage.

Un élément de type `'math'` introduit un environnement mathématique, et `'super'` et `'sub'` sont uniquement utilisés dans ce contexte. Les résultats pour ces types de balises ne sont pas aussi clairs que les autres[7] ; nous avons déjà fait remarquer qu'on pouvait les considérer comme une exception dans la classification.

Les résultats obtenus pour les balises `'a'` et `'ref'` sont particulièrement intéressants. La première représente des liens vers des adresses Internet. Il ne s'agit pas d'une balise d'emphase, mais elle n'interrompt pas le contexte de lecture :

(16) The <a *href="http:...">*complete survey results showed that ...
(Les résultats complets de l'étude ont montré que ...)

Enfin, `'ref'` regroupe à lui seul les liens vers les références bibliographiques, les figures, les notes de bas de page et autres. Son mode d'utilisation est caractéristique des balises de saut ; son taux pour les balises transparentes est proche de zéro.

---

6. Nous appelons balises d' « emphase » tout balisage *procédural* et toute marquage de *présentation* (voir Coombs et al. (1987)). Tous les éléments concernés par la présentation du texte, indépendamment de sa structure, sont compris dans cette catégorie. Nous ne faisons pas dans ce cas la distinction entre balisage de présentation physique et logique.
7. Ceci étant probablement dû au fait que Cass ne parvient pas à analyser correctement son contenu

Type (n)	N	$p_f(T,n)$	$s_f(T,n)$	$s_t(T,n)$
a (link)	91	60.44 %	**74.55 %**	45.05 %
ariel	5800	42.38 %	**67.98 %**	28.81 %
b (bold)	160171	38.03 %	**61.40 %**	23.35 %
bi (emphasis)	4233	12.93 %	**58.32 %**	7.54 %
bu (emphasis)	206	29.61 %	**31.15 %**	9.22 %
bui (emphasis)	53	13.20 %	**42.86 %**	13.21 %
it (italics)	1070877	32.94 %	**60.46 %**	19.92 %
large	1	100.00 %	**100.00 %**	100.00 %
math	76270	61.06 %	**17.84 %**	10.89 %
ref (hyperlink)	395946	79.21 %	**2.86 %**	2.27 %
rm (roman)	3548	42.45 %	**49.60 %**	21.05 %
scp (small caps)	114255	52.62 %	**72.70 %**	49.22 %
ss (typeface)	4402	62.49 %	**50.97 %**	31.85 %
sub (subscript)	291661	19.64 %	**21.80 %**	4.28 %
super	44567	31.18 %	**28.93 %**	9.02 %
tt (typeface)	47517	64.45 %	**59.72 %**	38.45 %
u (underlined)	2713	30.48 %	**37.97 %**	11.57 %
ub (medium bold)	2	50.00 %	**100.00 %**	50.00 %
url	25050	54.32 %	**48.39 %**	26.28 %

Tableau 1 – Résultats des recherches de balises transparentes.

Type (n)	N	$p_f(S,n)$	$s_f(S,n)$	$s_t(S,n)$
math	76270	61.06 %	**47.68 %**	29.11 %
ref (hyperlink)	395946	79.21 %	**35.86 %**	28.40 %

Tableau 2 – Résultats des recherches de balises de saut.

4.2.3. *Balises de saut*

Le tableau 2 contient des résultats obtenus pour la procédure des balises de saut. Nous n'avons conservé que les balises `'ref'` et `'math'`, qui sont les seules ayant reçu un score non nul à cette étape (rappelons que les balises ayant été validées comme balises transparentes n'ont pas continué le processus). Les scores sont dans ce cas moins tranchés que les précédents. Pour les références (`'ref'`), seuls 36 % des textes ont été validés par l'analyseur, ce qui suffit cependant à classer ce type dans la classe de saut.

4.2.4. *Résultats finals*

Avec un seuil de 20 %, nous obtenons en fin de compte les résultats suivants pour la classification des balises. La comparaison manuelle des résultats avec le rôle sémantique des balises est notre seul moyen d'évaluation :

– Balises transparentes :

- `'tt'` (typewriter font) et `'ss'` (sans serif) : polices de caractères ;
- `'b'` (bold) et `'ub'` (medium) : graisse des caractères ;
- `'it'` (italics) et `'rm'` (roman) : apparence des caractères ;
- `'scp'` (small caps) : petites capitales ;
- `'u'` (underlined) : souligné ;
- `'sub'` (subscript) et `'super'` (superscript) : modes indice et exposant ;
- `'large'` : taille des caractères ;
- `'ariel'`, `'bi'`, `'bu'`, `'bui'` : apparence des caractères ;
- `'a'` (href) et `'url'` : liens vers des URL.

– Balises de saut :

- `'ref'` (hyperlink) : références diverses ;
- `'math'` : environnement mathématique.

– Balises dures : toutes les autres...

5. L'outil XGTagger

XGTagger est une interface générique traitant le texte contenu par les documents XML. Il ne fonctionne pas seul, mais englobe n'importe quel système S d'analyse textuelle souhaité par l'utilisateur. À partir d'une liste de balises transparentes d'une part, de balises de saut d'autre part[8], il analyse le document XML et recompose les contextes de lecture (texte pur). Il fournit ce texte en entrée du système S. Après exécution du programme, il récupère sa sortie, l'analyse et recompose le document initial, enrichi par les données ajoutées par S (voir figure 7). La seule contrainte importante concernant le système S est qu'il doit reproduire le texte original d'une façon ou d'une autre dans sa sortie.

XGTagger est conçu pour éviter toute perte d'information, et l'opération est totalement réversible, le document initial pouvant être récupéré grâce à une simple feuille de style. Les paragraphes suivants proposent quelques exemples d'utilisation de ce logiciel générique : analyse morphosyntaxique, lexicale, syntaxique ou simple regroupement des contextes de lecture. Le fonctionnement pratique de l'outil est détaillé dans Tannier et Garnier (2005).

8. Ce logiciel n'intègre pas encore la détection automatique.

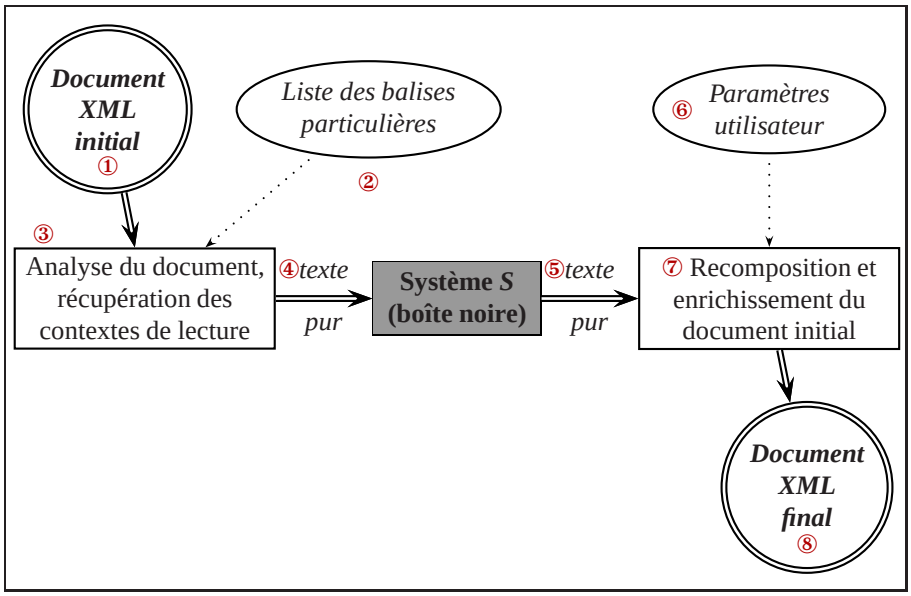


Figure 7 – Schéma de fonctionnement de XGTagger. Le processus est réversible, c'est-à-dire qu'il est possible de retrouver le document initial à partir du document final.

5.1. *Analyse morphosyntaxique*

Voici un exemple d'analyse morphosyntaxique réalisé avec XGTagger, interfacé avec l'analyseur TreeTagger (Schmid, 1994) (donc système $\mathcal{S}$ = TreeTagger).

① Soit le document XML suivant :

```
<article>
    <titre>Visitez I<pc>stanbul</pc></titre>
    <par>
        Cette ancienne capitale de trois empires<note>Istanbul traversa les
        empires romain, byzantin et ottoman</note> est maintenant en
        <gras>Turquie</gras>.
    </par>
</article>
```

② 'note' est une balise de saut, 'gras' et 'pc' des balises transparentes.

③ Avec l'aide de cette information, XGTagger reconstitue les contextes de lecture et insère des points entre chacun d'entre eux, « forçant » ainsi le système à ne pas les mélanger. Ces points seront retirés à la fin du processus. Le texte constitué par XGTagger et donné en entrée de TreeTagger est donc :

④
```
Visitez Istanbul . Cette ancienne capitale de trois empires est
maintenant en Turquie. . Istanbul traversa les empires romain,
byzantin et ottoman
```

⑤ Sortie de TreeTagger (présentée en deux colonnes) :

Visitez	VER:pres	visiter	Turquie	NAM	Turquie
Istanbul	NAM	Istanbul	.	SENT	.
.	SENT	.	.	SENT	.
Cette	PRO:DEM	ce	Istanbul	NAM	Istanbul
ancienne	ADJ	ancien	traversa	VER:simp	traverser
capitale	NOM	capitale	les	DET:ART	le
de	PRP	de	empires	NOM	empire
trois	NUM	trois	romain	ADJ	romain
empires	NOM	empire	,	PUN	,
est	VER:pres	être	byzantin	ADJ	byzantin
maintenant	ADV	maintenant	et	KON	et
en	PRP	en	ottoman	ADJ	ottoman

⑥ Les paramètres utilisateur précisent le format de la sortie du système $\mathcal{S}$, notamment le séparateur de mots (ici le retour chariot), le séparateur de champs (ici les espaces ou la tabulation), le champ correspondant au mot initial (ici le premier) et la sémantique des champs importants (ici la catégorie – *cat* – est dans le deuxième champ et le lemme – *lem* – dans le troisième).

⑦ La sortie du système $\mathcal{S}$ et ces paramètres sont utilisés pour recomposer le document initial et l'enrichir avec les informations apportées par l'analyse.

⑧ La sortie finale est donnée à la figure 8. Notons que les identifiants (‘`id`’) permettent de reconstituer les contextes de lecture : deux identifiants consécutifs correspondent au même contexte de lecture (voir autour de l'élément ‘`note`’). De plus, deux balises ayant le même identifiant représentent un même mot (par exemple l'identifiant 2).

Ainsi XGTagger peut permettre de faire le lien entre les documents XML et des formats d'annotation morphosyntaxique utilisant XML, comme MAF (Clément et de la Clergerie, 2004).

5.2. *Enrichissement lexical*

On peut également utiliser un système $\mathcal{S}$ ajoutant des informations à certains mots du texte. Par exemple, un enrichissement à base de synonymes ou une traduction des noms dans diverses langues :

① Document XML :

```
<proverbe>De gustibus et coloribus non disputandum</proverbe>
```

⑤ Exemple de sortie possible pour $\mathcal{S}$:

```
<article>
     <titre>
          <w id="1" cat="VER:pres" lem="visiter">Visitez</w>
          <w id="2" cat="NAM" lem="Istanbul">I</w>
          <pc>
               <w id="2" cat="NAM" lem="Istanbul">stanbul</w>
          </pc>
     </titre>
     <par>
          <w id="4" cat="PRO:DEM" lem="ce">Cette</w>
          <w id="5" cat="ADJ" lem="ancien">ancienne</w>
          <w id="6" cat="NOM" lem="capitale">capitale</w>
          <w id="7" cat="PRP" lem="de">de</w>
          <w id="8" cat="NUM" lem="trois">trois</w>
          <w id="9" cat="NOM" lem="empire">empires</w>
          <note>
               <w id="16" cat="NAM" lem="Istanbul">Istanbul</w>
               <w id="17" cat="VER:simp" lem="traverser">traversa</w>
               ...
               <w id="24" cat="ADJ" lem="ottoman">ottoman</w>
          </note>
          <w id="10" cat="VER:pres" lem="être">est</w>
          <w id="11" cat="ADV" lem="maintenant">maintenant</w>
          <w id="12" cat="PRP" lem="en">en</w>
          <gras>
               <w id="13" cat="NAM" lem="Turquie">Turquie</w>
          </gras>
          <w id="14" cat="SENT" lem=".">.</w>
     </par>
</article>
```

Figure 8 – Document XML après étiquetage morphosyntaxique.

```
De
gustibus/goût/Geschmack, Vorliebe
et
coloribus/couleurs/Farbe
non
disputandum/débattre, discuter/diskutieren
```

⑥ Options : 2^e champ = `français` ; 3^e champ = `allemand` ; délimiteur de champs : ‘/’. ⑧ La sortie est présentée à la figure 9.

```
<proverbe>
      <w>De</w>
      <w français="goûts" allemand="Geschmack, Vorliebe">gustibus</w>
      <w>et</w>
      <w français="couleurs" allemand="Farbe">coloribus</w>
      <w>non</w>
      <w français="débattre, discuter" allemand="diskutieren">disputandum</w>
</proverbe>
```

Figure 9 – Enrichissement lexical de document XML avec XGTagger.

5.3. *Analyse syntaxique*

Une option particulière de XGTagger permet d'effectuer (toujours par l'intermédiaire du système « boîte noire » $\mathcal{S}$) des analyses dépassant le cadre du mot.

Nous prenons ici rapidement l'exemple de l'analyse syntaxique, en imaginant par simplicité un système $\mathcal{S}$ qui regroupe les groupes nominaux *GN* formés de noms séparés par une préposition (comme *« boson de Higgs »*).

Ainsi, pour l'élément suivant :

```
<phrase>Le futur accélérateur LHC pourrait permettre d'observer le boson de <a href="http://fr.wikipedia.org/wiki/Higgs">Higgs</a> dans quelques années.</phrase>
```

Il est possible, grâce aux identifiants, de retrouver le groupe nominal malgré la présence de la balise `'a'`, ce qui est illustré par l'exemple de la figure 10. Un exemple complet d'utilisation de cette option est proposé dans Tannier et Garnier (2005).

```
<phrase>
      ... <w id="8">observer</w>
      <w id="9">le</w>
      <w id="10">boson</w>
      <w id="10">de</w>
      <a href="http://fr.wikipedia.org/wiki/Higgs">
            <w id="10">Higgs</w></a> ...
</phrase>
```

Figure 10 – Document XML après analyse syntaxique.

Cependant XGTagger n'est pas initialement conçu pour le traitement des groupes de mots ; ainsi, il n'est pas possible en l'état d'attribuer des valeurs supplémentaires à l'intérieur des groupes et de conserver la totalité des informations concernant, par exemple, des arbres ou des structures de traits. L'utilisation de cette technique reste donc limitée.

5.4. *Regroupement des contextes de lecture*

Un dernier exemple d'application consiste en un système qui se contenterait de retourner en sortie le texte fourni en entrée (éventuellement avec une séparation de la ponctuation). Le résultat est que les mots sont insérés dans des balises et que, grâce aux identifiants des éléments, les contextes de lecture sont regroupés et les mots coupés réassemblés. Cette opération peut servir de prétraitement avant une indexation des documents, ou avant des recherches prenant en compte la proximité logique (voir tous les exemples précédents, en supprimant les attributs des balises `'w'`).

6. Conclusion

Dans cet article, nous avons étudié les problèmes posés par la souplesse de la structure pour l'analyse du contenu des documents XML. À travers la notion de contexte de lecture, puis la classification automatique des balises, et enfin le logiciel XGTagger, nous avons apporté des réponses théoriques et pratiques à ces problèmes[9].

Tout cela a pour but de permettre aux utilisateurs de documents XML de conserver tous les avantages des balises, que ce soit au niveau de la structuration ou de la présentation, tout en ayant accès au texte contenu dans ces documents avec la même facilité que pour les documents plats.

Par ailleurs, il apparaît que les schémas des documents XML étant conçus comme une aide à la compréhension des documents, les utiliser et les interpréter en conjonction avec les techniques linguistiques, plutôt que les contourner pour pouvoir appliquer ces techniques, comme cet article le propose, semblent également être une orientation fructueuse à plus long terme. Ceci reste cependant difficile à envisager dans l'immédiat pour une application générale[10].

7. Bibliographie

Abiteboul S., Manolescu I., Nguyen B., Preda N., « A Test Platform for the INEX Heterogeneous Track », *in* Fuhr et al. (2005), p. 358-371, 2005.

Abney S., « Partial Parsing via Finite-State Cascades », *Journal of Natural Language Engineering*, vol. 2, n° 4, p. 337-344, 1996.

Aït-Mokhtar S., Lux V., Banik E., « Linguistic Parsing of Lists in Structured Documents », *Proceedings of the EACL Workshop on Language Technology and the Semantic Web (3rd Workshop on NLP and XML, NLPXML-2003)*, Budapest, Hungary, April, 2003.

9. Il est possible de télécharger gratuitement le logiciel XGTagger à l'adresse http://rim.emse.fr/Downloads/XGTagger/index.html

10. L'auteur tient à remercier les relecteurs anonymes de cet article. Leurs remarques pertinentes tant sur la forme que sur le fond ont permis d'en améliorer le contenu.

Clément L., de la Clergerie E. V., « MAF: a morphosyntactic annotation framework », *Proceedings of 2nd Language and Technology Conference (LT'05)*, Poznan, Poland, p. 90-94, April, 2004.

Colazzo D., Sartiani C., Albano A., Ghelli G., Manghi P., Lini L., Paoli M., « A Typed Text Retrieval Query Language for XML Documents », *Journal of American Society for Information Science and Technology (JASIST), Special Topic Issue on XML and Information Retrieval*, vol. 53, n° 6, p. 467-488, April, 2002.

Coombs J. H., Renear A. H., DeRose S. J., « Markup Systems and the Future of Scholarly Text Processing », *Communications of the ACM*, vol. 30, n° 11, p. 933-947, 1987.

Douglas S., Hurst M., Quinn D., « Using Natural Language Processing for Identifying and Interpreting Tables in Plain Text. », *Proceedings of the Fourth Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, Nevada, USA, p. 535-546, 1995.

Fuhr N., Großjohann K., « XIRQL: A Query Language for Information Retrieval in XML Documents », *in* W. Croft, D. Harper, D. Kraft, J. Zobel (eds), *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM Press, New York City, NY, USA, p. 172-180, September, 2001.

Fuhr N., Lalmas M., Malik S., Szlàvik Z. (eds), *Advances in XML Information Retrieval. Third Workshop of the Initiative for the Evaluation of XML retrieval (INEX)*, vol. 3493 of *Lecture Notes in Computer Science*, Springer-Verlag, Schloss Dagstuhl, Germany, 2005.

Goldfarb C. F., « Introduction to Generalized Markup », *Proceedings of the ACM SIGPLAN-SIGOA Symposium on Text Manipulation*, ACM Press, New York City, NY, USA, New York City, NY, USA, p. 68-73, 1981.

Goldfarb C. F., *The SGML Handbook*, Oxford University Press, 1991.

Ide N., Veronis J. (eds), *The Text Encoding Initiative: Background and Context*, Kluwer Academic Publisher, 1995.

Lini L., Lombardini D., Paoli M., Colazzo D., Sartiani C., « XTReSy: A Text Retrieval System for XML documents », *in* D. Buzzetti, H. Short, G. Pancalddella (eds), *Augmenting Comprehension: Digital Tools for the History of Ideas*, Office for Humanities Communication Publications, King's College, London, 2001.

Malik S., Lalmas M., Fuhr N., « Overview of INEX 2004 », *in* Fuhr et al. (2005), p. 1-15, 2005.

Renear A., Dubin D., Sperberg-McQueen C. M., Huitfeldt C., « Towards a Semantics for XML Markup », *Proceedings of the 2002 ACM Symposium on Document Engineering*, ACM Press, New York City, NY, USA, McLean, Virginia, USA, p. 119-126, 2002.

Schmid H., « Probabilistic Part-of-Speech Tagging Using Decision Trees », *International Conference on New Methods in Language Processing*, September, 1994.

Tannier X., Garnier A., XGTagger, a generic interface for analysing XML content, Technical Report n° 2005-400-008, Ecole Nationale Supérieure des Mines de Saint-Etienne, 2005.

van Zwol R., Wiering F., Dignum V., « The Utrecht Blend: Basic Ingredients for an XML Retrieval System », *in* Fuhr et al. (2005), p. 140-152, 2005.

W3C, « Semantic Web. World Wide Web Consortium (W3C). », 2001a. http://www.w3.org/2001/sw/.

W3C, « XML Schema (W3C) », 2001b. http://www.w3.org/XML/Schema/.

W3C, « Extensible Markup Language (XML). World Wide Web Consortium (W3C) Recommandation », 2004. http://www.w3.org/TR/2004/REC-xml-20040204/.