UNIVERSITÉ DU QUÉBEC À MONTRÉAL

EXTRACTION DE DONNÉES À PARTIR DU WEB

MÉMOIRE PRÉSENTÉ
COMME EXIGENCE PARTIELLE
DE MAITRISE EN INFORMATIQUE

PAR

BADR ACHIR

JUILLET 2013

REMERCIEMENTS

Mes premiers remerciements iront à Monsieur Aziz Salah, pour m'avoir soutenu durant mon projet de la maitrise. J'aimerais lui adresser mes plus vifs remerciements pour tout son dynamisme et ses compétences scientifiques qui m'ont permis de mener à bien cette étude. Ce travail n'aurait jamais pu aboutir sans lui, qui a toujours su me consacrer des moments de son temps, me guider et me conseiller, et me témoigner son soutien et sa confiance.

J'exprime ma profonde gratitude à ma famille pour leur encouragement et tout leur apport qui n'´était pas moindre.

Je tiens également à remercier mes professeurs du département d'informatique qui m'ont formé tout au long de ces années de maitrise ; particulièrement Monsieur Obaid Abdellatif qui a considérablement participé à mon initiation à la recherche.

Je remercie tout particulièrement les membres de mon jury de maîtrise, qui ont accepté de juger ce travail et de participer au jury.

J'adresse aussi mes très sincères remerciements à Madame Lise Arsenault secrétaire du programme de la maîtrise et doctorat en informatique pour ses conseils et ses encouragements durant ma formation.

Je clôture par une salutation destinée aux membres du laboratoire de recherche sur les technologies du commerce électronique (LATECE) de l'UQÀM.

TABLE DES MATIERES

LISTE DES FIGURES

LISTE DES TABLEAUX

RÉSUMÉ

Le Web est devenu riche en informations circulant à travers le monde entier via le réseau Internet. Cela a provoqué l'expansion de grandes quantités de données. De plus, ces données sont souvent non structurées et difficiles à être utilisées dans des applications Web. D'une part, l'intérêt des utilisateurs pour l'exploitation de ces données a augmenté d'une façon concurrentielle. D'autre part, les données ne sont pas faciles à être consultées par l'humain. Cet intérêt a motivé les chercheurs à penser à des approches d'extraction des données à partir du Web, d'où l'apparition des adaptateurs.

Un adaptateur est basé sur un ensemble des règles d'extraction définissant l'emplacement des données dans le document à extraire. Plusieurs outils existent pour la construction de ces règles. Notre travail s'intéresse au problème de l'extraction de données à partir du Web. Dans ce document, nous proposons une méthode d'extraction des données à partir du Web basée sur l'apprentissage machine pour la construction des règles d'extraction. Les résultats de l'extraction de notre approche démontrent une importance en matière de précision d'extraction et une meilleure performance dans le processus d'apprentissage. L'utilisation de notre outil dans une application d'interrogation de sources de données a permis de répondre aux besoins des utilisateurs d'une manière très simple et automatique.

Mots clés : extraction, adaptateurs, règles d'extraction, apprentissage machine, Web, applications Web.

INTRODUCTION

De nos jours, l'information sur le web est disponible avec de grosse quantité ce qui rend son traitement très difficile et complexe. Ainsi, avec l'apparition du web les utilisateurs ont bénéficié d'un accès à de multiples informations appartenant aux différentes sources. Ces informations sont souvent complètes et disponibles en grandes masses. Cependant, le web est basé sur un paradigme de navigation qui rend très difficile la récupération et l'intégration des informations dans des applications. Plusieurs outils ont été utilisés pour remédier à ce problème, la plus récente est l'utilisation des agents d'informations basés sur des requêtes de base de données (e.g WHIRL [W. COHEN, 1998], Ariadne [C.A. Knoblok et al., 1998], [T. Kirk et al., 1995]. Afin de répondre au problème de la récupération des informations à partir du web, ces approches ont utilisé un ensemble de requêtes de base de données. Ces requêtes sont applicables sur un certain nombre de sites web pré-spécifiés. L'accès à une telle information se fait par l'exécution d'une requête appropriée.

Les agents de traitement d'information sont généralement fondés sur des adaptateurs qui permettent d'extraire l'information à partir des pages web semistructurées. Chaque adaptateur est constitué d'un ensemble de règles d'extraction et d'un logiciel pour les appliquer. En effet, le problème de la génération d'un adaptateur est souvent lié au processus de la construction des règles d'extraction. Plusieurs outils existent qui permettent de guider l'utilisateur dans la construction des règles d'extraction. Parmi ces outils :

- Les langages de programmations [V. Crescenzi and G. Mecca, 1998].

- L'arbre hiérarchique et le parseur automatique Html-awar [V. Crescenzi et al., 2001].

- Le traitement en langage naturel NLP [S. Soderland, 1999].

- Le modèle de description de format [B. Adelberg, 1998].

- Les ontologies [D.W. Embley et al., 1999].

- Et l'apprentissage machine [N. Kushmerick, 1997].

La première apparition des adaptateurs s'était à base des méthodes manuelles pour la construction des règles d'extraction. Dans ces méthodes, les règles d'extraction doivent être définies par l'être humain en se basant sur les langages de programmation comme Java ou Perl, ainsi que l'outil de l'ontologie. Dans ce type d'approches, l'intervention de l'être humain est indispensable alors que le processus de la définition des règles d'extraction exige un haut niveau de précision et il deviendra très difficile lorsqu'un domaine d'application nécessite un grand nombre de sources où le formatage des données change assez souvent.

Pour pallier au problème de la construction manuelle des règles, les méthodes semi-automatiques sont apparues. En effet, différentes approches ont été réalisées, parmi lesquelles [Ion Muslea et al., 1998]. Dans ce travail, les auteurs ont développé un algorithme d'induction générique (algorithme de STALKER) pour générer les règles d'extraction. Ils ont introduit la notion d'analyse lexicale pour définir un document sur le web. Ils ont considéré un document comme un ensemble de jetons qui représente le code html propre au document. En se basant sur cet ensemble de jetons, ils ont représenté le contenu de ce document sous forme d'un arbre hiérarchique (*embeded catalog tree*) ECT.

Suivant la représentation de l'arbre ECT proposée, le processus d'apprentissage des règles d'extraction consiste à calculer la règle d'extraction propre à chaque nœud en se basant sur la notion de délimiteurs. Dans l'approche STALKER une règle d'extraction est composée de deux délimiteurs, d'un délimiteur gauche caractérisant le début de l'information à extraire et d'un autre délimiteur droit caractérisant la fin de l'information à extraire. Un délimiteur gauche est calculé à partir du processus d'apprentissage des exemples d'entrainements qui précédent l'information à extraire, ainsi que le délimiteur droit est le résultat d'apprentissage des exemples d'entrainement qui succèdent l'information à extraire. On comprend bien que dans cette approche le contenu de l'information à extraire n'est pas utilisé dans l'apprentissage. Dans des documents Web pauvres en texte, le processus d'apprentissage

revient très difficile dans cette approche et parfois impossible. L'approche SOFTMEALY est apparue pour pallier à ce problème, dans cette approche, l'information à extraire est caractérisé par deux séparateurs, d'un séparateur gauche qui représente le résultat d'apprentissage des exemples d'entrainement précédant l'information à extraire, et d'un séparateur droit représentant l'information à extraire. Dans des sites Web dont les éléments à extraire ont le même type d'information, le séparateur droit n'est pas suffisant pour caractériser la fin de l'information à extraire. Dans ce travail, nous avons répondu au problème du manque d'expressivité du délimiteur de l'approche STALKER, ainsi de celui du séparateur introduit par SOFTMEALY. Nous nous sommes servis des notions traitées par les deux approches, nous avons utilisé la notion du délimiteur ainsi celui du séparateur.

Nous avons étudié le processus d'extraction pour l'algorithme de STALKER dans l'optique de l'améliorer en introduisant les techniques de l'approche du système SOFTMEALY. Cette dernière a le mérite d'utiliser le type de l'attribut à extraire dans l'opération d'extraction des données. Nous avons implémenté l'algorithme d'extraction et nous l'avons testé sur des sites populaires afin de valider nos résultats.

Nous avons aussi utilisé le processus d'extraction pour l'intégration des données pour construire une base d'informations complémentaire.

Ce mémoire est composé de 5 chapitres. Dans le premier chapitre nous présentons une esquisse de quelques travaux déjà réalisés traitant du processus de l'extraction. Ce chapitre contient également une présentation de quelques projets de recherche reliés à ce domaine. Par la suite, dans le deuxième chapitre, nous présentons le processus et le fondement algorithmique d'extraction d'information à partir du web. Ensuite, dans le troisième chapitre, nous décrivons l'implémentation et l'analyse des résultats de notre algorithme. Dans le quatrième chapitre, nous expérimentons l'algorithme d'extraction de STALKER et nous l'intégrons dans une application de base de données. Enfin dans le dernier chapitre, nous terminons par une conclusion.

CHAPITRE I

ÉTAT DE L'ART

1.1 Introduction

La popularité et le développement du Web ont mené à l'expansion de la quantité d'information disponible. Ainsi, de multiples sources d'informations en ligne sont apparues. D'autre part, le nombre d'utilisateurs navigant sur le Web augmente continuellement. Ceci a rendu les méthodes traditionnelles de recherche d'informations, telles que la navigation et la recherche d'information par mots clés, de plus en plus imprécises, souvent incapables de trouver l'information exacte sur le Web, et non adaptées au traitement automatique de ce gigantesque lot d'informations.

Dans cette section, nous nous intéressons au problème de l'extraction de l'information provenant du Web. Le but derrière le processus de l'extraction est de définir un accès automatique aux données. Le fait de rendre l'accès automatisé à des informations provenant de multiples sources ouvre la porte pour la réutilisation de ces données dans des domaines distincts. Ces données peuvent être intégrées pour répondre à des besoins de l'utilisateur. Prenons par exemple un site permettant d'afficher les prix de certains produits appartenant à la ville de Montréal, un autre site publiant les mêmes produits mais pour la ville de Paris en France, et un autre publiant les prix de ces produits pour la ville d'Athènes en Grèce. En appliquant l'extraction automatique sur les différents sites, on obtiendra 3 différents sous-ensembles d'informations de même type mais venant de régions différentes. Il est donc possible de les intégrer dans la conception d'un nouveau site pour faire une comparaison de prix de tels produits.

D'un autre côté, les données extraites pourraient être utilisées par des agents intelligents afin de répondre à des requêtes de base de données (interrogations). Supposons par exemple, un site Web A publiant le nom, l'adresse, le téléphone, la description, et le type de la carte de crédit acceptée pour l'acquittement du paiement de certains restaurants appartenant à une région. Un autre site B faisant la publication pour les mêmes restaurants du précédent site mais avec des informations différentes, telles-que le décor, le prix, le nom, l'adresse et le numéro de téléphone. L'extraction des informations de la source A et B, la sauvegarde de celles-ci dans une base de données et l'interrogation de ces dernières pourra répondre à de multiples requêtes. Citons par exemple, l'ensemble de restaurants portant le nom X dont le prix de repas ne dépasse pas 20$, tous les restaurants qui acceptent la carte visa, etc.

En ignorant la tâche de l'automatisation du processus d'extraction, l'utilisateur se trouve face au problème de trouver l'information qui répond à ces besoins. Il devra procéder à l'interrogation de plusieurs sources contenant l'information désirée. Cette interrogation se fait à travers un ou plusieurs formulaires de la source d'information. Une fois rempli, le formulaire permet de construire une requête dont l'exécution engendre une réponse qui prend la forme d'une ou plusieurs pages Web. Généralement, ces pages sont destinées à être lisibles par l'utilisateur humain, et dans la plupart des cas leurs contenus portent plein d'informations. Cependant, il est compliqué d'utiliser ces informations pour d'autres besoins car elles ne sont pas destinées à être exploitables par un ordinateur. En effet, les données provenant du Web suivent certaines régularités de mise en forme, et sont parfois disponibles sous forme de listes ou de tableaux pour les rendre plus claires et lisible pour l'utilisateur. En se basant sur la manière dont les données sont présentées sur le Web, et suivant les régularités de mise en forme qu'elles prennent (ces données dites semi-structurées), il est possible de procéder à leur extraction en se basant sur leurs structures. L'objectif du processus de l'extraction d'information à partir du Web est de construire des programmes capables d'extraire des informations à partir d'une source et de les sauvegarder dans un format structuré pour qu'elles soient consommables par la machine. Chacun de ces programmes est applicable sur une seule source de données ou sur un domaine bien spécifique, et il est adapté

pour qu'il soit capable d'extraire les informations provenant de différentes pages Web de la même source.

Le programme permettant de générer un ensemble de règles d'extractions, de les expérimenter, et de les appliquer sur un document Web est appelé adaptateur.

Dans la littérature, nous distinguons 2 types d'extractions, l'extraction d'informations à partir des pages Web de données semi-structurées, et celui de l'extraction d'information provenant des textes décrits en langage naturel. Dans ce qui suit, nous nous intéressons au problème de l'extraction d'informations du Web, à cet effet, nous essayons de présenter le problème de la construction d'adaptateur pour ce type d'extraction.

Le début des années 90 a connu une concurrence dans les recherches reposant sur le problème de l'extraction provenant du Web. Cependant, chacune de ces approches utilisent des méthodes différentes. Cette distinction réside dans la nature des données utilisées en entrée et le résultat de l'adaptateur généré en sortie.

Les premiers adaptateurs étaient construits manuellement. Dans ces approches, les règles d'extraction sont écrites par le concepteur humain. À cet effet, plusieurs outils ont été proposés afin de faciliter au concepteur la tâche de la construction des règles.

La construction manuelle d'un adaptateur exige un haut niveau de précision et elle suppose que l'utilisateur soit un expert dans certains langages de programmation tels que Java ou Perl pour qu'ils soient capable de créer les règles d'extraction. La tâche de la création des règles d'extraction peut devenir fastidieuse et très difficile à réaliser lorsqu'un domaine d'application nécessite un grand nombre de sources ou lorsque le format des données change au fur et à mesure dans le temps.

Afin de répondre à ce problème, plusieurs approches sont apparues visant l'automatisation du processus de construction des règles. Chacune de ces méthodes utilise un outil assistant approprié pour la construction des règles d'extraction.

1.2 Construction manuelle d'adaptateur

De multiples approches ont été proposées décrivant chacune un système permettant l'accès aux données provenant de sources Web, et l'extraction d'informations à partir de ces sources, en appliquant un ensemble de règles d'extraction.

1.2.1 Méthodes basées sur un fichier de description de regles

Dans [J. Hammer et al., May 1997], les auteurs ont conçu un programme d'extraction d'information à partir de pages Web semi-structurées. Ce programme est configurable et il utilise en entrée un fichier de spécifications permettant de définir l'emplacement des données dans la page Web (source html), et comment ces données sont-elles stockées dans les objets.

Ce fichier est composé de plusieurs commandes où chacune d'elle a la forme d'un triplé [variables, source, motif]; source représente le texte considéré contenant les informations à extraire, variables l'ensemble des éléments à extraire, et motif les délimiteurs à appliquer pour extraire les différentes variables à partir de leur source. Le fichier de la figure 1.1 représente une spécification de l'extraction des informations de la prévision des températures dans certaines villes. Chaque commande de la liste permet d'extraire une information pour la récupérer dans une variable réutilisable par la commande suivante. Prenons par exemple la première commande lignes [1-4] permettant d'extraire le contenu de la page Web et de l'enregistrer dans la variable root, ensuite la seconde commande lignes [5-8] permet d'appliquer le motif "*<TABLE*<TABLE*</TR>#</TABLE>*" sur la source root pour extraire des nouvelles informations. Ces informations seront stockées dans la nouvelle variable temperatures, le symbole # représente la partie à extraire. La troisième commande lignes [9-12] permet de découper la source temperatures en fragments de texte selon le séparateur '<TR ALIGN=left>', ensuite la quatrième commande lignes [13-16] enregistre l'ensemble de fragments de texte dans le tableau city_temp.

Finalement, la dernière commande lignes [17-20] applique le motif "*<TD>#</TD>*HREF=#>#</A>*<TD>#</TD>*<TD>#/#</TD>*<TD>#</TD>*<TD>#/#*" sur les différentes cellules du tableau afin d'extraire les informations élémentaires désirées (country,c_url,city,weath_tody,hgh_tody,low_today,weath_tomorrow,hgh_tomorrow,low_tomorrow).

Cette méthode suit un processus hiérarchique récursif résidant dans le fait que l'extraction d'une telle information atomique nécessite l'extraction successive de l'ensemble des parties engendrant celle-ci.

```
1 [["root",
2   "get('http://www.intellicast.com/weather/europe/')",
3   "#"
4  ],
5  ["temperatures",
6   "root",
7   "*<TABLE*<TABLE*</TR>#</TABLE>*"
8  ],
9  ["_citytemp",
10  "split(temperatures,'<TR ALIGN=left>')",
11  "#"
12 ],
13 ["city_temp",
14  "_citytemp[1:0]",
15  "#"
16 ],
17 ["country,c_url,city,weath_tody,hgh_tody,low_today,weath_tomorrow,hgh_tomorrow,low_tomorrow",
18  "city_temp",
19  "*<TD>#</TD>*HREF=#>#</A>*<TD>#</TD>*<TD>#/#</TD>*<TD>#</TD>*<TD>#/#*"
20 ]]
```

Figure 1.1 Fichier de spécification d'extraction [J. Hammer et al., May 1997]

1.2.2 Méthode basée sur la description

[W. May, G. Lausen, 2004], les auteurs ont défini un système qui met en œuvre un ensemble de Frameworks intégrés pour explorer les données provenant du Web. Ils ont

considéré un Framework permettant l'extraction d'information, un autre pour intégrer ces informations dans d'autres applications (la médiation), et un autre pour les interroger afin de répondre à un certain nombre de requêtes. Dans ce papier, l'intégration s'appuie sur la conception d'une architecture qui fait l'intégration des fonctionnalités des trois *Framework* précédents. Ils ont considéré le contenu d'une page Web comme une unité d'information représentée sous forme d'un modèle orienté objet semi-structuré. Ce modèle est doté d'une base de règles décrites en langage la F-logic [M. Kifer et al., May 1995]. Dans ce travail, les auteurs ont proposé une méthode pour la maintenance des informations provenant du Web. Par contre, ils ont ignoré la phase permettant de décrire comment les règles d'extraction peuvent être produites.

1.2.3 TSIMMIS

Dans l'approche [J. Hammer et al., 1997], les auteurs ont conçu un système TSIMMIS permettant d'utiliser en entrée un fichier de spécification comme celui défini dans l'approche [J. Hammer et al., May 1997] afin de générer l'adaptateur. Ce système est considéré parmi les premiers systèmes manuels aidant l'utilisateur à la construction d'adaptateur. Ce système a introduit 2 nouveaux opérateurs très importants "CASE" et "SPLIT". L'opérateur "CASE" permet de fragmenter la liste d'attributs en entrée en plusieurs attributs élémentaires, le second a pour rôle d'aider l'utilisateur à lever l'ambiguïté dans les pages à extraire. L'opérateur "CASE" permet de vérifier si le document à extraire ne présente pas des exceptions pour éviter des erreurs durant le processus d'extraction. Le fait de fragmenter la liste en attributs permet de résoudre le problème de l'extraction d'attribut à plusieurs valeurs. Ainsi, la levée de l'ambiguïté a aussi permis de notifier les sources contenant des attributs manquants.

1.2.4 MINERVA

[V. Crescenzi et al., 2001]. Dans cette approche, les auteurs ont développé un système MINERVA dont l'écriture des règles d'extraction est basée sur une grammaire régulière de style EBNF[1]. L'objectif du travail et d'essayer d'introduire dans la règle d'extraction des procédures permettant de lever les exceptions au moment de l'application de cette dernière

[1] http://www.waterproof.fr/products/PHPEdit/manual/fr/module.FindRegExp.html

sur une source Web dont la structure n'est pas conforme à la grammaire. Le rôle de ces procédures d'exception est d'éviter d'extraire des données non désirées par l'utilisateur. Cet avantage permet d'avoir des résultats d'extraction pertinents.

1.2.5 W4F

Dans [A. Saiiuguet et F. Azavant, 2001], les auteurs ont développé un système W4F qui est un kit à outils java permettant à l'utilisateur de parcourir le contenu html d'un document Web. La construction d'adaptateur consiste en trois couches indépendantes. La couche récupération qui a pour rôle de récupérer les documents à partir du Web via un protocole de communication http. Les documents récupérés sont alors sélectionnés, puis transformés sous forme d'un arbre suivant le modèle DOM via le parseur html. La deuxième couche qui est la couche d'extraction permet l'extraction des données du document en appliquant les règles d'extraction correspondantes sur l'arbre obtenu durant la couche récupération. Ces données seront après enregistrées dans le système W4F sous la forme des d'arbres de chaînes de caractères hiérarchiques NSL. La troisième couche est la couche de mappage, elle consiste en la construction de règles d'extraction suivant le modèle d'arbre NSL construit précédemment. La construction d'une règle d'extraction est basée sur l'arbre NSL ainsi que le chemin qui mène à l'attribut à extraire. Pour augmenter l'expressivité des règles d'extraction, les auteurs ont introduit quelques expressions régulières inspirées du langage de programmation Perl. D'autres opérateurs ont été aussi utilisés pour permettre de répondre au problème de la hiérarchie, de l'attribut à de multiples valeurs, et de la décomposition.

Les méthodes manuelles traditionnelles s'intéressaient seulement à exposer la solution d'utiliser les règles d'extraction. D'autre part, d'autres approches manuelles ont proposé des outils afin de faciliter l'écriture des règles. Ces approches n'ont pas offert des méthodes pour automatiser le processus du calcul des règles d'extraction, mais ont juste guidé le programmeur dans leur construction.

1.3 Automatisation de la génération d'adaptateur

Les approches manuelles ne permettent pas l'automatisation de la construction des règles d'extraction. Ainsi, ces approches ont été limitées dans leur utilisation sur une portion de sites Web. Cependant, plusieurs travaux ont été suscités pour remédier à ce problème.

Parmi ces travaux, il y a ceux qui sont basés sur des méthodes d'apprentissage semi-automatiques dites supervisées, d'autres ont utilisé des méthodes semi-supervisées pour la génération d'adaptateur, et d'autres se sont basés sur des méthodes automatiques non supervisées. Dans ce qui suit, nous allons présenter une revue de littérature des travaux reliés au problème de l'automatisation de l'adaptateur.

1.3.1 Extraction d'information basée sur un apprentissage supervisé

Ces méthodes dites supervisées semi-automatiques nécessitent une phase d'étiquetage manuel de quelques pages exemples à utiliser pour la phase d'apprentissage. Celle-ci permet la construction des règles d'extraction généralisées. Pour induire de nouvelles règles d'extraction, la procédure compare les chaînes de caractères précédant et succédant les attributs de la donnée à extraire. Elle apprend donc les délimiteurs et les motifs d'extraction pour chaque attribut.

1.3.1.1 WIEN

Wien est le premier système d'induction d'adaptateur [N. Kushmerick. 1997] développé en 1997 à l'Université de Washington par Kushmeric. Ce système permet de traiter à la fois les pages Web et les textes libres. Kushmeric a considéré le problème d'extraction comme une réponse à une requête. À cet effet, et à partir d'un ensemble de pages Web exemples étiquetées représentant des réponses liées à une requête fictive, l'algorithme d'induction permet de trouver une hypothèse. Cette hypothèse est un adaptateur candidat permettant d'extraire les données à partir des pages Web similaires de celles utilisées comme exemples. Il a défini six classes différentes d'adaptateurs afin d'exprimer les structures des sites Web. Sa première apparition était avec la classe LR (left, right) dont l'objectif était de déterminer le début et la fin de chaque attribut à extraire du document. Il a représenté un adaptateur sous la forme d'un modèle relationnel à 2 k-uplets (l1, r1,...,lk, rk) où li, ri représentent respectivement le délimiteur gauche et le délimiteur droit d'un attribut i représentant une donnée à extraire. De plus, Kushmerick a introduit des contraintes sur les délimiteurs :

Un délimiteur de début doit être un suffixe propre c'est-à-dire ne doit pas se présenter plusieurs fois dans une chaîne de caractère, ainsi que ce délimiteur ne doit pas figurer dans la fin de la page à extraire.

Un délimiteur de fin ne doit pas être une valeur d'un attribut à extraire, et ne doit pas faire partie de chaque valeur d'un attribut i.

En se basant sur ce formalisme, l'algorithme d'induction cherche à apprendre l'adaptateur pour qu'il soit généralisé pour des autres pages similaires, l'arrêt de l'apprentissage est repéré par certaines heuristiques introduites par Kushmerick. Cette méthode est limitée dans le fait d'être appliquée seulement sur les sources qui ont une structure tabulaire, en plus, un adaptateur LR permet d'extraire un seul type de tuple par page. Pour lever cette limite Kushmerick a étendu le système WIEN en introduisant d'autres classes d'adaptateurs.

[N. Kushmerick et al., 1997], les auteurs ont défini une nouvelle classe d'adaptateurs HLRT en intégrant 2 nouveaux paramètres au modèle relationnel précédent. Supposant un tuple M comprenant K attributs, dans cette approche, un adaptateur est représenté sous la forme d'une relation de (2k+2)-uplets (h, l_1, r_1,..., l_k, r_k, t) où h représente le début du fragment de texte du tuple M, l_i, r_i représentent le délimiteur gauche, respectivement droit de l'attribut i, et t représente la fin du tuple M.

La classe HLRT a donné plus d'expressivité à l'adaptateur du fait qu'elle permet aussi l'extraction de plusieurs types de tuples dans une page. Suivant le même principe, plusieurs d'autres classes d'adaptateurs du système WIEN ont été proposées. En combinant ces différentes méthodes, l'approche a pu répondre à 70% des sources utilisées [N. Kushmerick, 1997]. Le schéma de la figure 1.2 illustre la hiérarchie des classes d'adaptateurs proposées dans le cadre du système WIEN suivant leur expressivité. La méthode BELR est une extension de la méthode LR en introduisant deux nouveaux paramètres, B qui représente le début du tuple et E qui représente la fin du tuple. HBELRT est une combinaison des méthodes HLRT et BELR. Dans ce système, les auteurs ont fusionné les quatre paramètres vus dans ces deux méthodes avec ceux de la méthode LR. La direction des flèches dans le

schéma exprime l'augmentation de l'expressivité du délimiteur d'une méthode par rapport à l'autre.

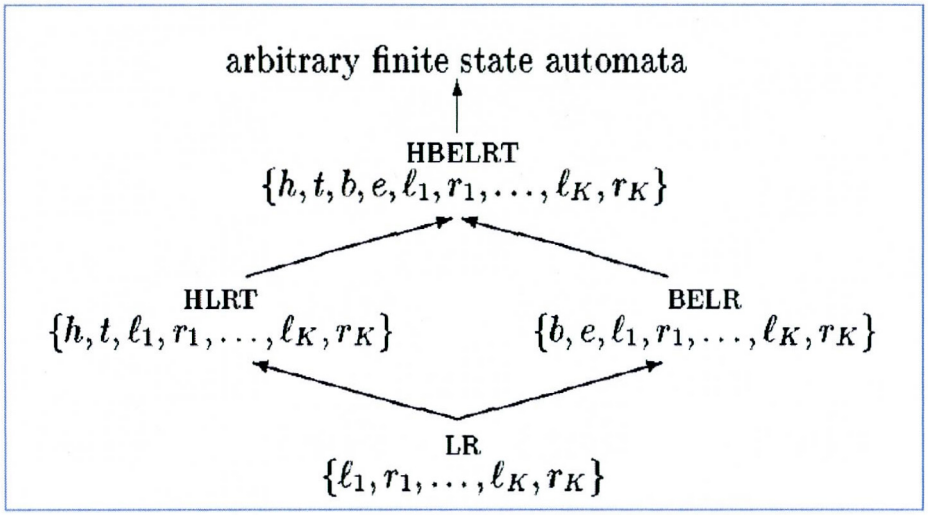


Figure 1.2 Les classes d'adaptateur dans WIEN [Nicholas Kushmerick et al., 1997]

Malgré cela, le système WIEN dans ses différentes variations ne permet pas de traiter les documents ayant une structure hiérarchique. D'autre part les attributs d'un même tuple doivent être ordonnés et toujours présents.

Toutefois, les données contenues dans une source Web ne sont pas souvent ordonnées, et généralement se présentent sous un schéma hiérarchique, ce qui a ouvert les portes à d'autres approches plus intéressantes tentant de relever ces défis.

1.3.1.2 STALKER

STALKER [I. Muslea et al., 1997, 1998, 1999] est un système d'extraction des données des pages Web tenant compte de leur représentation hiérarchique. En s'inspirant de l'analyse lexicale du code html de la source, et pour simplifier le problème d'extraction

dans des documents dont le contenu est hiérarchique, Muslea et al., ont proposé le formalisme d'arbre hiérarchique ECT (embeded catalog tree). Chaque nœud de cet arbre représente un ensemble de tags défini dans un ensemble d'alphabet Σ. L'arbre hiérarchique ECT permet de décrire la décomposition logique du document. Son avantage est d'optimiser le processus d'extraction dans des documents hiérarchiques non tabulaires. Cet arbre est composé de nœuds, où chacun d'eux pourrait être :

Une feuille : Un élément primitif qui représente une information à extraire dans l'arbre.

Une liste : Un nœud comprenant 0 ou 1 fils.

Un tuple : Un nœud comprenant 1 ou plusieurs fils.

Il est à noter bien que l'ensemble de fils d'une liste ont le même type d'information, par contre ceux d'un nœud tuple n'ont pas forcément le même type d'information. La figure 1.4 illustre l'arbre hiérarchique ECT du document de la Figure 1.3.

En se basant sur le formalisme ECT, un ensemble de pages exemples étiquetées, ainsi sur la notion de délimiteur défini dans un alphabet Σ, l'algorithme d'apprentissage permet de trouver les règles d'extraction associées à chaque nœud de l'arbre ECT. Le contenu d'une règle est représenté par une disjonction de délimiteurs dont chaque délimiteur est une chaîne de caractère définie dans un ensemble généralisée de l'alphabet Σ. L'extraction d'un nœud se fait en appliquant les règles d'extraction appropriées sur l'occurrence de son nœud parent, ne règle dans STALKER est représentée par un automate à état finis dont le passage d'un état i vers un autre état j se fait si l'adaptateur reconnaît le délimiteur associé à la transition s(i, j).

Le système STALKER a montré une convergence sur des sources ayant une structure hiérarchique de plusieurs niveaux. Dans ce système, l'ordre des attributs d'un tuple n'est pas indispensable. De plus, les auteurs ont donné une importance à l'expressivité du délimiteur. Tous ces avantages ont aidé l'opérateur à extraire des informations provenant des sources Web complexes ayant un manque d'informations, et des fois, présentant des erreurs.

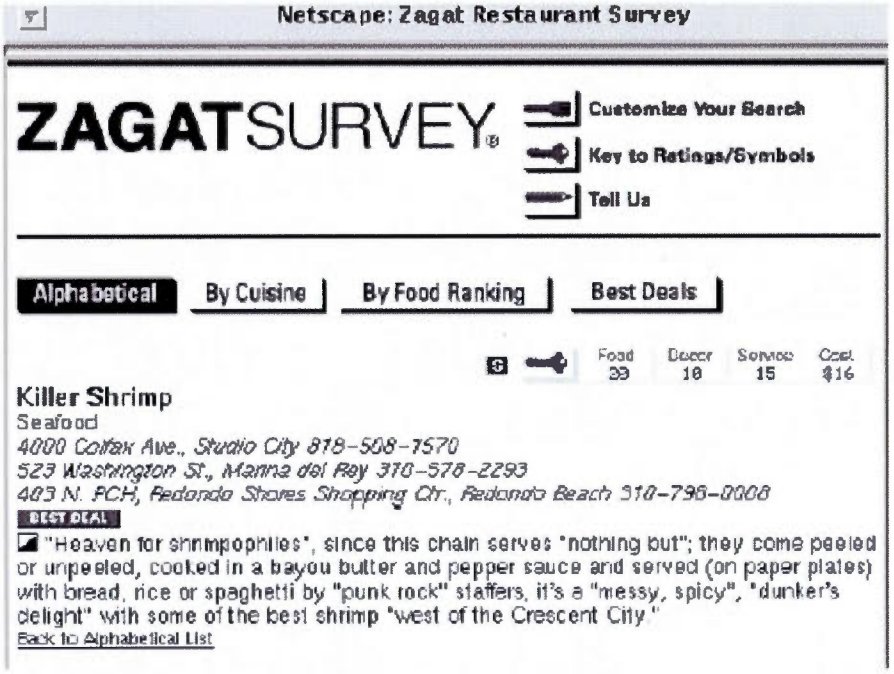


Figure 1.3 Page d'un document représentant un restaurant ZAGAT[2]

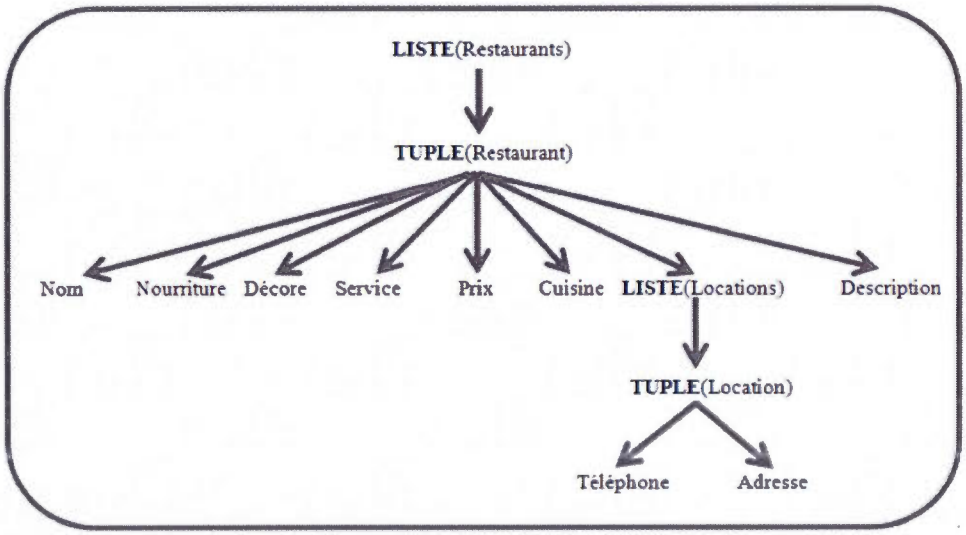


Figure 1.4 Arbre ECT représentant le document de la Figure 1.3

[2] http://www.isi.edu/integration/RISE/repository.html

D'autres méthodes ont été proposées afin de répondre aux problèmes des sources contenant des erreurs d'insertion, du manque d'information et la permutation des attributs.

1.3.1.3 SOFTMEALY

[C. Hsu, 1998] a défini un nouveau contexte pour l'adaptateur Web. L'objectif derrière est de trouver des règles d'extraction permettant de répondre au problème de l'ordre des attributs et des attributs manquants. À cet effet, ils ont considéré l'adaptateur sous la forme d'un transducteur à état fini FST qui consomme une chaîne de caractères en entrée en franchissant une certaine transition afin de produire une nouvelle chaîne en sortie. La chaîne consommée en entrée est l'alphabet d'entrée du transducteur, la chaîne produite est l'alphabet de sortie, et la transition représente la règle contextuelle de l'attribut. Le FST consiste en deux parties différentes, un FST permettant d'extraire le fragment du texte de la page Web contenant des tuples à extraire dit FST de corps, et un FST pour extraire les différents attributs d'un même tuple dit FST de tuple.

Du fait que la notion de délimiteur introduite dans les systèmes WIEN et STALKER n'était plus suffisante pour extraire des informations à partir d'un fragment du texte pauvre en caractères entre 2 attributs adjacents, Chun a tenté de donner plus de précision à un délimiteur, alors il l'a remplacé par un séparateur. Un séparateur d'après Chun est un ensemble de symboles représentant les extrémités gauche et droite d'un attribut. Le séparateur prend en compte l'ensemble des symboles composant l'attribut. Afin d'extraire un attribut, l'adaptateur reconnaît les séparateurs autour de cet attribut, ces séparateurs ne peuvent pas être les mêmes pour d'autres attributs de la page à extraire.

Un transducteur tuple est le corps d'adaptateur SoftMealy. Formellement, il est composé de 5-uplets (Σ_1, Σ_2, Q, R, E) où Σ_1 est l'ensemble de séparateurs d'entrée, Σ_2 est l'ensemble des caractères de sortie, Q est un ensemble fini d'états contenant :

Un état initial b, et un état final e.

k est un état s'il existe un attribut k à extraire.

Pour chaque attribut k, il existe un état -k correspond à un attribut factice représentant l'ensemble des symboles à consommer entre l'attribut k est son suivant.

R est l'ensemble des règles contextuelles dont chaque règle est dénotée par s(i, j) représentant la classe de séparateurs figurant entre l'attribut i et l'attribut j, incluant les attributs factices, ainsi que l'état initial b et l'état final e.

$E = Q \times R \times \Sigma_2^* \times Q$ est un ensemble de transitions, une transition de i vers j est possible s'il existe une transition $(i, r, o, j) \in E$ tel que le prochain séparateur d'entrée satisfait la règle r. Une transition de i vers j montre que les attributs i et j sont adjacents dans les pages Web cible lorsque leur séparateur répond à la règle contextuelle associée.

L'apprentissage permet de reconnaître l'ensemble des séparateurs en définissant leurs extrémités de début et celles de fin par l'ensemble des règles contextuelles. Il consiste à apprendre l'ensemble des transitions et leurs règles contextuelles associées, en commençant avec un ensemble d'exemples étiquetés, le processus de SOFTMEALY permet de généraliser les règles contextuelles pour qu'elles soient capables d'extraire le maximum d'éléments d'un même attribut.

Le processus du SOFTMEALY permet de modéliser le FST dont des permutations ou des changements d'ordre apparaissent. À partir d'un ensemble d'exemples étiquetés, et pour chaque instance d'attribut i, l'utilisateur calcule l'ensemble des classes de séparateurs pour les 2 extrémités gauche et droite. Le processus de SOFTMEALY refuse d'avoir un séparateur en commun entre les extrémités gauche et droite. Les séparateurs de gauche sont alignés à droite, et ceux de droite sont alignés à gauche. Ensuite, ils seront généralisés suivant quelques heuristiques introduites par Chun, puis après, les doublons des séparateurs seront supprimés. À la fin du processus, l'ensemble des séparateurs obtenu représente une règle contextuelle gauche, respectivement droite de l'attribut concerné i. La figure 1.6 ci-dessous illustre le transducteur FST modélisant l'exemple de la page Web étiquetée de la figure 1.5. La page de l'exemple représente un tuple de quatre attributs :

U : L'adresse url de la page d'auteur.

N : Le nom de l'auteur.

A : Le nom de l'académie.

M : Le nom de l'administrateur.

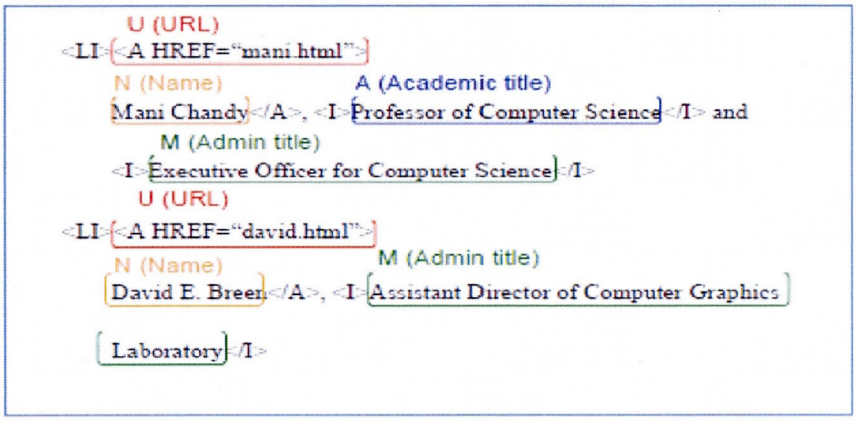


Figure 1.5 Exemple d'une page Web étiquetée

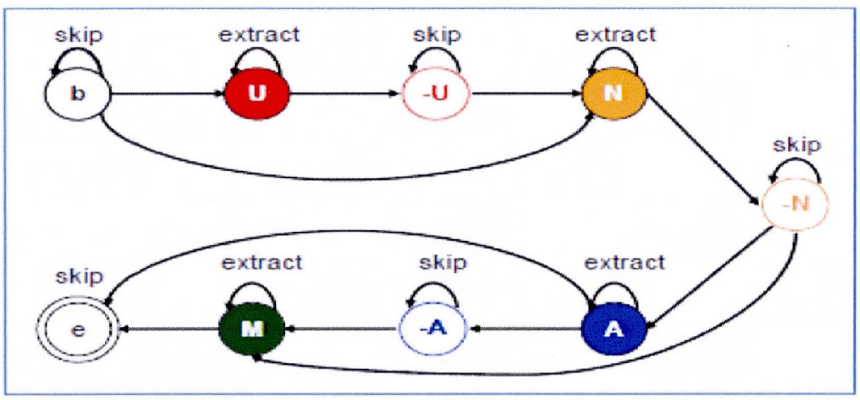


Figure 1.6 SOFTMEALY modélisant l'exemple de la figure 1.5

Avec l'utilisation des séparateurs l'approche SOFTMEALY a permis d'analyser sémantiquement le contenu d'un attribut à extraire, ceci est du par exemple dans le fait qu'un séparateur permet de spécifier le contexte de l'extrémité gauche par la définition du type de chaîne de caractères formant l'attribut. Cet aspect sémantique est vu comme capital avantage par rapport aux systèmes WIEN et STALKER.

1.3.2 Méthodes semi-supervisées pour la construction d'adaptateur

Une des questions importante dans l'apprentissage d'adaptateur est la phase manuelle de l'étiquetage des exemples d'entraînement. Pour assurer la précision d'apprentissage, un grand nombre d'exemples d'entraînement est nécessaire et peut prendre un temps très long. Dans cette section, on va présenter quelques travaux qui ont été basés sur des méthodes semi-supervisées pour la construction des règles d'extraction.

1.3.2.1 CO- TESTING

Une des approches est celle permettant de développer le processus de STALKER. Les auteurs ont tenté dans [I. Muslea et al., 2000] de performer la phase d'étiquetage des exemples. En fait, il est inutile d'étiqueter les exemples d'entraînement qui ne portent pas de nouvelles informations au moment de l'extraction. À cet effet, les auteurs ont développé une méthode appelée CO-TESTING permettant l'étiquetage des exemples d'une façon intelligente et automatique. Pour l'induction d'adaptateur, la méthode consiste à lancer l'induction des règles d'extraction en commençant par un nombre très réduit d'exemples d'entraînement. En appliquant les règles d'extraction résultat sur quelques pages de la source, si des données sont mal extraites alors les fragments de textes où ces données se trouvent seront étiquetés et ajoutés à la liste des exemples d'entraînement. De plus, ils ont défini 2 manières pour parcourir un exemple afin de localiser un élément à extraire. Ils ont considéré une règle permettant la consommation des jetons de l'exemple du début jusqu'à la fin, et d'autre permettant la consommation des jetons à partir de la fin de l'exemple vers le début. Selon ce concept, deux règles d'extractions différentes représentant le début sont associées à chaque élément à extraire.

Étant donné un ensemble d'exemples non étiquetés U, dans le cadre d'induction d'adaptateur, l'approche fonctionne comme suit :

1. Choisir au hasard un petit sous-ensemble L des exemples non étiquetés de U;
2. Étiqueter manuellement les exemples de L et U = U - L;
3. Apprendre l'adaptateur W avec l'ensemble L comme exemples d'entraînement;
4. Appliquer W sur U afin de trouver un ensemble d'exemples informatifs L;
5. Arrêtez si L = ø, sinon, passez à l'étape 2.

À partir d'un ensemble d'exemples non étiquetés U, en étiquetant un ensemble L contenant un minimum d'exemples parmi l'ensemble U, l'adaptateur est initialisé en apprenant les règles d'extraction sur l'ensemble L. L'adaptateur est appliqué sur un ensemble d'exemples de U non étiquetés, les deux types de règles seront appliquées afin d'extraire l'ensemble des éléments de ces exemples. Pour chaque exemple, les deux types de règles seront appliqués alors. Si ces deux règles ne permettent pas de localiser correctement les éléments à extraire, cet exemple doit être étiqueté par l'utilisateur et il est appelé un exemple informatif. Par contre, lorsque les deux règles à la fois localisent correctement le début de cet élément, l'extraction est très correcte et l'étiquetage de l'exemple n'est pas demandé. Ainsi, lorsque les deux règles ne donnent pas la même position dans l'exemple, l'une d'entre elles est supposée fausse.

1.3.2.2 IEPAD

[C.H. Chang et S. Lui, 2001]. Dans ce travail, les auteurs ont implémenté un système d'extraction d'information automatique nommé IEPAD. Ce système est l'un des premiers adaptateurs qui permettent de généraliser automatiquement les règles d'extraction sans pages Web étiquetées, ni exemples d'entraînement. Le fait que la majorité des pages provenant du Web suivent une certaine régularité syntaxique, les informations à extraire issues de ces pages peuvent avoir des motifs communs qui séparent les informations situant dans un même niveau hiérarchique. Sur la base de cette idée, les auteurs se sont basés sur le motif commun pour la généralisation des règles d'extraction. À cet effet, ils ont représenté la page Web sous forme d'un arbre syntaxique PAT[3] [C.H. Chang et S. Lui, 2001] en se basant sur son contenu html. Afin de faciliter la détection du motif commun, ils ont codé les balises html en binaire suivant une certaine classification introduite sur les différentes balises. Ensuite, un algorithme d'alignement est appliqué sur l'arbre PAT résultat pour permettre de découvrir le motif commun de longueur maximum. Le schéma de la figure 1.8 illustre l'encodage du document HTML d la figure 1.7. Les parties texte ne sont pas prises en considération dans le processus de classification, les auteurs s'intéressent seulement aux différentes balises HTML. En se

[3] Un arbre PAT d'une chaîne s est un arbre Patricia (Practical Algorithm to Retrieve Information Coded in Alphanumeric) construit sur l'ensemble suffixes de s.

basant sur la classification introduite par les auteurs, le tableau de la figure 1.1 présente le codage binaire du document HTML.

```
<html>
  <body>
     <h1>Les tarifs du distributeur</h1>
        <table>
             <tr><td>Café</td>
                <td>0.40 EUR (2.96 F)</td></tr>
             <tr><td>Soda</td>
                <td>0.75 EUR (4.92 F)</td></tr>
             <tr><td colspan="2"><hr></td></tr>
             <tr><td>Gâteau</td>
                <td>0.50 EUR (3.28 F)</td></tr>
        </table>
  </body>
</html>
```

Figure 1.7 Document HTML à extraire

```
HTML(<html>) HTML(<body>) HTML(<h1>) TEXT HTML(</h1>) HTML(<table>)
HTML(<tr>) HTML(<td>) TEXT HTML(</td>) HTML(<td>) TEXT HTML(</td>)
HTML(</tr>) HTML(<tr>) HTML(<td>) TEXT HTML(</td>) HTML(<td>) TEXT
HTML(</td>) HTML(</tr>) HTML(<tr>) HTML<td colspan="2">) HTML(<hr>)
HTML(</td>) HTML(</tr>) HTML(<tr>) HTML(<td>) TEXT HTML(</td>)
HTML(<td>) TEXT HTML(</td>) HTML(</tr>) HTML(</table>) HTML(</body>)
HTML(</html>)
```

Figure 1.8 Codage du document HTML de la figure 1.7

Symbole	Code
HTML(<html>)	0000
HTML(<h1>)	0010
HTML(</h1>)	0100
HTML(<tr>)	0110
HTML(</td>)	1000
HTML(<td	1010
HTML(</table>)	1100
HTML(</html>)	1110
HTML(<body>)	0001
TEXT	0011
HTML(<table>)	0101
HTML(<td>)	0111
HTML(</tr>)	1001
HTML(<hr>)	1011
HTML(</body>)	1101

Tableau 1.1 Codage binaire du document de la figure 1.8

Le système IEPAD ne nécessite pas un étiquetage des pages exemples. En effet, l'intervention de l'utilisateur permet uniquement de préciser les motifs d'extraction des enregistrements. Par ailleurs, l'extraction d'information dans IEPAD se limite uniquement sur des informations d'un même enregistrement ce qui est inapplicable dans le cas des données hiérarchiques. Ce système est utilisé seulement sur des pages Web contenant une liste d'enregistrements (au moins 2 enregistrements par page). Les résultats d'extraction de ce système ont montré que sur quatorze moteurs de recherche les plus populaires, les précisions d'extraction ont atteint 97% [C.H. Chang et S. Lui, 2001].

1.3.2.3 OLERA

[C.H. Chang et S. Kuo, 2004], dans ce travail, les auteurs ont développé un système OLERA qui ressemble au système IEPAD mais qui permet de lui ajouter une option permettant de répondre au problème de la non extraction des données provenant de pages Web ayant un seul enregistrement de données. Ce système permet la génération des règles d'extraction en commençant par un seul exemple qui représente un bloc des informations à extraire. Le principe est de trouver l'ensemble des blocs similaires à l'exemple 'entrainement,

ainsi à partir de cet ensemble, une généralisation de motif doit être faite pour permettre de localiser correctement les enregistrements de l'ensemble des pages d'entrainement.

À cet effet, le processus de la construction des règles d'extraction est composé de trois phases :

1- Codage de la page d'apprentissage : Cette phase consiste à traduire la page d'apprentissage sous forme d'un schéma facile à manipuler en se basant sur son contenu. La technique du codage permet de prendre en considération la hiérarchie de la page ainsi que quelques délimiteurs introduits par les auteurs. Les auteurs ont classifié le contenu de la page en plusieurs niveaux hiérarchiques, et pour chaque niveau, ils lui ont affecté un code significatif. Le tableau 1.1 ci-dessous montre un exemple de classification hiérarchique dans le système OLERA.

Category	Encoding scheme	Delimiters
Markup-level	Block-level tag	block-level tags
	text-level tag	text-level tags
Text-level	Paragraph	NewLine, CarriageReturn, Tab
	Sentence	Period, Question Mark, Exclamation Mark
Word-level	Phrase	Colon, Comma, Semicolon, Bracket, Quotation Mark
	Others	(,), $, -, /, @, Blank, etc.

Tableau 1.2 Classification et codage d'une page Web [C.H. Chang et S. Kuo, 2004]

2- Détection de motif : Dans cette phase, le système permet de trouver le motif du bloc utilisé comme exemple. En comparant ce motif avec d'autres pages d'entrainement, le système peut détecter les enregistrements similaires à ce bloc. La technique utilisée par les auteurs est l'algorithme de détection approximatif.

3- Alignement des enregistrements : Cette phase consiste à aligner l'ensemble des enregistrements découverts durant la phase de détection de motif en utilisant des différentes techniques d'alignement de chaîne de caractères. Le résultat est présenté sous la forme d'un tableau de m enregistrements et n slots. Le slot représente un délimiteur ou un code significatif introduit par les auteurs.

Le schéma de la figure 1.7 ci-dessous illustre en détail les trois opérations principales du système OLERA.

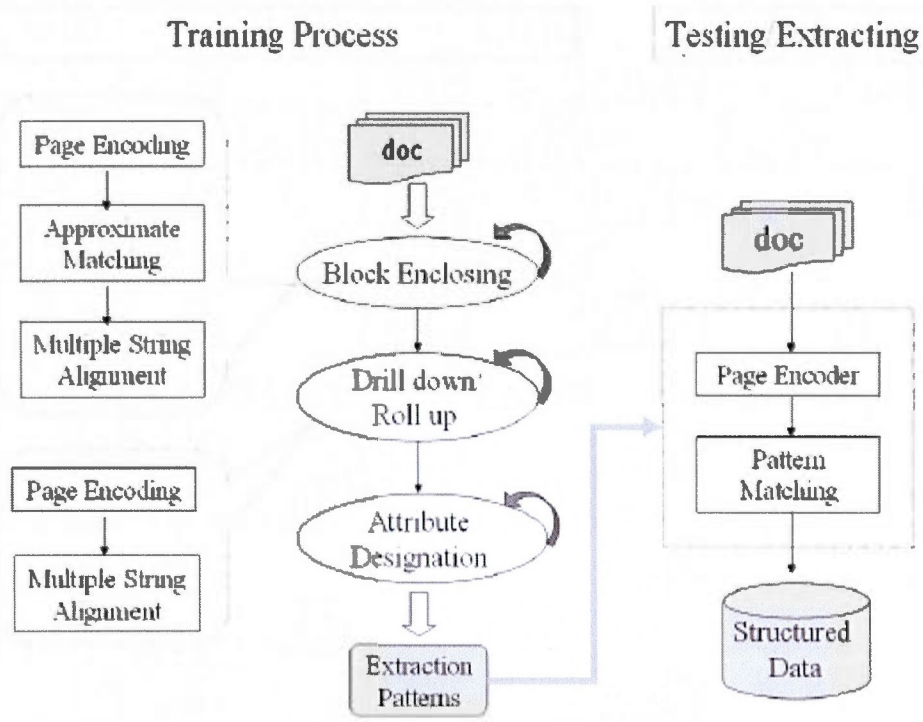


Figure 1.9 Système d'extraction OLERA [C.H. Chang et S. Kuo, 2004]

1.3.3 Méthodes non supervisées pour la construction d'adaptateur

1.3.3.1 DELA

[J. Wang et F.H. Lochovsky, 2002], les auteurs ont conçu le système DELA. C'est une extension du système IEPAD. Ce système a réussi d'extraire des données provenant d'objets imbriqués. Dans ce travail, les auteurs ont permis d'automatiser complètement le processus de la construction des règles d'extraction. L'intervention de l'utilisateur humain n'est pas nécessaire. Dans ce papier, les auteurs ont introduit deux techniques intéressantes, une qui permet d'identifier les régions contenant les données que l'on veut extraire, et l'autre a pour rôle d'identifier la structure des données dans ces régions. Le processus de la construction des règles est basé sur deux étapes importantes :

1- Algorithme d'extraction des régions : Cette étape consiste à comparer deux arbres DOM Représentant deux pages Web différentes d'un même site Web afin d'identifier les régions de textes contenant des informations à extraire. Il s'agit après d'ignorer les nœuds de l'arbre ayant le même sous arbre. L'algorithme utilisé est l'algorithme de détection des régions DES (Data rich-Section Extraction).

2- Extracteur du motif : C'est le composant principal de l'approche DELA, il permet de construire la représentation symbolique des arbres suffixes. Ensuite, il s'agit d'appliquer l'algorithme de la découverte de motif pour détecter les motifs des objets dans un même arbre suffixe. Cette technique a bien répondu au problème de données imbriquées. D'après les auteurs, le système DELA est considéré à 100% automatique. La précision d'extraction est souvent très élevée.

1.3.3.2 ROADRUNNER

[V. Crescenzi et al., 2001], les auteurs ont conçu un système nommé ROADRUNNER. ROADRUNNER permet d'extraire les données à partir des pages denses ayant une bonne régularité structurelle. Il compare la structure du code html d'un certain échantillon de pages Web afin de conclure le schéma des emplacements des données à extraire dans une page. La construction d'un adaptateur dans ROADRUNNER se réduit à un problème d'inférence d'une grammaire régulière sans union[4]. L'algorithme d'extraction produit une expression régulière qui accepte tous les attributs des pages de l'échantillon. La construction des règles est un processus complètement automatique dans ROADRUNNER. L'absence d'intervention de l'utilisateur dans ce processus est l'avantage principal de ce système.

1.3.3.3 Méthode d'annotation sémantique

Dans [M. Michelson et C.A. Knoblock, 2004], les auteurs ont présenté une méthode d'extraction d'information basée sur un algorithme d'annotation sémantique qui est considéré

[4] Une grammaire régulière sans union est une grammaire qui peut être décrite à l'aide d'une expression régulière avec les opérateurs habituels (*i.e.* _, +, ?, (,), |) sauf |. Une telle expression s'appelle une expression régulière sans union ou ERSU.

comme un pré-processus qui précède la phase de l'extraction. Cette approche vise des types de données non structurées. Elle a été appliquée sur des textes non structurés. Dans cette approche un tel texte provenant d'une source ayant un contenu non structuré est appelé ''message'' ou en anglais ''post''. Ces messages sont sans construction grammaticale alors il est difficile de procéder à l'extraction de leurs informations. Pour rendre le message facile à extraire, les auteurs ont introduit une liste de noms de référence. Ces noms représentent des chaînes de caractères ayant une sémantique et qui sont souvent utilisés dans la source pour un domaine précis. L'objectif de l'approche est d'adapter le texte en entrée à l'ensemble des noms de la liste des références. Cette adaptation permet de faciliter la détection des attributs dans le message. L'adaptation consiste à comparer le message texte avec l'ensemble des noms de la liste des références selon des métriques de similarité. Le résultat de l'adaptation donne lien à un message bien structuré et fragmenté selon la liste de référence. On appelle ce processus le lien d'enregistrement. Le schéma de la figure 1.10 ci-dessous illustre l'algorithme d'annotation sémantique sur un exemple d'un message contenant des informations sur un hôtel. Dans cet exemple, on remarque bien que la liste des références est composée de quatre noms ('Holiday Inn Sellect', 'University Center', 'Hyat Regency', et 'Downtown'). En se basant sur l'ensemble des attributs (name et area), et en comparant le message texte avec ces noms durant le processus de lien d'enregistrement, on remarque bien que l'algorithme a pu sélectionner les deux noms qui ont plus de similarité ('Holiday Inn Sellect', et 'University Center').

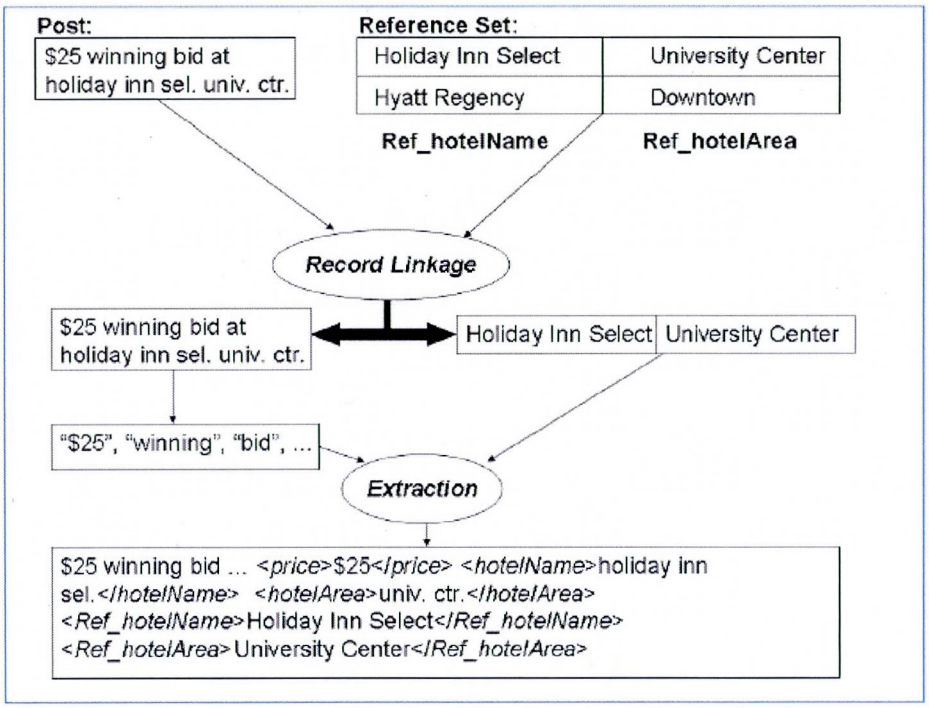


Figure 1.10 Algorithme d'annotation sémantique [M. Michelson et C.A. Knoblock, 2004]

Cette méthode a été implémentée sur le système *Phoebus*[5] et a été testé sur deux domaines d'informations hotel *postings*[6] et *comic books*[7]. Les résultats ont montré une donne précision d'extraction selon la pertinente de la liste des références. Pour extraire des informations provenant d'un tel domaine, et pour que les résultats de l'extraction soient correctes, la méthode nécessite que la liste de référence soit riche en noms représentant le domaine en question.

1.3.3.4 Construction des règles par généralisation de contexte

Benjamin HABEGGER dans [B. HABEGGER, 2004] présente une méthode basée sur la généralisation de contexte. Il s'agit d'utiliser en entrée un ensemble d'instances d'une relation à extraire. Ces instances sont choisies par l'utilisateur parmi l'ensemble de pages appartenant à une source de données. Les contextes de ces instances de la relation à extraire

[5] http://www.phoebus.co.uk

sont cherchés dans les pages sources. La généralisation des contextes de ces instances permet de construire des nouveaux motifs permettant d'extraire les informations de n'importe quelle page similaire dans la source. L'application de l'adaptateur sur les pages de la source permet alors d'obtenir les instances de la relation décrite par les exemples fournis par l'utilisateur. Cette méthode a montré des avantages par rapport aux autres méthodes non automatiques, dans lesquelles, la réadaptation de l'adaptateur ne nécessite pas un étiquetage des exemples même dans le cas où la mise en forme des pages a changé. Ceci est dû au fait que les instances extraites avant le changement du site seront utilisées pour construire le nouveau adaptateur. Cette méthode a été implémentée dans le système IEREL et a été évaluée sur des données de différents domaines tels que les moteurs de recherche, les annuaires en ligne et les sites de commerce électronique. Les résultats ont montré que souvent un adaptateur est construit en moins d'une seconde. Cette méthode est beaucoup plus pratique et a permis un gain du temps important durant l'apprentissage.

Les méthodes à base d'apprentissage non supervisé ont permis d'automatiser complètement le processus de la construction des règles d'extraction de l'adaptateur. Néanmoins, la précision d'extraction est liée au nombre et à la pertinence des exemples étiquetés, ainsi à l'outil introduit par chacune des approches. Cependant, les résultats ont montré que l'extraction des sources par un adaptateur construit à base des approches non supervisées est souvent moins précise par rapport à celles des approches supervisées [C.H. Chang et al., 2006].

1.4 Comparaison des méthodes d'extraction

Plusieurs études ont été élaborées pour comparer les différentes méthodes d'extraction qui existent dans la littérature. Dans cette section, on va essayer de présenter la différence entre ces méthodes d'extraction en nous basant sur quelques travaux réalisés.

[C.H. Chang et al., 2006], les chercheurs ont présenté une taxonomie pour les différentes méthodes d'extraction et ils les ont comparées sur la base de 3 dimensions: les tâches de domaine, le degré de l'automatisation de la méthode et la technique d'extraction utilisée. À cet effet, ils ont proposé un ensemble de critères pour mesurer la fiabilité et l'avantage de chacune par apport à l'autre. Les résultats ont montré que pour les méthodes

manuelles, les données extraites sont plus pertinentes mais nécessitent un haut niveau de programmation. En plus, Les méthodes manuelles ne requièrent pas une particularité au niveau de la structure de la page. N'importe quelle forme de page est acceptable en entrée, quel que soit un texte libre, une page Web semi-structurée ou un modèle de données structuré. Par contre, dans les méthodes d'extraction semi-supervisées et supervisées la structure interne des pages à extraire est très importante. Ces méthodes donnent des bons résultats pour des pages Web semi-structurées, et divergent souvent dans le cas des textes en langage naturel. La figure 1.11 montre la comparaison entre les différents types de pages d'entrée utilisée par chacune des approches. De plus, ces méthodes ne nécessitent pas l'intervention de l'utilisateur pour la création et la construction des règles d'extraction mais son intervention se limite seulement dans le marquage des exemples d'apprentissage. Les résultats ont montré que les méthodes d'apprentissage non supervisées permettent d'automatiser complètement le processus d'extraction. Dans ces méthodes, l'intervention de l'utilisateur est inutile et la génération de l'adaptateur se fait automatiquement. Par contre, ces méthodes sont limitées sur certaines structures de pages Web, et elles divergent souvent dans l'extraction de pages non conforme au modèle de données spécifié.

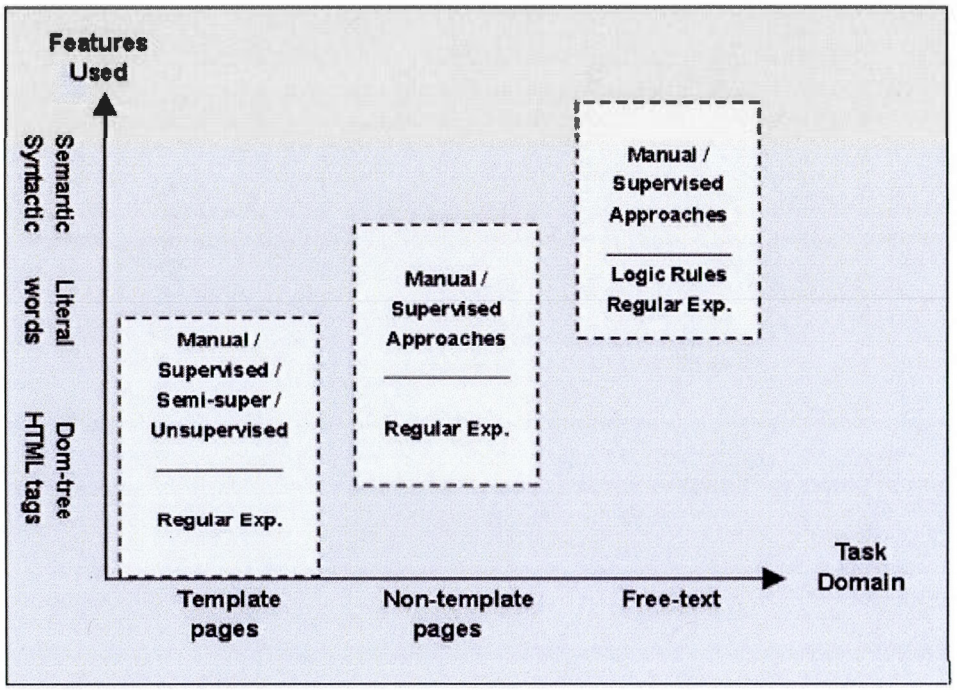


Figure 1.11 Comparaison entre les approches [C.H. Chang et al., 2006]

[H. Alberto et al., 2004], dans ce travail les auteurs ont présenté une taxonomie sur les différents outils d'extraction d'information à partir du Web. Ils se sont basés sur l'outil utilisé par chaque méthode pour la génération de l'adaptateur. À cet effet, ils ont considéré 5 outils qui existent et qui assistent au processus de la définition des règles d'extraction :

- Les langages de programmation : C'est une approche basée sur le codage suivant un langage de programmation spécifique tel que Java, Perl, ect.
- L'arbre syntaxique HTML : C'est une approche basée sur le parseur Html et l'arbre Html pour la modélisation du contenu de la page Web.
- Le traitement automatique en langage naturel : Il s'agit d'une analyse qui repose sur une identification de notions morphologiques, syntaxiques, et aussi sémantiques. L'analyse se fait souvent via un outil afin de faciliter les tâches.
- L'induction : C'est un processus basé sur l'apprentissage machine.

- L'ontologie : Est une approche qui permet la modélisation des informations sous la forme d'un graphe afin de leur accorder un sens bien défini. Cette modélisation est basée sur un ensemble de concepts sémantiques et d'héritage.

Afin d'évaluer les performances de chacune des approches, les auteurs ont introduit un ensemble de critères et mesures :

- Le degré de l'automatisation : Permet de mesurer l'intervention de l'être humain dans la phase de la création des règles d'extraction. Quand l'intervention de l'utilisateur est moins, le degré de l'automatisation de l'outil est élevé.
- la complexité de la structure de l'objet : Souvent, les données sur le Web suivent une structure complexe. On peut trouver des données composées de liste d'attributs, d'autres suivent une hiérarchie, ect.
- Le contenu de la page : Dans ce travail, les auteurs se sont basés sur 2 types de pages à extraire, une page Web semi-structurée et un texte libre semi-structuré.
- La facilité de l'utilisation de l'outil : Il s'agit de mesurer la complexité de l'outil en question.
- Sources non Html : Ce sont des documents provenant des sources non Html.
- La flexibilité : Il s'agit de mesurer le degré de la flexibilité de l'outil en question.

Le tableau 1.3 récapitule les résultats de l'analyse qualitative conclue par les auteurs. D'après ce tableau, on remarque que le degré de l'automatisation dans les méthodes automatiques basées sur les approches Html- aware est très élevé, ainsi que l'utilisation de l'outil est toujours facile pour l'utilisateur. Par contre, ces méthodes ne s'appliquent pas sur des documents provenant d'une source non Html et elles restent limiter sur les sources Html.

Pour les méthodes semi-automatiques basées sur l'approche d'induction, on remarque que l'outil est capable d'extraire des informations provenant de quelques textes non Html, ainsi qu'il supporte les objets complexes par exemple le cas du STALKER. Le degré de l'automatisation est moyen suite à l'intervention partielle de l'utilisateur pour l'étiquetage des exemples d'entrainement. On remarque aussi que les méthodes basées sur les approches de l'ontologie et les langages de programmation sont toujours manuelles et le support de

l'utilisation des objets complexes peut se faire via le codage de ces derniers. Par ailleurs, ces méthodes peuvent supporter les 2 types de textes citées ci-haut par exemple le cas de l'outil

BYU [D.W. Embley et al., 2006]. Dans les méthodes basées sur le traitement du langage naturel, le degré de l'automatisation est moyen et ces méthodes peuvent extraire des informations provenant des sources non Html. Par contre, ces méthodes n'acceptent pas les documents contenant des objets complexes.

Pour le cas des méthodes basées sur le modèle prédéfini, le degré de l'automatisation est moyen et l'utilisation de l'outil est facile pour l'utilisateur. Ces méthodes peuvent supporter l'extraction des informations provenant des sources autres que Html.

Tools		Degree of Automation	Support for Complex Objects	Ease of Use	XML Output	Support for Non-HTML Sources	Type of Page Contents
Languages	Minerva	Manual	Coding	I	Yes	Partial	SD
	TSIMMIS	Manual	Coding	I	No	Partial	SD
	Web-OQL	Manual	Coding	I	No	None	SD
HTML-aware	W4F	Semi-Automatic	Coding	+ +	Yes	None	SD
	XWRAP	Automatic	Yes	+ + + +	Yes	None	SD
	RoadRunner	Automatic	Yes	+ + + +	No	None	SD
NLP-based	WHISK	Semi-Automatic	No	+ +	No	Full	ST
	RAPIER	Semi-Automatic	No	+ +	No	Full	ST
	SRV	Semi-Automatic	No	+ +	No	Full	ST
Induction	WIEN	Semi-Automatic	No	+ +	No	Partial	SD
	SoftMealy	Semi-Automatic	Partial	+ +	No	Partial	SD
	STALKER	Semi-Automatic	Yes	+ +	No	Partial	SD
Modeling-based	NoDoSE	Semi-Automatic	Yes	+ I +	Yes	Partial	SD
	DEByE	Semi-Automatic	Yes	+ I +	Yes	Partial	SD
Ontology-based	BYU	Manual	Coding	+ +	No	Full	ST/SD

Tableau 1.3 Résumé de l'analyse qualitative [H.Alberto et al., 2004]

1.5 Contribution

Dans le cadre de ce travail, nous allons développer un algorithme d'extraction d'information, on va se baser sur les méthodes d'induction semi-automatique. À partir des résultats des études comparatives décrites dans la section 1.4 et en nous basant sur les approches à bases de l'apprentissage machine, on peut remarquer facilement que l'outil STALKER est le meilleur outil parmi ceux abordés dans ce cadre. À cet effet, dans la suite on va se concentrer sur la méthode de STALKER.

L'objectif de ce travail est d'améliorer l'algorithme de STALKER en introduisant quelques modifications au niveau de l'expressivité du délimiteur. Afin de donner plus d'expressivité au délimiteur utilisé dans l'approche STALKER, nous avons combiné le délimiteur de l'outil STALKER avec celui de séparateur utilisé dans le système SOFTMEALY. À cet effet, nous avons considéré aussi le contenu de l'information à extraire comme une chaîne de caractères à inclure dans l'apprentissage. Le processus d'apprentissage dans l'approche STALKER repose sur l'apprentissage des chaînes de caractères précédant et succédant l'information à extraire. Dans l'approche SOFTMEALY, le processus d'apprentissage est basé sur l'apprentissage de l'ensemble de chaines de caractères précédant l'information à extraire ainsi que sur le contenu de l'information elle-même. Notre approche couvre les avantages de ces deux méthodes, elle utilise à la fois les chaînes de caractères qui précédent et qui succèdent l'information à extraire ainsi que le contenu de l'information elle-même. Le fonctionnement de l'algorithme sera expliqué en détail dans le chapitre fondement algorithmique de l'extraction suivant.

CHAPITRE II

FONDEMENT ALGORITHMIQUE DE L'EXTRACTION

2.1 Introduction

Dans notre travail, nous nous basons sur une méthode d'induction pour la construction d'adaptateurs. Dans le chapitre précédent, nous avons fait une synthèse sur les différentes méthodes de construction d'adaptateurs. Dans le cas général, un adaptateur est un programme qui permet d'utiliser un ensemble de règles afin d'extraire quelques informations provenant de certaines sources Web. Nous avons constaté que le système STALKER est celui qui présente le plus d'avantages parmi les systèmes étudiés dans notre état de l'art et qui sont basés sur les méthodes d'induction. STALKER représente la source Web sous forme d'un arbre ECT. Cette représentation logique permet à STALKER de supporter des sources web ayant une structure hiérarchique. De plus, l'ordre des attributs dans l'arbre ECT n'est pas important ce qui implique que STALKER est insensible à l'ordre des attributs dans un tuple. L'apprentissage des règles d'extraction d'un nœud de l'ECT est indépendant de ces voisins. Par conséquent, STALKER n'est pas sensible au manque d'attributs dans un tuple du document source. Malgré ses avantages, STALKER n'exploite pas le format (numérique, date, adresse e-mail, etc.) des données à extraire contrairement à SOFTMEALY qui le fait partiellement. Nous pensons que le format des données est une information qui pourrait enrichir les capacités des adaptateurs d'une part et accélérer le processus d'apprentissage des règles, d'autre part.

Le but de ce travail est d'étendre l'algorithme de STALKER en le combinant avec le système d'induction SOFTMEALY.

STALKER définit la notion de délimiteur qui permet de représenter la position du début ou la position de fin de chaque attribut à extraire. Un délimiteur de début dans STALKER est un patron caractérisant la position du début de l'attribut à extraire en se basant sur ce qui le précède. SOFTMEALY définit la notion de séparateur pour caractériser la position de début et de fin de chaque attribut à extraire. Cependant, un séparateur du début, comme, de fin, est définit par deux patrons, l'un pour caractériser la position du début de l'attribut à extraire en se basant sur ce qui le précède, similaire au délimiteur de STALKER ; et l'autre caractérisant l'attribut à extraire lui-même en se basant sur le type de la valeur à extraire.

Les méthodes à base de délimiteurs ne prennent en compte que la chaine de caractères précédant (ou suivant) les valeurs à extraire alors que le format des données à extraire est une information exploitable qui enrichit l'expressivité des motifs. Les séparateurs ont l'avantage de pouvoir déterminer le début d'un attribut à extraire à partir d'un texte pauvre en caractères.

L'objectif principal de notre travail est d'étendre STALKER en y introduisant la notion de séparateur.

Nous allons présenter le processus de construction d'adaptateur basé sur l'algorithme de STALKER étendu par l'introduction de la notion du séparateur du SOFTMEALY. Nous allons alors décrire en détail les différents algorithmes introduits dans cette approche, ensuite, nous allons donner un exemple d'apprentissage des règles d'extraction pour illustrer l'avantage de l'utilisation de la notion du séparateur par rapport à celle du délimiteur. Enfin, nous allons finir par une synthèse de comparaison entre les deux notions.

L'objectif de l'extraction est de transformer la page Web sous la forme d'informations organisées, bien structurées et facile à intégrer dans d'autres applications. Dans le système STALKER, une page Web est représentée par un arbre imbriqué dit en anglais *Embedded catalog tree* (ECT). Le fait de représenter la page Web sous une structure d'arbre ECT rend le problème de la construction d'adaptateur facile à gérer et ne nécessite pas beaucoup de difficulté au moment de l'apprentissage.

2.2 Code html de la page Web

Les pages sur le Web sont décrites d'une façon à être plus lisibles par l'opérateur humain. À cet effet, plusieurs conventions communes ont été appliquées sur la structure du code html. L'information sur le Web est dite semi-structurée et elle est souvent présentée sous la forme de données hiérarchiques. [I. Muslea et al., 1998] se sont basés sur le code source html de la page Web. Ils ont considéré que chaque document est vu sous forme d'un ensemble de jetons. Afin de faciliter le traitement du contenu du code html, ils ont introduit la notion de l'analyse lexicale sur laquelle ils ont introduit un certain nombre de jetons, dits alphabet de base. Trois types de jetons se distinguent :

- Un mot : Correspond à toute chaîne de caractères alphanumériques.
- Un symbole de ponctuation : Correspond à tout symbole de ponctuation comme ',' ';', '?' etc.
- Une balise html : Correspond à toute balise html, par exemples (<html>, <tr>, </td>, etc...).

Noté par Σ_d l'ensemble de l'alphabet, et supposant P un fragment du texte représentant un document à traiter, donc Σ_d s'écrit sous la formule suivante :

Σ_d = words(P) ∪ symbols(P) ∪ balises(P) avec words(P) l'ensemble des mots contenus dans P, symbols(P) l'ensemble des symboles de ponctuation contenus dans P, et balises(P) l'ensemble des balises html contenues dans P.

2.3 L'arbre hiérarchique ECT

En s'inspirant de l'analyse lexicale du code html de la page Web, [I. Muslea et al., 1998] ont proposé l'arbre ECT *(embeded catalog tree)* pour décrire la décomposition logique du document source et simplifier le problème de l'extraction le cas de données

présentant une certaine hiérarchie. Chaque nœud de l'ECT représente un ensemble de jetons défini dans l'ensemble de l'alphabet cité précédemment.

L'ECT est composé de trois types de nœuds :

1- Une feuille : C'est un élément primitif qui représente une information à extraire dans l'arbre.

2- Un nœud liste : C'est un élément dans l'arbre qui peut avoir aucun ou un seul fils représentant le type des éléments d'une liste.

3- Un nœud tuple : C'est un élément dans l'arbre qui peut avoir un ou plusieurs fils. Il est à noter que les fils d'un tuple n'ont pas le même type d'information.

La figure 1.2 du chapitre état de l'art représente l'arbre hiérarchique ECT d'un document restaurant de la figure 1.1 appartenant à la source ZAGAT[6]. En remarquant bien que les nœuds Nom, Food, Décore, Service, Prix, Cuisine, Adresse, Téléphone, et Description sont des feuilles, Les nœuds Restaurants, et Locations sont des nœuds liste, et les nœuds Restaurant, et Location sont des nœuds tuple.

2.4 Définition des termes

Un délimiteur

Dans le contexte de notre travail, un délimiteur est un patron utilisé pour localiser une position dans un fragment de texte (document html).

Un séparateur

Un séparateur est un patron utilisé pour localiser une position dans un fragment de texte (document html) en se basant sur ce qui la précède et ce qui lui succède. Lorsque cette

[6] http://www.isi.edu/info-agents/RISE

position correspond au début de la valeur d'un attribut à extraire, le séparateur l'identifie par ce qui le précède et par une caractérisation de l'attribut lui-même.

Faux positifs et vrais positifs

Supposons que nous sommes en train de calculer la règle gauche de l'un des nœuds de l'arbre ECT. Durant l'apprentissage de la règle, l'algorithme renvoie à chaque étape un ensemble de règles candidates. L'application de ces règles sur les exemples d'entrainement permet l'extraction de certaines valeurs. Les valeurs extraites qui sont correctes sont dites les vrais positifs et les incorrectes sont les faux positifs.

Un exemple d'entrainement

Une chaîne de caractères qui précède ou succède l'information à extraire dans un document html est appelée exemple d'entrainement. On parle d'un exemple d'entraînement gauche dans l'apprentissage des règles gauches, et d'un exemple d'entraînement droit dans l'apprentissage des règles droites. L'exemple d'entrainement

Un joker

En analysant syntaxiquement la source du document à extraire, l'algorithme permet de générer un alphabet dont chaque symbole représente un type bien particulier. La généralisation de cet alphabet génère des symboles caractérisant toute chaîne de caractère de même type que les symboles de l'alphabet.

Un contexte d'information

Nous avons introduit cette terminologie dans le but d'analyser et d'étudier le format des valeurs à extraire. Un contexte d'information est un joker permettant de donner un sens et un format régulier aux valeurs des attributs. Prenons par exemple le cas d'un attribut numéro de

téléphone, le contexte d'information dans ce cas peut être représenté par le patron ###-###-####. Le contexte d'information ajoute une sémantique de typage à la valeur en question. L'exemple suivant illustre un cas du contexte d'information représentant un prix du logement. Le texte en gris de l'exemple ci-dessous représente le contexte d'information de la valeur 'prix' à extraire.

<hr> Belltown

 CONCEPT ONE

 1 BDRM, $775

Une règle d'extraction

Une règle d'extraction est une disjonction de séparateurs localisant un nœud. Dans ce travail, nous distinguons deux règles d'extraction, une pour déterminer la position gauche et l'autre pour déterminer la position droite. Ces règles permettent de déterminer exactement la position de l'occurrence de l'attribut à extraire dans un fragment du texte.

2.5 Apprentissage

La détermination d'une information à extraire est basée sur la notion du séparateur. Chaque nœud à extraire doit avoir 2 séparateurs caractérisant sa position gauche, et celle de droite dans son nœud parent. La phase permettant de calculer ces séparateurs s'appelle le processus d'apprentissage. Dans ce qui suit on va décrire en détail le fonctionnement de l'algorithme d'extraction utilisé dans ce travail.

La fonction Apprentissage ( ) de la figure 2.1 est invoquée avec comme paramètres les exemples d'entraînement E. Au début, la disjonction de règles R est initialisée à vide. La procédure Terminaux ( ) de la figure 2.2 renvoie l'ensemble des terminaux utilisant comme paramètre l'ensemble des exemples E. Cet ensemble représente les symboles apparaissant dans chacun des exemples de E. Ces symboles sont extraits à partir des chaines de caractères qui précèdent, succèdent les occurrences recherchées dans les exemples E. Ils sont définis dans l'ensemble Σ_p.

```
Apprentissage(Exemples) {
        ReglesGauches = Ø;
        Candidats = ContextInformation(Exemples);
        Positions = position(Exemples);
        Exemple = Exemples;
        While ( Exemple != Ø ) {
            T = Terminaux(E);
              T' = GeneraliserTerminaux(T);
              Delimiteur = ApprendreDelimiteur(E, T');
              ReglesGauches = ReglesGauches U Delimiteur;
              Positions = position(Exemple);
              Exemple = NonCouvert(Exemple, Delimiteur, Positions);
        }
return ReglesGauches;
}
```

Figure 2.1 Algorithme d'apprentissage

```
Terminaux(E) {
        Terminaux = Ø;
        Pour chaque exemple de E {
            Balisehtml = Calculer l'ensemble de balises html; #balises html de
                         chaque exemple
            T = T + Balisehtml; // ajouter les balises à l ensemble des terminaux
            Mots = Calculer l'ensemble de mots; //l'ensemble des mots
            T = T + Mots; ajouter les mots à l ensemble des terminaux
            Symboles = Calculer l'ensemble de symboles; l'ensemble des symboles
            T = T + Symboles; ajouter les symboles à l'ensemble des terminaux
        }
       return T;  // l ensemble des terminaux des exemples
}
```

Figure 2.2 Algorithme de calcul des terminaux

Ensuite, Le programme fait appel à la procédure GeneraliserTerminaux ( ) avec l'ensemble T en paramètre pour généraliser l'ensemble des terminaux; le résultat renvoyé est stocké dans l'ensemble T'. L'objectif de cette fonction est d'ajouter à l'ensemble T l'ensemble des jokers définis correspondant à chaque symbole de T.

Les jokers introduits dans notre travail sont :

- [',', '.', ';', '!', '?'] : Un symbole de ponctuation quelconque.
- <.+> : Un symbole quelconque de l'ensemble des balises html.
- .+ : Un symbole quelconque de l'ensemble des mots.
- \d : l'ensemble des nombres.
- [A-Z]{1,} : L'ensemble des mots en majuscule.
- [A-Z]3[A-Z-a-z]{n,} : L'ensemble des mots dont les 3 premières lettres sont en majuscule.

- \d3-\d3 : L'ensemble des chaînes de caractères respectant la casse 3 chiffres ensuite un tiret, et ensuite 3 chiffres.
- Un nombre : \d{1,}.
- Un nombre composé de 4 chiffres et plus: \d{4, }.
- Un nombre comprenant 4 chiffres seulement : \d4.
- Balise image : C'est une balise html qui permet de signaler la présence d'une image.
- Balise lien : C'est une balise html représentant un lien vers une autre page.
- Balise adresse : C'est une balise html utilisée pour l'insertion des adresses mail, l'adresse mail se trouve juste après cette balise.
- Mot comprenant au moins n lettres majuscules : [A-Z]{n, }. Avec n un entier positif.

Remarque

Nous confirmons que la description de délimiteurs ci-haut est présentée selon le langage de programmation utilisé dans notre approche et qui est Perl.

Après que l'ensemble T' soit calculé, la fonction ApprendreDelimiteur ( ) de la figure 2.3 est invoquée pour renvoyer le délimiteur. Cette méthode est appelée avec comme paramètres l'ensemble des exemples E et l'ensemble des symboles T'. Elle fait appel à la fonction GenererCandidats ( ) de la figure 2.4, sur laquelle, elle renvoie l'ensemble des candidats avec l'ensemble des valeurs de types des éléments à extraire. La fonction ContextInformation ( ) de la figure 2.5 permet de calculer l'ensemble des types d'informations de chaque valeur à extraire. Ces candidats représentent les symboles qui précèdent immédiatement les occurrences à chercher dans les exemples ainsi que l'ensemble des types de valeurs à extraire. Chaque candidat représente une règle simple sous la forme SkipTo(X) ou SkipUntil(X), où X est un symbole défini dans Σ_d représentant un élément de l'ensemble de délimiteurs initiaux.

```
ApprendreDelimiteur(E, T') {
        Candidats = GenererCandidat(E);
        Positions = Position(Exemples);
        Delimiteurs = Ø;
        C = Ø;
        Tant que ( Candidat != Ø ) {
          VP = Vrais_Positifs ( Candidats, Exemples, Positions);
          FP = Faux_Positifs (Candidats, Exemples, Positions);
          MeilleuresCandidats = MeilleurCandidat(Candidats, VP, FP, Terminaux);
          Candidats = Ø;
          Pour chaque MeilleuresCandidats {
              Si ( Si_Est_Parfait (MeilleursCandidats, E, Positions) ) {
                  Delimiteurs = Delimiteurs U MeilleursCandidats;
              }
              Sinon {
                  C = Raffinement (MeilleursCandidats, T');
                  Candidats = Candidats U C;
              }
```

Figure 2.3 Algorithme d'apprentissage de délimiteur

```
GenererCandidat(E) {
    Pour chaque exemple de E {
        Candidats = L'ensemble des symboles qui précèdent l'information à extraire dans
        chaque élément de l'ensemble E;
        ContexteInformations = ContexteInformation(E);
        Candidats = Candidats U ContexteInformations;
    }
 return Candidats;
}
```

Figure 2.4 Algorithme de la génération des candidats

```
ContextInformation (E) {
    Contexts = ∅;
    Pour chaque exemple e représentant l'information à extraire de l'ensemble E {
        Context = Type d'information (e);
        Contexts = Contexts ∪ Context;
    }
  Return Contexts;
  }
```

Figure 2.5 Algorithme de calcul des contextes d'informations

La fonction MeilleurCandidat( ) de la figure 2.6 permet de renvoyer le meilleur candidat parmi l'ensemble des candidats C. En effet, ce candidat doit vérifier l'ensemble des heuristiques suivantes:

1- Couvre le plus grand nombre d'exemples vrais positifs; c'est-à-dire le maximum d'informations à extraire.
2- Retourne moins d'exemples positifs faux.
3- Contient moins de jokers.
4- Possède un délimiteur de longueur maximal.

La fonction MeilleurCandidat( ) est appelée avec comme paramètres l'ensemble des candidats retournés par GenererCandidat( ) et l'ensemble des exemples initiaux E (tous les exemples). Dès que le meilleur candidat soit trouvé par la fonction MeilleurCandidat ( ), la fonction ApprendreDelimiteur ( ) fait appel à la fonction Si_Est_Parfait ( ) de la figure 2.7 pour tester si ce candidat est parfait ou non. Un candidat parfait est un meilleur candidat qui n'accepte pas d'exemples négatifs.

Une fois la valeur retournée par la fonction Si_Est_Parfait ( ) est vraie, le candidat donc est parfait et il est retourné dans D comme délimiteurs de début de l'ensemble d'exemples qui les couvre. Puis, ces exemples seront automatiquement retranchés de l'ensemble E. Le même processus sera réitéré pour l'ensemble des exemples restants.

```
MeilleurCandidat(Candidats, FP, VP, T') {
        Pour chaque Candidats {
            Max = Maximum(VP);
            MeilleursCandidats = Candidats qui vérifie le Max;
        }
        Pour chaque MeilleursCandidats {
            Min = Minimum (FP);
            MeilleursCandidats = MeilleursCandidats qui vérifie le Min;
            Max_Longueur= AXIMUM(LONGUEUR(MeilleursCandidats));
            MeilleursCandidats = le meilleur candidat qui contient un   Max_Longueur;
        }
    return MeilleursCandidats;
}
```

Figure 2.6 Algorithme de calcul des meilleurs candidats

```
Si_Est_Parfait (Candidat, E, Positions) {
        FP = Nombre_Exemples_ faux_positifs(Candidat, E, Positions);
        Si ( FP == 0 ) {
            #Candidat est parfait;
        return 1;
        }
        Sinon {
            #Candidat n'est pas parfait;
        return 0;
        }
}
```

Figure 2.7 Algorithme de recherche d'un candidat parfait

Dans le cas où la valeur retournée par la fonction Si_Est_Parfait ( ) est fausse (le candidat n'est pas parfait), la fonction Raffinement ( ) de la figure 2.11 est appelée avec comme paramètres le délimiteur D, et l'ensemble des terminaux généralisés T'. Cette fonction permet de raffiner le délimiteur D suivant 2 types de raffinements :

1- Le raffinement de délimiteur : Permet d'élargir le symbole X de délimiteur D = SkipTo(X), en ajoutant un symbole de l'ensemble T` à X. Ce processus est réitéré tant qu'il existe des symboles dans l'ensemble T`. Un nouveau candidat est sous la forme SkipTo(tX) tel que t appartient à l'ensemble T`. Suivant la représentation de l'automate à état fini caractérisant la règle d'extraction D, un raffinement de délimiteur permet de modifier la chaîne à consommer qui permet le passage de l'état initial vers l'état final en la concaténant avec le symbole t en question. La figure 2.9 illustre un automate à état fini avec raffinement de délimiteur par le symbole t.

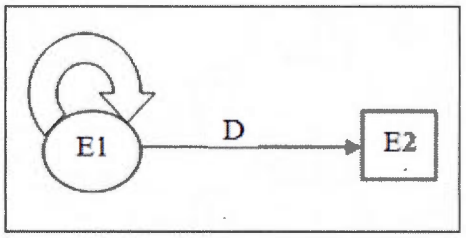


Figure 2.8 Automate à état fini de la règle D

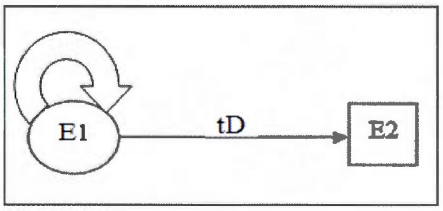


Figure 2.9 Automate à état fini après raffinement de délimiteur avec t

1- Le raffinement de topologie : Permet d'introduire un état intermédiaire au niveau du délimiteur D. Supposons D = SkipTo(X), le raffinement de topologie permet d'ajouter un nouveau candidat sous forme C = SkipTo(t)SkipTo(X) tel que t représente un symbole de l'ensemble T`. Ce processus est réitéré pour chaque symbole t appartenant à l'ensemble T`. Dans le formalisme de l'automate à état fini représentant la règle d'extraction D, un raffinement de topologie utilisant le symbole t permet d'introduire un nouvel état intermédiaire entre l'état initial et l'état final. Le passage de l'état initial vers le nouvel état intermédiaire se fait par la consommation de la chaîne de caractère t. Le schéma de la figure 2.10 ci-dessous illustre l'automate à état fini du résultat de raffinement de topologie du délimiteur D et le symbole t.

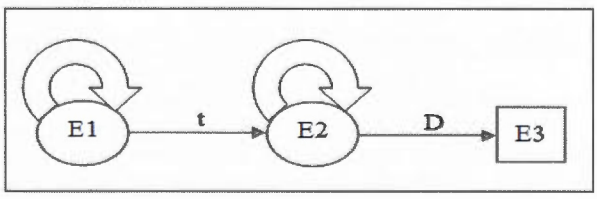


Figure 2.10 Automate à état fini représentant la règle après le raffinement

Étant donné l'ensemble des terminaux T = { '.', '
', '[', '.', ';', '!', '?']', '*h' }.

Supposons nous voulons calculer l'ensemble des candidats après le processus de raffinement du candidat SkipTo(.).

Nous aurons deux types de raffinement, le raffinement de délimiteurs et le raffinement de topologie. Le tableau 2.1 ci- dessous illustre l'ensemble des candidats conclus après raffinement.

Raffinement de délimiteur	Raffinement de topologie
SkipTo(..) SkipTo(.) SkipTo(*s.) SkipTo(*h.)	SkipTo(.)SkipTo(.) SkipTo( )SkipTo(.) SkipTo(*s)SkipTo(.) SkipTo(*h) SkipTo(.)

Tableau 2.1 Tableau de raffinement de délimiteur et de raffinement de topologie

```
Raffinement(MeilleursCandidat, T) {
        RaffinementDelimiteur = Ø;
        Raffinement Topologie = Ø;
        Raffinements = Ø;
        Pour chaque terminal t de l'ensemble T{
           RaffinementDelimiteur =  SkipTo(t. MeilleursCandidat);
           Raffinements = Raffinements U  RaffinementDelimiteur;
           DelimiteurTopologie  =  SkipTo(t).SkipTo(MeilleursCandidat);
           Raffinements = Raffinements U RaffinementTopologie;
        }
  return Raffinements;
}
```

Figure 2.11 Algorithme de raffinement des candidats

À la fin du processus de raffinements, la fonction Raffinement ( ) retourne l'union des candidats renvoyés par chaque type de raffinements, et la fonction ApprendreDelimiteur ( ) renvoie ces candidats dans C. Cette fonction est réitérée à partir de la fonction MeilleurCandidat ( ). Tant que le candidat parfait n'est pas encore trouvé, le processus de raffinements est réitéré.

Dès que le meilleur candidat est trouvé, la fonction Raffinement ( ) renvoie comme résultat le délimiteur D, et la procédure Apprentissage ( ) ajoute ce délimiteur à l'ensemble de disjonction R. Ensuite, les exemples couverts par D seront retranchés de l'ensemble des exemples E. La procédure Apprentissage ( ) est répétée avec les nouveaux exemples E et leurs marqueurs de débuts. Elle s'arrête dès qu'il ne reste plus d'exemple dans E, et dans ce cas la disjonction R est retournée comme résultat représentant la règle gauche.

2.6 Extraction

Après avoir calculé les règles d'extraction de chacun des nœuds de l'arbre ECT (avec exception la racine de l'arbre ne nécessite pas le calcul des règles), l'utilisateur peut procéder à la phase d'extraction, il suffit de faire entrer un lien Web ou un chemin dans le disque dur représentant la page à extraire. L'algorithme de l'extraction est appliqué sur le contenu Html du document à extraire.

Dans la phase de l'extraction, l'étiquetage du document à extraire n'est pas indispensable, et notre algorithme est capable de déduire les nœuds parents et leurs informations d'une manière automatique, en appliquant successivement l'algorithme d'extraction en commençant de la racine.

L'algorithme de l'extraction doit être appliqué sur chaque nœud de l'arbre ECT dont ce nœud n'est pas une feuille. Ce dernier utilise comme paramètre le fragment du texte représentant le nœud (le contenu du nœud). Dans le processus d'extraction, nous distinguons 2 types d'extractions possibles.

2.6.1 Extraction des éléments d'un nœud tuple

Supposons un nœud N de type tuple contenant k occurrences $n_i,..,n_k$, l'extraction des éléments (champs de l'occurrence k) n_{ik} à partir du nœud tuple exige l'étiquetage des débuts et des fins de chaque élément dans son nœud parent N. En plus, le nœud N appartenant au nœud parent M rend indispensable également le marquage du début et de fin de chaque occurrence m_i du nœud parent M. Le processus d'extraction permet de retourner à la fois l'ensemble des éléments de N.

Pour définir le début, ainsi que la fin des éléments de chaque occurrence n_i dans son parent m_i, on doit définir l'ensemble des exemples à fin de pouvoir apprendre les règles d'extraction.

Les exemples d'entrainement pour les débuts, respectivement les fins représentent les chaînes de caractères qui précédent respectivement succèdent directement les occurrences du nœud N.

Pour la règle gauche (début) : Le premier exemple représente la chaîne de caractères située entre le début de l'occurrence m_i et le début de l'occurrence n_{i1}, l'exemple j (j = 2, k) représente la chaîne de caractères se trouvant entre la fin de l'occurrence $n_{i\ j-1}$ et le début de l'occurrence $n_{i\,j}$.

Pour la règle droite (fin) : L'exemple j (j= 1, k-1) représente la chaîne de caractères se trouvant entre la fin de l'occurrence $n_{i\,j}$ et le début de l'occurrence $n_{i\ j+1}$ .Le dernier exemple représente la chaîne de caractères se trouvant entre la fin de l'occurrence n_i k et la fin de l'occurrence m_i .

Les parties grisées de la figure 2.12 représentent les exemples d'entrainement de la règle d'extraction gauche du nœud n_i. Ces exemples représentent les parties gauches qui définissent le contexte gauche du séparateur dans le cas du système SOFTMEALY. Les contenus des valeurs à extraire représentent le contexte droit du séparateur.

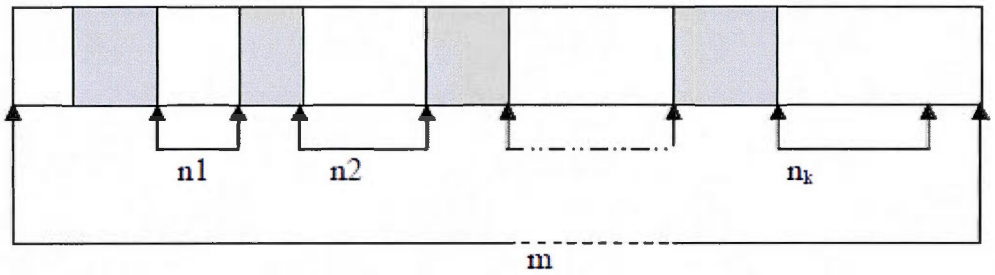


Figure 2.12 Occurrences n du nœud N dans l'occurrence m du nœud parent M

La figure 2.13 représente une page web extraite de la source Web des restaurants LAWEEKLY[7]. Cette page donne des détails sur un ensemble de restaurants. Elle est représentée sous forme d'un ensemble de tuples, chaque tuple représente cinq informations d'un restaurant. Ces informations sont le nom du restaurant, son adresse, son téléphone, sa description, le type de la carte de crédits utilisés pour le paiement des repas.

LA Restaurants

[7] http://www.isi.edu/integration/RISE/repository.html

Search Criteria: Location: Any Cuisine: American

ARBUCKLE'S AMERICAN CUISINE
829 N. La Cienega Blvd., L.A.
(310) 657-9220
Built in the early 1900's, Arbuckle's, once the home of Fatty Arbuckle, is the last original row house on La Cienega. Although Fatty doesn't live here anymore, there are still plenty of reminders of days gone by in photos of silent film stars. Relax on a patio covered in geraniums, ivy and fresh jasmine and enjoy delish morsels like a martini of chilled shrimp ($7.95), luscious crab cakes ($15.50) or tender lamb chops ($18.95). Dinner seven nights. Full bar; takeout; catering; valet parking; AE, MC, V.

BOOK SOUP BISTRO
8800 Sunset Blvd., W. Hollywood
(310) 657-1072
Featuring a handsome, comfy atmosphere indoors and al fresco patio seating, the Book Soup Bistro provides a common ground for hardcore power lunchers as well as the funky cappuccino crowd. Known for the homey, healthy-helpings menu of items such as turkey meat loaf with mashed potatoes and cranberry sauce ($11.50) and a seared ahi tuna sandwich ($11.50), Book Soup continuing Saturday and Sunday brunch and the "Short Stories" menu of new items and abbreviated regular dishes, perfect for light afternoon snacks. Lunch and dinner seven days. Full bar; takeout; catering; parking; reservations accepted. AE, DC, DIS, MC, V.

Outdoor Grill
12630 1/2 Washington Place, Culver City
(310) 636-4745
From as early as 9 a.m. there's no question about what's going on here. In a condolike two-story building adjacent to Handy J's Hand Wash, pieces of chicken, racks of baby back pork ribs and thick triangles of Nebraska beef sirloin are already sizzling on the 10-foot-long outdoor grill. The meat has absorbed enough flavors from the marinade and the grill smoke to stand alone. Though of course there's also the smoky, sweet (but not too) barbecue sauce, which doesn't overshadow the meat if used sparingly. Excellent spicy turkey chili, tasty soups, salads topped with good homemade dressings, and sides like macaroni and cheese here can turn simple barbecue into a real meal for about $6 to $15. Eat inside looking out at the grill, or on the top deck with its unobstructed view of the car wash. Doesn't matter--the grub's the thing.

DAILY GRILL
Beverly Center, 100 N. La Cienega Blvd., L.A.
(310) 659-3100
Hearty portions and excellent service give the Daily Grill ä an ever-increasing following of lovers of freshly made American favorites. Blue Plate specials change daily ($10.95-$16.95). A popular Blue Plate, the wholesome meat loaf is topped with homemade gravy and comes with mashed potatoes and Daily's trademark steamed broccoli ($10.95). A selection of done-just-right steaks ($13.95-$17.95) will satisfy the discerning carnivore, while the broiled half chicken with garlic ($9.95), with shoestring potatoes and broccoli, satiates lighter appetites. Lunch and dinner seven days. Full bar; takeout; delivery; catering, parking; reservations suggested. AE, DC, MC, V.

Figure 2.13 Projection d'une source Web Laweekly[8] restaurant représentant 4 tuples

Le tableau 2.2 Ci-dessous montre les résultats retournés après l'extraction de la source de la figure 2.13.

[8] http://www.isi.edu/info-agents/RISE/LAW/__Source__.html

Nom	Adresse	Numéro du téléphone	Description	Carte du crédit
ARBUCKLE'S AMERICAN CUISINE	829 N. La Cienega Blvd., L.A.	(310) 657-9220	Builtin...valet parking;	AE, MC, V.
BOOK SOUP BISTRO	8800 Sunset Blvd., W. Hollywood	(310) 657-1072	Featuring a ... Reservations accepted.	AE, DC, DIS, MC, V.
Outdoor Grill	12630 1/2 Washington Place, Culver City	(310) 636-4745	From as... grub's the thing.	
DAILY GRILL	Beverly Center, 100 N.La Cienega Blvd., L.A.	310) 659-3100	Hearty portions... Reservations suggested.	AE, DC, MC, V.

Tableau 2.2 Résultat d'extraction de la page de la figure 2.13

2.6.2 Extraction des éléments d'un nœud liste

Supposons un nœud N de type liste, bien que le nœud N peut avoir 0 ou un seul fils n, l'extraction de chacun des éléments du fils n dans le nœud parent N permet d'extraire les occurrences n_i à partir du nœud fils n. Ce processus est le même que celui de l'extraction des éléments d'un nœud tuple sauf qu'un tuple peut avoir plusieurs fils de type différents.

2.6.3 Exemple d'un scénario d'apprentissage

Dans cette section, on va appliquer notre algorithme d'apprentissage sur un ensemble d'exemples d'entrainement et on va expliquer en détail le scénario d'apprentissage de calcul de la règle gauche (règle de début). Nous allons étudier les deux cas possibles que nous avions introduits dans notre algorithme.

Soit N un ensemble d'occurrences extraites à partir de la source Web Rentals[9]. Cet ensemble représente un tuple 'LOCATION' décrivant des informations pour des appartements. Ces informations sont les suivantes :

[9] http://www.isi.edu/info-agents/RISE/Rentals/__Source__.html

- Adresse.
- Nombre de chambres.
- Prix
- Numéro du téléphone.

Le scénario d'apprentissage consiste à calculer la règle d'extraction gauche (règle de début) de l'élément 'PRIX' appartenant au tuple 'LOCATION' précédent. $N = \{n_1, n_2, n_3, n_4\}$.

n_1 = <hr> Belltown

 CONCEPT ONE

 1 BDRM, $775

 Lake Union & Sound Views
 Fplc, W/D, Gar Prkg Available

 206-728-9515

n_2 = <hr> BALLARD AT LOCKS
 Charming, security bldg, on bus-
 line, pool. Studio/1 BR $535.
 206 784-5797. 3025 NW Market St.

n3 = <hr>
 BALLARD - 1 Bedroom
 new cpts, drapes & paint, nr hospital & bus. $535.
 1519 NW 65 th, Apt #4
 206-542-4600

n4 = <hr> Ballard-Open Su 11-3 -
 Walk to town, spac 2 br bsmt unit in 3 plex. New carpet, paint, frpl, w/d, $795 inclds all utils!
 2244 NW 62 nd. 425-881-8922

1- Cas du délimiteur :

- Le texte en jaune représente l'information à extraire dans chacune des occurrences.
- Soit E l'ensemble des exemples d'entrainement extrait à partir de l'ensemble des occurrences de l'ensemble OC. Cet ensemble représente les chaines de caractères qui précèdent les valeurs de l'attribut 'PRIX' dans chacune des occurrences.

$E = \{E_1, E_2, E_3, E_4\}$.

E_1 = <hr> Belltown

 CONCEPT ONE

 1 BDRM,

E_2 = <hr> BALLARD AT LOCKS
 Charming, security bldg, on bus-
 line, pool. Studio/1 BR

E_3 = <hr>
 BALLARD - 1 Bedroom
 new cpts, drapes & paint, nr hospital & bus.

E4 = <hr> Ballard-Open Su 11-3 -
 Walk to town, spac 2 br bsmt unit in 3 plex. New carpet, paint, frpl, w/d,

Dans la suite et pour la simplification, on va noter par :

- T l'ensemble des terminaux.
- T' l'ensemble des terminaux généralisés.
- C l'ensemble des candidats.
- C' l'ensemble des candidats généralisés.
- D l'ensemble des délimiteurs.
- NEFP le nombre d'exemples faux positifs.
- NEVP le nombre d'exemples vrais positifs.

Durant la première itération :

C = {BR , .}, C' = {BR , . [',', '.', ';', '!', '?'] .+}.

Analyse des candidats : Les détails de l'analyse des candidats sont illustrés dans le tableau 2.3 ci-dessous.

Candidat	NEPF	NEPV
BR	0	1
,	3	1
.	2	1
[',', '.', ';', '!', '?']	3	1
.+	4	0

Tableau 2.3 Détails de l'analyse des candidats

Suivant les heuristiques dont nous avions parlé plus haut, le candidat 'BR' est le meilleur candidat, ce candidat est parfait car il satisfait l'exemple E2 et il ne satisfait aucun exemple faux positif. L'algorithme retourne comme résultat la règle R1 = SkipTo('BR').

E = E – {E_2} = {E_1, E_3, E_4}.

Deuxième itération :

C = {, . }, C' = {, . [',', '.', ';', '!', '?'] }.

Analyse des candidats : Le tableau ci-dessous récapitule les résultats de l'analyse des candidats durant l'itération.

Candidat	NEPF	NEPV
,	2	1
.	1	1
[',', '.', ';', '!', '?']	2	1

Tableau 2.4 Résultat de l'analyse des candidats durant la deuxième itération

Le meilleur candidat est '.', ce candidat n'est pas parfait car il satisfait un exemple positif faux.

Première itération de raffinement : Le tableau 2.5 ci-dessous présente l'ensemble des candidats trouvés après le raffinement de la première itération.

T = {<hr>
}, T' = {<hr>
 <.+ ?>}.

Raffinement de délimiteur	Raffinement de topologie
C_1 = SkipTo('<hr>.').	C_4 = SkipTo('<hr>') SkipTo('.').
C_2 = SkipTo(' .').	C_5 = SkipTo(' ') SkipTo('.').
C_3 = SkipTo('<.+?>.').	C_6 = SkipTo('<.+?>') SkipTo('.').

Tableau 2.5 Candidats de raffinement durant la première itération

Analyse des candidats : Le tableau ci-dessous montre le détail de l'analyse des candidats durant l'itération.

Candidat	NEPF	NEPV
SkipTo('<hr>.')	0	0
SkipTo(' .')	0	0
SkipTo('<.+?>.')	0	0
SkipTo('<hr>') SkipTo('.')	1	1
SkipTo(' ') SkipTo('.')	1	1
SkipTo('<.+?>') SkipTo('.')	1	1

Tableau 2.6 Détail de l'analyse des candidats

Les meilleurs candidats sont C_4 et C_5 car les deux satisfont E_3, ces candidats ne sont pas parfais car ils satisfont un exemple positif faux.

Deuxième itération de raffinement : Le tableau 5 ci-après présente l'ensemble des candidats après raffinement.

T = {<hr>
}, T' = {<hr>
 <.+ ?>}.

Raffinement de délimiteur	Raffinement de topologie
C_1 = SkipTo('<hr><hr>')SkipTo('.').	C_7 = SkipTo('<hr>')SkipTo('<hr>') SkipTo('.').
C_2 = SkipTo(' <hr>')SkipTo('.').	C_8 = SkipTo(' ')SkipTo('<hr>') SkipTo('.').
C_3 = SkipTo('<hr> ')SkipTo('.').	C_9 = SkipTo('<hr>')SkipTo(' ') SkipTo('.').
C_4 = SkipTo(' ')SkipTo('.').	C_{10} = SkipTo(' ')SkipTo(' ') SkipTo('.').
C_5 = SkipTo(<.+?><hr>')SkipTo('.').	C_{11} = SkipTo('<.+?>')SkipTo('<hr>') SkipTo('.').
C_6 = SkipTo('<.+?> ')SkipTo('.').	C_{12} = SkipTo('<.+?>')SkipTo(' ') SkipTo('.').

Tableau 2.7 Liste des candidats après raffinement

Analyse des candidats : Le tableau 2.8 ci- après illustre le détail de l'analyse des candidats.

Candidat	NEPF	NEPV
SkipTo('<hr><hr>')SkipTo('.')	0	0
SkipTo(' <hr>')SkipTo('.')	0	0
SkipTo('<hr> ')SkipTo('.')	0	0
SkipTo(' ')SkipTo('.')	0	0
SkipTo(<.+?><hr>')SkipTo('.')	0	0
SkipTo('<.+?> ')SkipTo('.')	0	0
SkipTo('<hr>')SkipTo('<hr>') SkipTo('.')	0	0
SkipTo(' ')SkipTo('<hr>') SkipTo('.')	0	0
SkipTo('<hr>')SkipTo(' ') SkipTo('.')	1	1
SkipTo(' ')SkipTo(' ') SkipTo('.')	0	1
SkipTo('<.+?>')SkipTo('<hr>') SkipTo('.')	0	0
SkipTo('<.+?>')SkipTo(' ') SkipTo('.')	1	1

Tableau 2.8 Détail de l'analyse des candidats

Le meilleur candidat est C_{10} car il satisfait E_3, ce candidat est parfait car il ne satisfait aucun exemple positif faux. L'algorithme retourne R2 = SkipTo('
')SkipTo('
') SkipTo('.') comme délimiteur gauche.

$E = E - E_3 = \{E_1, E_4\}$.

Durant la troisième itération :

C = { , }, C' = { , .+ }.

Analyse des candidats : Le tableau 7 suivant montre le détail de l'analyse des candidats.

Candidat	NEPF	NEPV
,	1	1
[',', '.', ';', '!', '?']	1	1

Tableau 2.9 Détail de l'analyse des candidats

Le meilleur candidat est ',' car il satisfait E_1, ce candidat n'est pas parfait car il satisfait un exemple de faux positif.

Première itération de raffinement : Le tableau 2.10 ci-après présente l'ensemble des candidats retournés après la première itération de raffinement.

T = {<hr>
}, T' = {<hr>
 < .+ ?>}.

Raffinement de délimiteur	Raffinement de topologie
C_1 = SkipTo('<hr>,').	C_4 = SkipTo('<hr>')SkipTo(',').
C_2 = SkipTo(' ,').	C_5 = SkipTo(' ')SkipTo(',').
C_3 = SkipTo('<.+ ?> ,').	C_6 = SkipTo('<.+ ?> ')SkipTo(',').

Tableau 2.10 Candidats après raffinement

Analyse des candidats : Le tableau 2.11 ci-après décrit le détail de l'analyse des candidats.

Candidat	NEPF	NEPV
SkipTo('<hr>,')	0	0
SkipTo(' ,')	0	0
SkipTo('*h,')	0	0
SkipTo('<hr>')SkipTo(',')	1	1
SkipTo(' ')SkipTo(',')	1	1
SkipTo(('<.+?>')SkipTo(',')	1	0

Tableau 2.11 Détail de l'analyse des candidats

Les meilleurs candidats sont C_4 et C_5, ils ne sont pas parfaits car ils satisfont un exemple de faux positif.

Deuxième itération de raffinement : Le tableau 10 ci-après présente l'ensemble des candidats retournés après la deuxième itération de raffinement.

Raffinement de délimiteur	Raffinement de topologie
C_1 = SkipTo('<hr><hr>')SkipTo(',').	C_7 = SkipTo('<hr>')SkipTo('<hr>')SkipTo(',').
C_2 = SkipTo(' <hr>')SkipTo(',').	C_8 = kipTo('<hr>')SkipTo(' ')SkipTo(',').
C_3 = SkipTo(' ')SkipTo(',').	C_9 = SkipTo(' ')SkipTo(' ')SkipTo(',').
C_4 = SkipTo('<hr> ')SkipTo(',').	C_{10} = kipTo(' ')SkipTo('<hr>')SkipTo(',').
C_5 = SkipTo('<.+?> ')SkipTo(',').	C_{11} = SkipTo('<.+?>')SkipTo('<hr>')SkipTo(',').
C_6 = SkipTo('<.+?><hr>')SkipTo(',').	C_{12} = SkipTo('<.+?>')SkipTo(' ')SkipTo(',').

Tableau 2.12 Candidats après la deuxième itération de raffinement

Le meilleur candidat est C_9, ce candidat est parfait car il ne satisfait aucun exemple positif de faux. L'algorithme renvoie R2 = SkipTo('
')SkipTo('
')SkipTo(',') comme délimiteur gauche.

$E = E - \{E_1\} = E_4$.

Durant la troisième itération :

C = { , }, C' = { , .+ }.

Analyse de candidats : Le tableau 11 ci-après illustre le détail de l'analyse des candidats.

Candidat	NEPF	NEPV
,	1	0
[',', '.', ';', '!', '?']	1	0

Tableau 2.13 Détail de l'analyse des candidats

Le meilleur candidat est ',' n'est pas parfait car il satisfait un exemple de faux positif.

Première itération de raffinement : Le tableau 12 ci-après présente l'ensemble des candidats retournés après la première itération de raffinement.

Candidat	NEPF	NEPV
SkipTo('<hr>,')	0	0
SkipTo(' ,')	0	0
SkipTo(' ,')	0	0
SkipTo('<hr>')SkipTo(',')	1	1
SkipTo(' ')SkipTo(',')	1	1
SkipTo('<.+?>')SkipTo(',')	1	0

Tableau 2.14 Candidats trouvés après raffinement

Après raffinement, l'algorithme retourne SkipTo('/')SkipTo(',') comme meilleur candidat, il est aussi parfait car il ne satisfait aucun exemple faux positif. L'algorithme renvoie R3 = SkipTo('/')SkipTo(',') comme délimiteur gauche.

$E = E - E_4 = \phi$. L'algorithme d'apprentissage s'arrête et renvoie comme résultat final l'ensemble des règles gauches R = {R1, R2, R3} représentant le début de l'attribut 'PRIX'.

2- Cas du contexte d'information :

Dans cet exemple, il s'agit de calculer la règle du début de l'attribut 'PRIX'. En regardant l'ensemble des occurrences, on remarque facilement que cet attribut représente une valeur numérique précédée du symbole $.

L'algorithme Contexte(E) renvoie les candidats : $C = \{C_1, C_2, C_3\}$.

$C_1 = \$775$, $C_2 = \$535$, $C_3 = \$795$.

L'algorithme GeneraliserContexte(C) renvoie le joker $C_4 = \$NUMBER$.

Analyse des candidats : Le tableau 2.15 ci-dessous présente le détail de l'analyse des candidats.

Candidat	NEPF	NEPV
$775	0	1
$535	0	2
$795	0	1
$NUMBER	0	4

Tableau 2.15 Détail de l'analyse des candidats

L'algorithme retourne C4 comme meilleur candidat car il satisfait les quatre exemples à la fois, il est aussi parfait car il ne satisfait aucun exemple de faux positif. L'algorithme d'apprentissage renvoie R = SkipUntil($NUMBER) comme règle gauche de l'attribut 'PRIX'.

2.7 Conclusion

Dans ce chapitre, nous avons introduit le concept du contexte d'information qui représente le type de l'information de l'attribut à extraire. Nous avons démontré aussi dans l'exemple ci-haut le processus d'apprentissage avec les deux cas possibles (délimiteur STALKER et contexte d'information). Les résultats ont montré que l'utilisation du contexte d'information permet une rapidité de convergence pour l'algorithme d'apprentissage des règles d'extraction. Toutefois, la règle d'extraction appropriée au nœud de l'exemple a été

conclue durant la première itération. Le concept du type d'information donne un avantage complémentaire au délimiteur de STALKER dans la construction des règles d'extraction.

CHAPITRE III

IMPLÉMENTATION & APPLICATION

Dans ce chapitre, nous présentons les outils et technologies utilisés et le pourquoi du choix de ces outils. Ensuite, nous présentons l'outil que nous avons développé et sur lequel se base notre approche. Enfin, nous présentons l'expérimentation dans le cadre d'un projet de médiation de base de données.

3.1 Outils et technologies utilisés

Pour l'implémentation de notre approche, nous avons utilisé le langage de programmation Perl[10], nous avons ainsi utilisé la boite à outils TK[11] pour la création de notre interface graphique. Pour la gestion de notre base de données, nous avons opté pour l'environnement WampServer[12] dans sa version 2.1 qui est disponible gratuitement sur le Web et qui fonctionne sur la plate-forme Unix et Windows.

3.1.1 Perl

P.E.R.L. signifie Practical Extraction and Report Langage, est un langage de programmation créé par Larry Wall en 1987, combinant les fonctionnalités du langage C et celles des langages de scripts sed, awk et shell (sh). L'association chargée du développement et de la promotion de Perl est la fondation Perl.

Il a été conçu principalement pour accomplir des tâches d'administration système sous UNIX. Aussitôt, il fonctionne sur de nombreuses plate-formes, comme MS-DOS, OS/2, MacOS, et toute la famille des Windows. La version Perl5 du langage a rajouté des notions

[10] http://www.perl.org/
[11] http://www.tkdocs.com/
[12] http://www.wampserver.com/

de programmation objet, de structures de données complexes, des modules, et un espace de nommage, qui en font un langage de haut niveau. Ses capacités de traitement des chaînes de caractères en font un langage de choix pour la programmation des scripts CGI-bin[13].

Dans le cadre de notre projet, nous avons utilisé Perl dans sa version 5.12.4 qui fait usage de la programmation orientée objet, et qui intègre automatiquement un utilitaire de gestion de packages nommé 'Perl Package Manager'.

3.1.2 Perl Package Manager PPM

PPM est un utilitaire destiné à simplifier les tâches de localiser, installer, mettre à jour et désinstaller des paquetages. C'est un frontal aux fonctionnalités fournies par le module PPM.pm. Il peut déterminer si la version la plus récente d'un paquetage est installée sur un système, et peut éventuellement installer ou mettre à jour ce paquetage depuis un hôte local ou distant.

PPM fonctionne dans l'un des deux modes suivants : en mode shell interactif dans lequel les commandes peuvent être entrées; en mode ligne de commande dans lequel une seule action spécifique est effectuée par invocation du programme.

3.1.3 TK

Tk, initialement créée par le Dr Jonh K. Ousterhout, correspond à un besoin d'une boîte à outil simple pour son langage TCL, sur plate-forme X11. Aujourd'hui, Tk peut être utilisé par de nombreux langages: Perl, ADA, Python, scheme, etc. De plus, il est possible d'embarquer un interprète tcl simplement dans une application écrite en C, en ADA, ou autre, ce qui permet de créer simplement une interface utilisateur indépendamment de la cible. Tk fonctionne sur de nombreuses plate-formes graphiques: X11, MacOS, Win32. C'est facile à apprendre, relativement ergonomique, et il suffit de peu de lignes pour créer une application.

[13] http://www.parkansky.com/tutorials/bdlogcgi.htm

3.1.4 WampServer

WampServer est connu sous sa nouvelle version WAMP5. Il se réfère à un ensemble de logiciels libres couramment utilisés ensemble pour lancer des sites Web dynamiques ou des serveurs. Il intègre à la fois :

- Le serveur Web Apatche.
- Le système de gestion de base de données MySQL[14].
- Les langages de programmation Php, Perl, et Python.

Tous ces logiciels s'exécutent sur le système d'exploitation Windows.

3.2 Choix des outils

Dans cette section, nous expliquons pourquoi nous avons choisi ces outils. Notre projet se focalise principalement sur le traitement du contenu du code Html de la page Web. En effet, souvent le parcours du texte et la recherche dans celui-ci sont nécessaires. D'une part, Perl est un langage de programmation qui a été conçu principalement pour le traitement des expressions régulières. C'est un langage interprété, polyvalent et particulièrement adapté au traitement et à la manipulation de fichiers texte, notamment du fait de l'intégration des expressions régulières dans la syntaxe même du langage.

L'utilisation et la manipulation des chaînes de chaînes de caractères sont faciles et ne nécessitent pas un code volumineux. En plus, Perl est un langage interprété donc ne nécessite pas la compilation du code source ce qui rend le programme portable et exécutable sur n'importe quelle plate-forme. L'utilisation de l'orienté objet est très facile en Perl. Les classes sont déclarées comme des modules sous l'extension ' .pm '.

D'autre part, le package TK est très simple à utiliser et il fournit une création d'interface graphique avec un code Perl de petite taille.

L'environnement WampServer nous fournit un système de gestion de bases de données fonctionnant avec le langage des requêtes SQL.

---

[14] http://www.mysql.com/

3.3 Implémentation de l'algorithme

Dans cette section, nous allons présenter en détail le processus d'extraction dans notre application. À cet effet, nous allons présenter un exemple d'utilisation réel que nous avons traité durant notre expérimentation. Nous allons ensuite, expliquer les différentes étapes du processus d'extraction en illustrant par des écrans extraits à partir de l'application développé.

3.3.1 Présentation de l'exemple

Dans cette partie, nous allons étudier le processus d'extraction de données à partir du site Web OKRA[15]. Nous allons décrire l'arbre hiérarchique ECT représentant la structure logique des documents provenant de ce site.

3.3.1.1 Structure logique de la source Web OKRA

En consultant quelques pages Web issues du site Web OKRA, on peut facilement remarquer que chacune des pages est vue sous forme d'une liste de coordonnées personnelles, chaque élément de la liste représente un tuple de coordonnées. Chaque tuple est composé de 4 éléments feuille représentant une information à extraire (nom, score, adresse mail, et la date de la première entrée). La figure 3.1 illustre l'arbre hiérarchique ECT de ce site.

[15] http://www.isi.edu/info-agents/RISE/wOKRA/__Source__.html

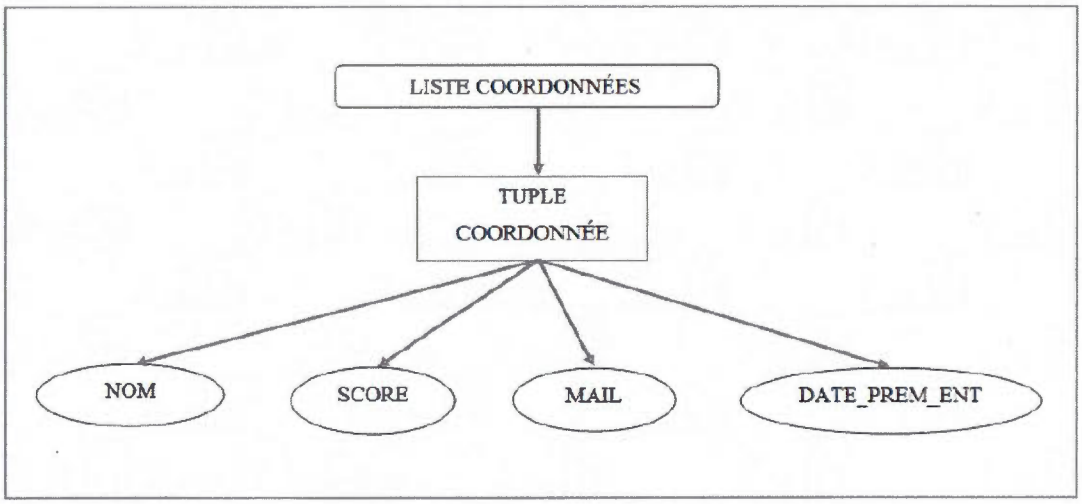


Figure 3.1 Arbre ECT du site Web OKRA

3.3.1.2 Chargement du site Web et création de l'arbre hiérarchique ECT

Cette phase permet de charger un nouveau site Web dont on veut extraire les informations. Un site est identifié par son nom et il possède un arbre unique ECT représentant le modèle général de la structure logique globale des pages. La figure 3.2 montre un écran de chargement du site Web OKRA.

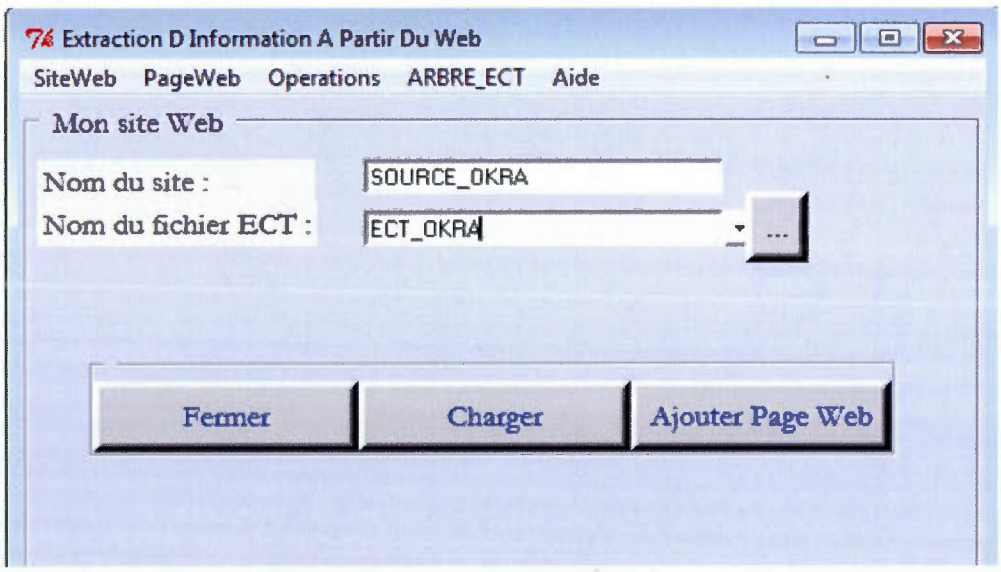


Figure 3.2 Écran de chargement d'un site Web

Construction de l'arbre ECT

Une fois le site Web est chargé et supposons que l'arbre ECT est déjà conçu manuellement sur papier, on peut commencer la construction de l'arbre ECT. Le processus de la construction de l'arbre ECT se fait en insérant ses nœuds un par un en indiquant pour chacun des nœuds les informations suivantes :

Type : Définit si le nœud est une liste, un tuple ou une feuille.

Nom : Un nom significatif approprié au nœud.

Type de données : Le type d'information que représente le nœud.

Nœud parent : Le nœud parent du nœud à insérer.

Type de la règle : C'est le type de la règle d'extraction du nœud, nous avons utilisé 2 types différents : SkipTo et SkipUntil.

Une fois tous les nœuds sont insérés et complétés par leurs informations, l'arbre construit sera enregistré comme arbre ECT global du site Web. Le schéma de la figure 3.3 illustre un écran de construction de l'arbre ECT avant l'insertion du nœud tuple COORDONNEE, et la figure 3.4 affiche l'arbre ECT après insertion de ce nœud.

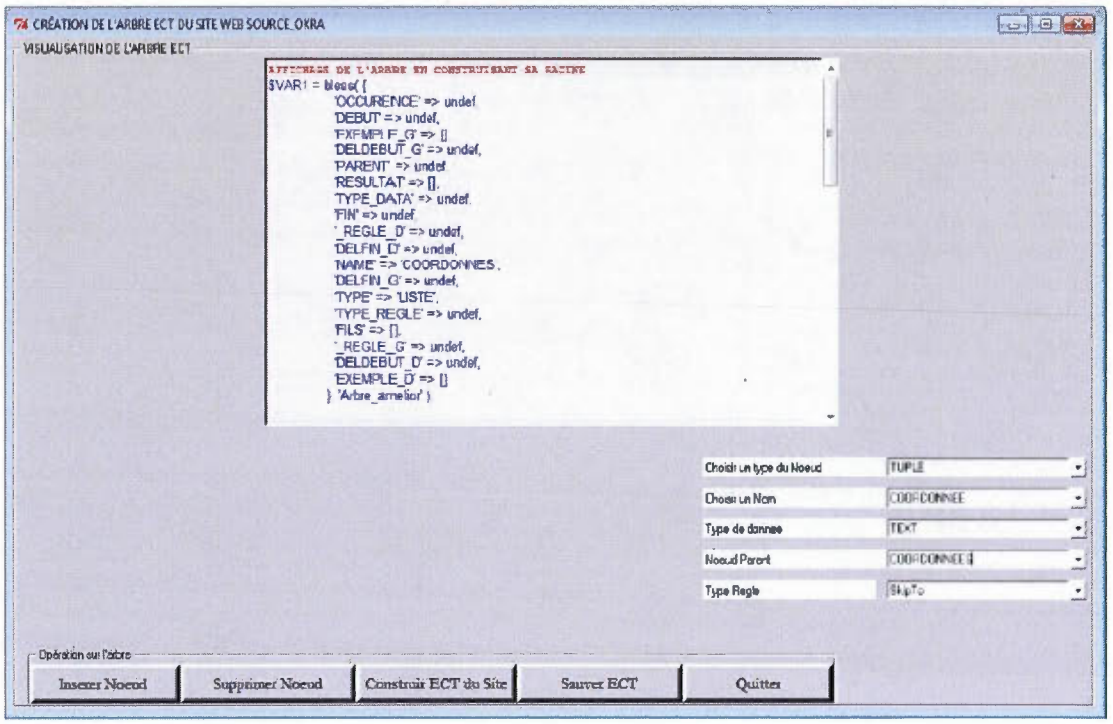


Figure 3.3 Écran de construction de l'arbre ECT avant l'insertion du nœud tuple COORDONNEE

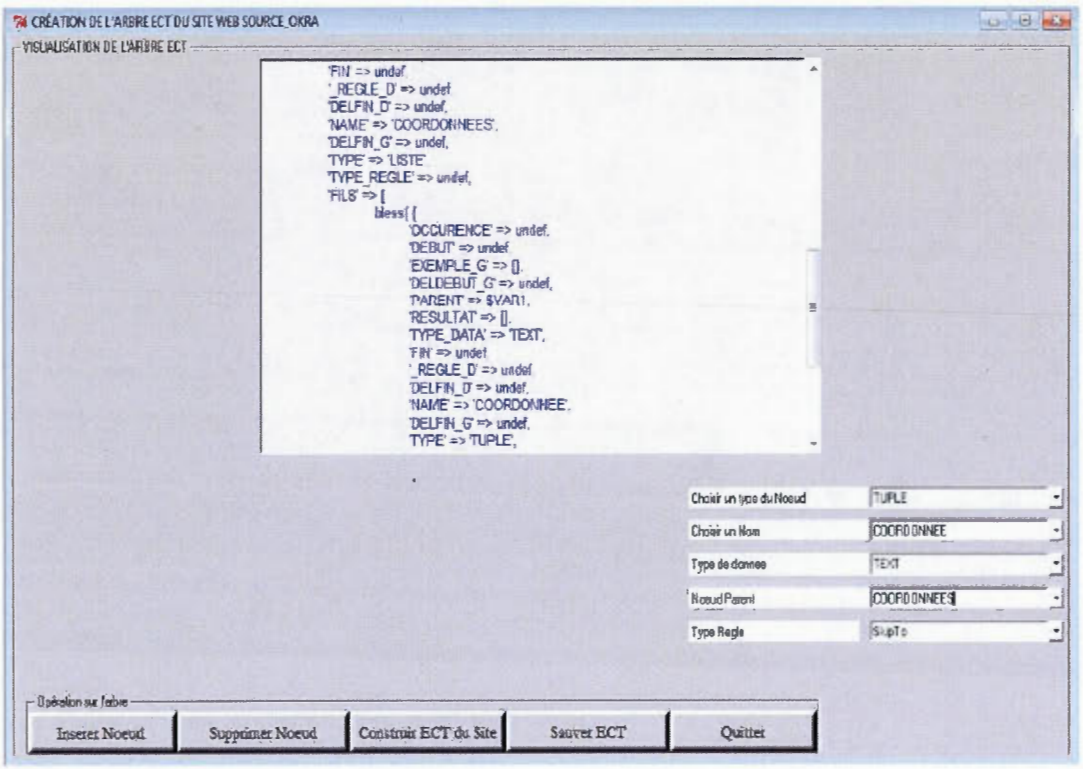


Figure 3.4 Écran de construction de l'arbre ECT après l'insertion du nœud

3.3.1.3 Apprentissage

Le processus d'apprentissage consiste à apprendre l'ensemble des règles d'extraction gauche et droite de chacun des nœuds de l'arbre ECT. Tout d'abord, on doit sélectionner une page Web appartenant au site, et on affiche son code Html. Ensuite, on doit procéder à la sélection des nœuds de l'arbre ECT un par un pour insérer des exemples d'entrainement. Une page Web est identifiée par son nom et son lien Web représentant un lien url ou un chemin définissant son emplacement d'enregistrement sur le disque dur. La sélection d'un tel exemple se fait par le click dans la zone du texte contenant le code Html en enregistrant la position du curseur comme délimiteur de début et de fin de l'exemple. Le texte contenu entre ces délimiteurs sera automatiquement ajouté comme exemple d'entrainement au nœud sélectionné. Une fois les exemples nécessaires pour l'apprentissage choisis, on invoque notre algorithme d'apprentissage afin de déduire les règles d'extraction appropriées aux nœuds. Le schéma de la figure 3.5 ci-après montre un écran du processus d'apprentissage. Dans cet exemple, il s'agit d'apprendre les règles d'extraction du nœud feuille 'Nom'.

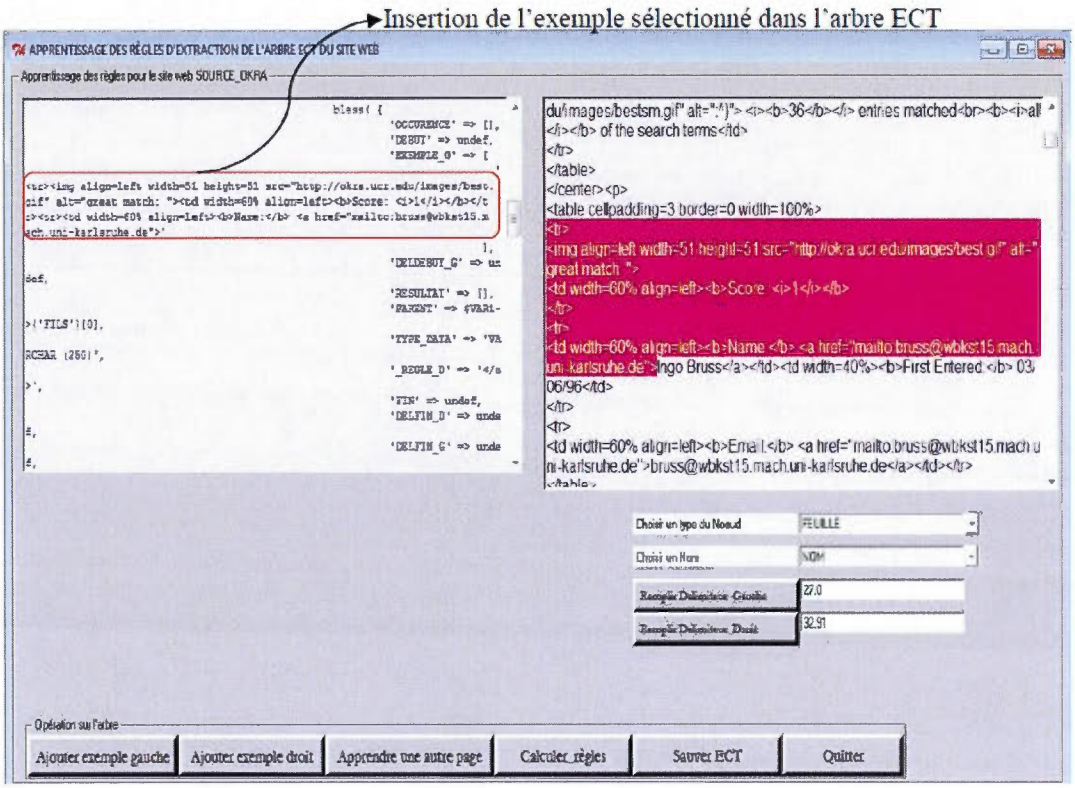


Figure 3.5 Écran d'apprentissage des règles d'extraction

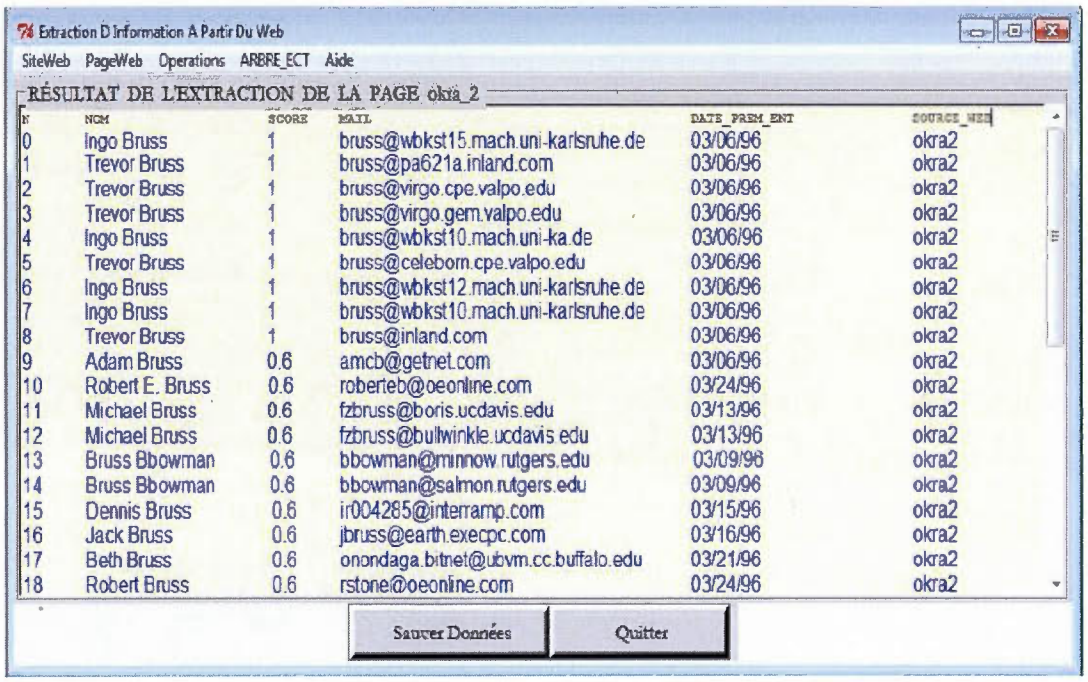
3.3.1.4 Extraction

Le processus d'extraction consiste en l'extraction de données provenant d'une source quelconque appartenant au site Web. On doit tout d'abord saisir le lien url de la page Web à extraire (un chemin enregistré sur le disque dur de l'ordinateur est suffisant), ensuite on sélectionne le nom du site Web, origine de la page. Le nom du site Web permet de déduire l'arbre ECT approprié à la page à extraire. On affecte le code Html de la page à l'attribut du nœud racine de l'arbre ECT, et en appliquant récursivement notre algorithme d'extraction sur l'arbre ECT. Le résultat de l'extraction permet d'afficher et d'enregistrer seulement les feuilles de l'arbre ECT. Le schéma de la figure 3.6 illustre un écran de l'extraction de la page Web okra_2. Tout d'abord, on insère une page Web à extraire et on sélectionne dans la liste son site Web approprié. Dans cet exemple, le lien de la page est 'file:///C:/Users/PERSON/Desktop/Repository/OKRA/okra-2.html', il représente un chemin

de l'emplacement d'enregistrement de la page Web sur le disque dur. Ainsi le nom du site Web est SOURCE_OKRA, et le nom identifiant la page Web est okra_2. Les données seront enregistrées dans un fichier texte afin de l'utiliser après pour remplir notre base de données. L'insertion des données dans la base de données se fait à partir du fichier texte en insérant les informations dans les champs de la table correspondante. Les champs de la table de données représentent les informations extraites (feuilles de l'arbre ECT). La table porte le nom du site Web à extraire, dans cet exemple c'est SOURCE_OKRA.

Figure 3.6 Écran d'extraction de la page Web okra_2

L'écran de la figure 3.7 montre un affichage des résultats de la page okra_2. La figure 3.8 illustre un écran d'affichage des résultats d'extraction de la page okra_2 dans le fichier texte.



	NOM	SCORE	MAIL	DATE_PREM_ENT	SOURCE_WEB
0	Ingo Bruss	1	bruss@wbkst15.mach.uni-karlsruhe.de	03/06/96	okra2
1	Trevor Bruss	1	bruss@pa621a.inland.com	03/06/96	okra2
2	Trevor Bruss	1	bruss@virgo.cpe.valpo.edu	03/06/96	okra2
3	Trevor Bruss	1	bruss@virgo.gem.valpo.edu	03/06/96	okra2
4	Ingo Bruss	1	bruss@wbkst10.mach.uni-ka.de	03/06/96	okra2
5	Trevor Bruss	1	bruss@celeborn.cpe.valpo.edu	03/06/96	okra2
6	Ingo Bruss	1	bruss@wbkst12.mach.uni-karlsruhe.de	03/06/96	okra2
7	Ingo Bruss	1	bruss@wbkst10.mach.uni-karlsruhe.de	03/06/96	okra2
8	Trevor Bruss	1	bruss@inland.com	03/06/96	okra2
9	Adam Bruss	0.6	amcb@getnet.com	03/06/96	okra2
10	Robert E. Bruss	0.6	roberteb@oeonline.com	03/24/96	okra2
11	Michael Bruss	0.6	fzbruss@boris.ucdavis.edu	03/13/96	okra2
12	Michael Bruss	0.6	fzbruss@bullwinkle.ucdavis.edu	03/13/96	okra2
13	Bruss Bbowman	0.6	bbowman@minnow.rutgers.edu	03/09/96	okra2
14	Bruss Bbowman	0.6	bbowman@salmon.rutgers.edu	03/09/96	okra2
15	Dennis Bruss	0.6	ir004285@interramp.com	03/15/96	okra2
16	Jack Bruss	0.6	jbruss@earth.execpc.com	03/16/96	okra2
17	Beth Bruss	0.6	onondaga.bitnet@ubvm.cc.buffalo.edu	03/21/96	okra2
18	Robert Bruss	0.6	rstone@oeonline.com	03/24/96	okra2

Figure 3.7 Écran d'affichage des résultats d'extraction

La sauvegarde des données se fait dans une base de données dont la table porte le nom du site Web.

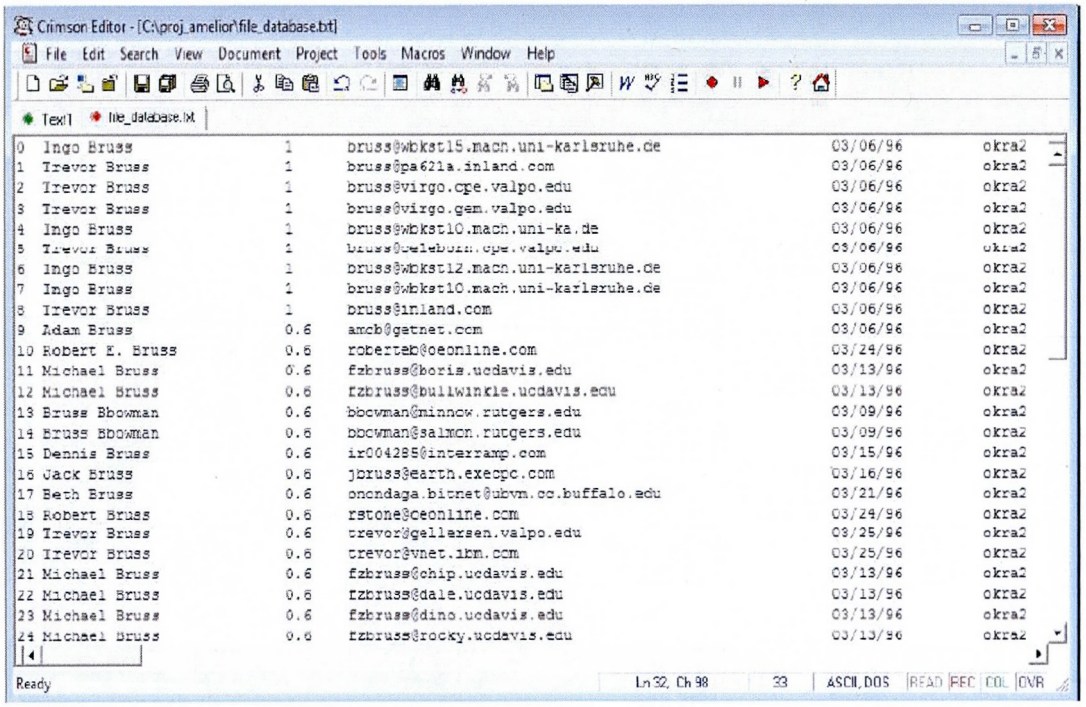


Figure 3.8 Écran d'affichage des résultats d'extraction dans le fichier texte

La figure 3.9 montre un écran d'affichage des données via l'environnement WampServer.

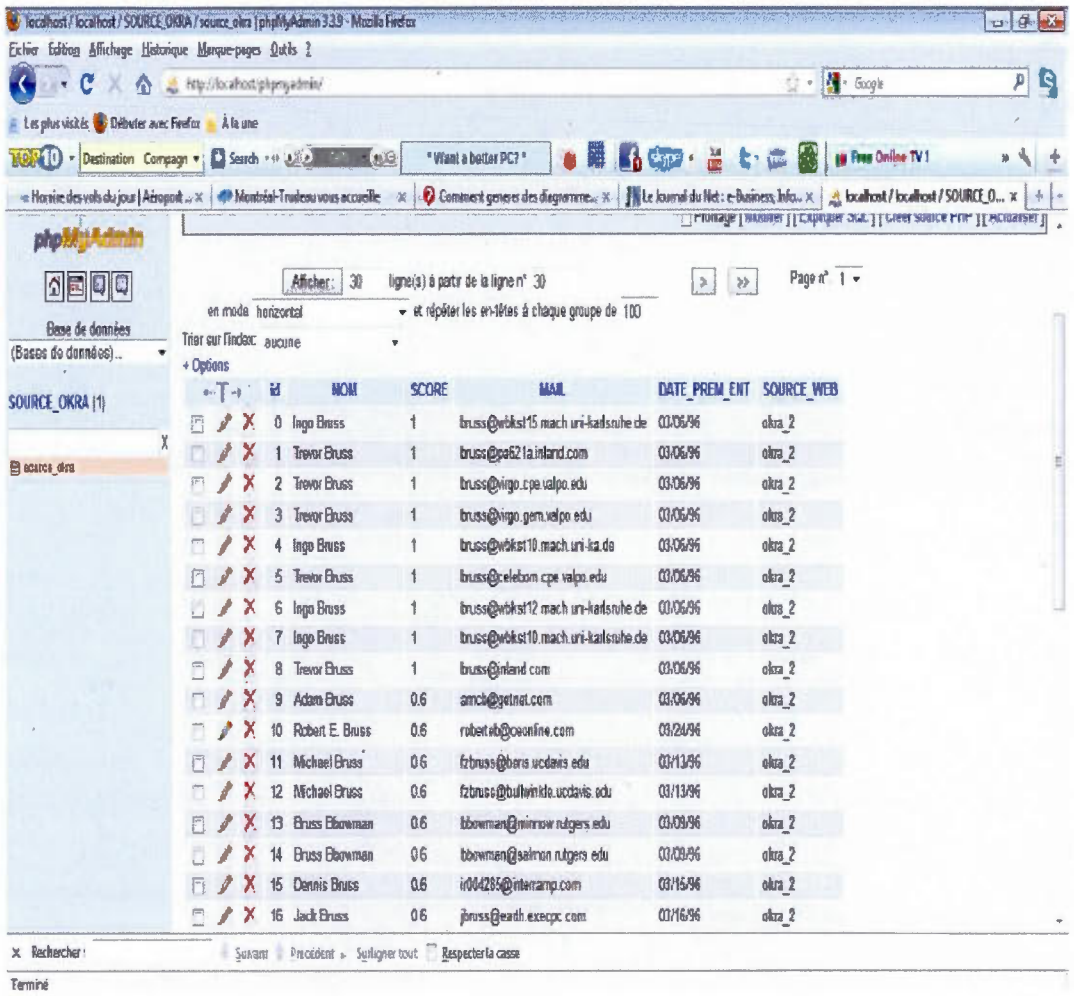


Figure 3.9 Écran d'affichage des données dans la base de données

3.4 Application

Dans cette section, nous allons présenter l'application de notre outil sur une source de données riche d'informations. Nous avons choisi le site Web www.kijiji.ca qui est un site Web très populaire et qui représente des annonces quotidiennement consultées par les gens du monde entier. Dans ce travail, nous avons fait l'extraction des annonces au complet du site Web kijiji, en se focalisant sur la catégorie des annonces des autos et véhicules. La figure 3.10 illustre un écran d'interrogation du site et d'extraction des annonces autos et camions à partir de ce site.

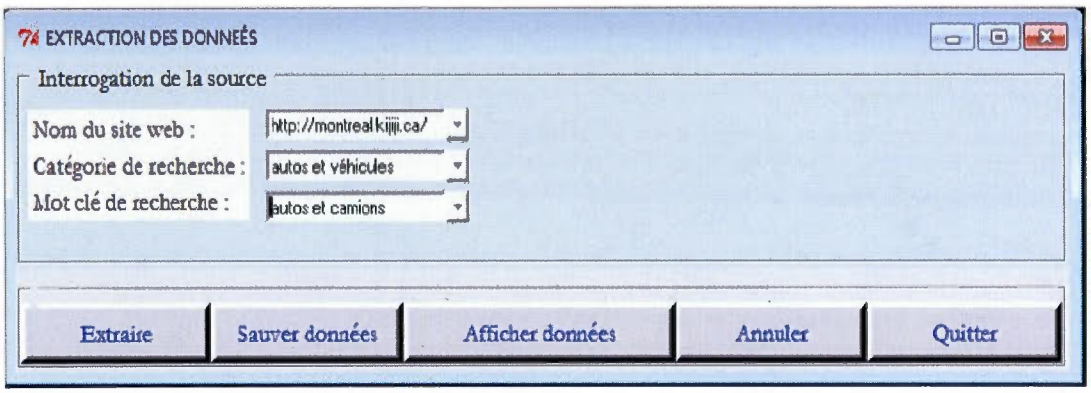


Figure 3.10 Écran d'interrogation et d'extraction des annonces

Nous avons extrait des informations provenant de la province du Québec. Au moment de l'extraction, nous avons remarqué que la source possède 2461 (voir figure 3.11 ci-après) pages Web, chaque page représente au minimum 23 annonces d'autos et camions. Au total, nous avons fait l'extraction de 49198 annonces. Notre outil nous a permis d'extraire toutes les annonces de la catégorie autos et camions de la province du Québec de façon simple, rapide et totalement automatique. Les résultats de l'extraction sont organisés et structurés dans le SGBD du serveur Wamp.

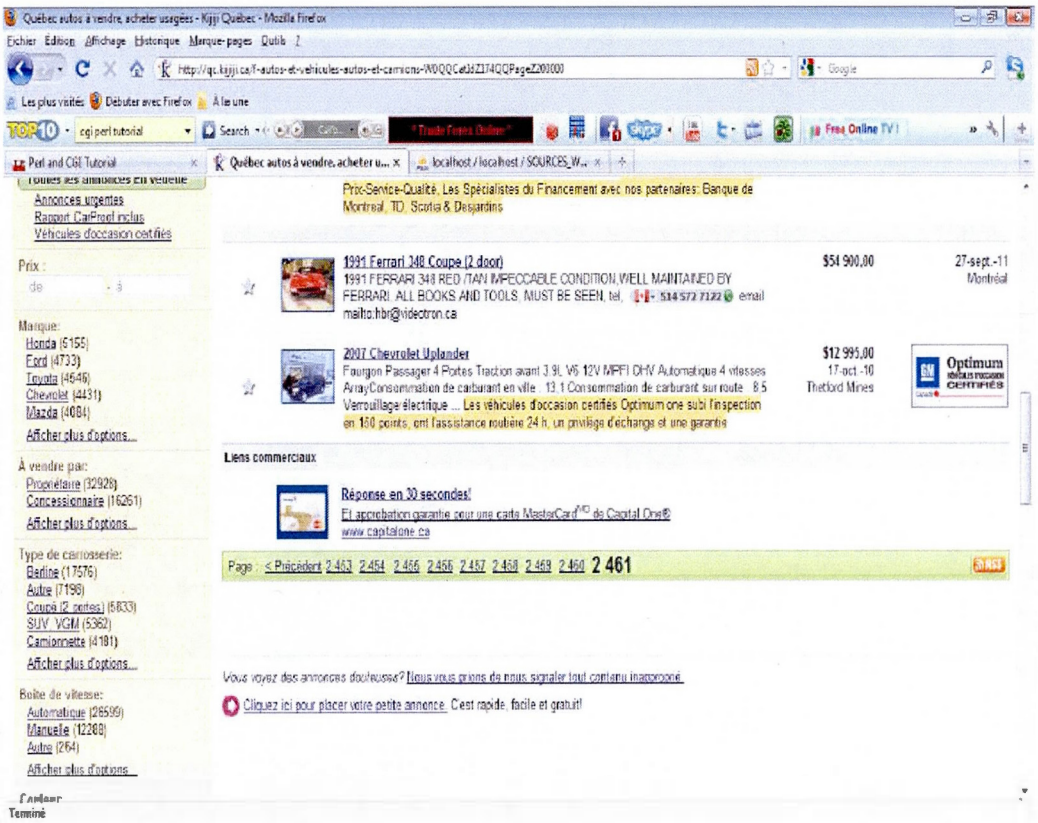


Figure 3.11 Écran de la source d'annonces autos et camions du site kijiji

La sauvegarde des résultats d'extraction dans une base de données a l'avantage de répondre aux besoins des utilisateurs qui font la consultation des annonces suivant des critères complexes. Supposons qu'un utilisateur veut extraire des annonces d'une adresse bien précise. Notre outil répond facilement à son besoin en interrogeant notre base de données par une simple requête MySQL. C'est un peu compliqué pour cet utilisateur de trouver sa réponse via le site kijiji. Tout d'abord, il doit choisir la catégorie des annonces qu'il cherche, ensuite, il fait la recherche par la ville correspondante à son adresse. Supposons que la réponse par ville retourne une centaine de pages Web, chaque page représente un nombre d'annonces, alors l'utilisateur est obligé de parcourir les pages une par une et de consulter l'ensemble des annonces par chacune des pages en vérifiant l'adresse de chaque annonce. Cette tâche est très complexe et aussi fastidieuse. La figure 3.12 ci-dessous illustre une interrogation de données, et la figure 3.13 affiche les résultats obtenus comme réponse au choix de l'utilisateur.

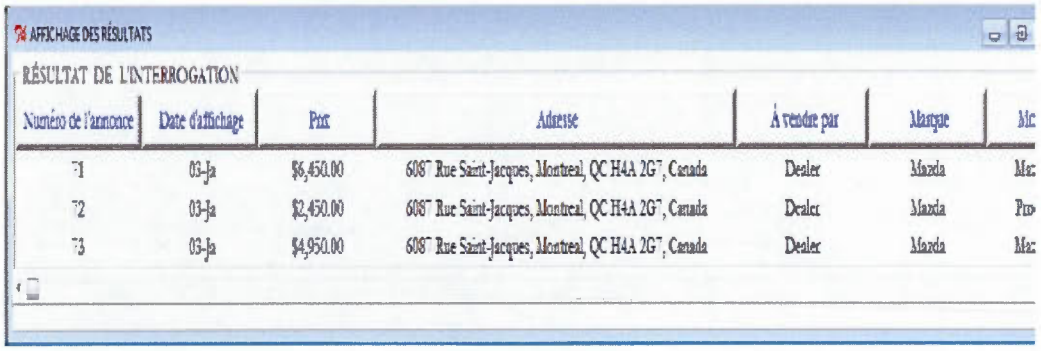


Figure 3.12 Interrogation des données de la source kijiji

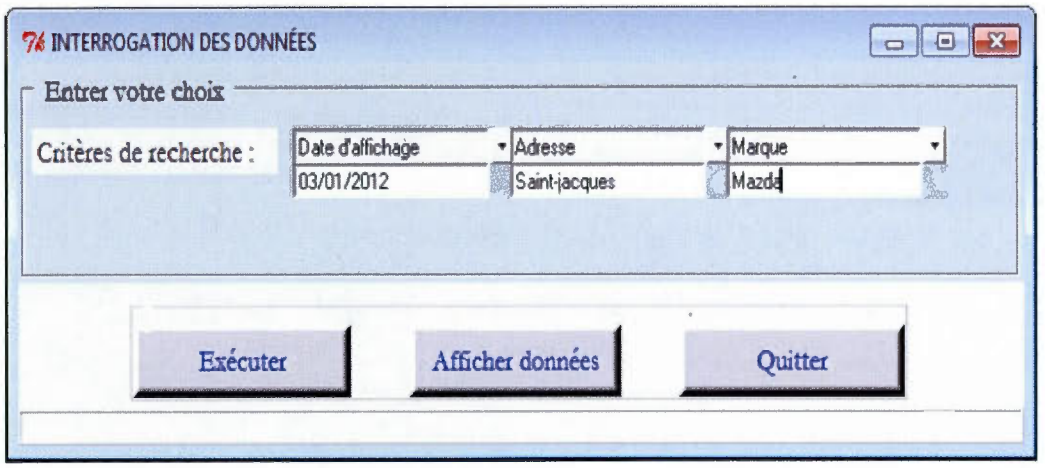


Figure 3.13 Réponse de la requête de la figure 3.12

D'une part, notre outil permet de faciliter le processus de recherche des informations pour les utilisateurs. D'autre part, l'extraction des données se fait en temps réel ce qui favorise la réutilisation de ces données résultats dans d'autres sources et application Web. Cet avantage ouvre la porte à de nombreuses applications de service Web. D'après les expériences précitées utilisant les algorithmes que nous avons développés. La figure 3.14 illustre le processus d'extraction des données à partir d'une source Web.

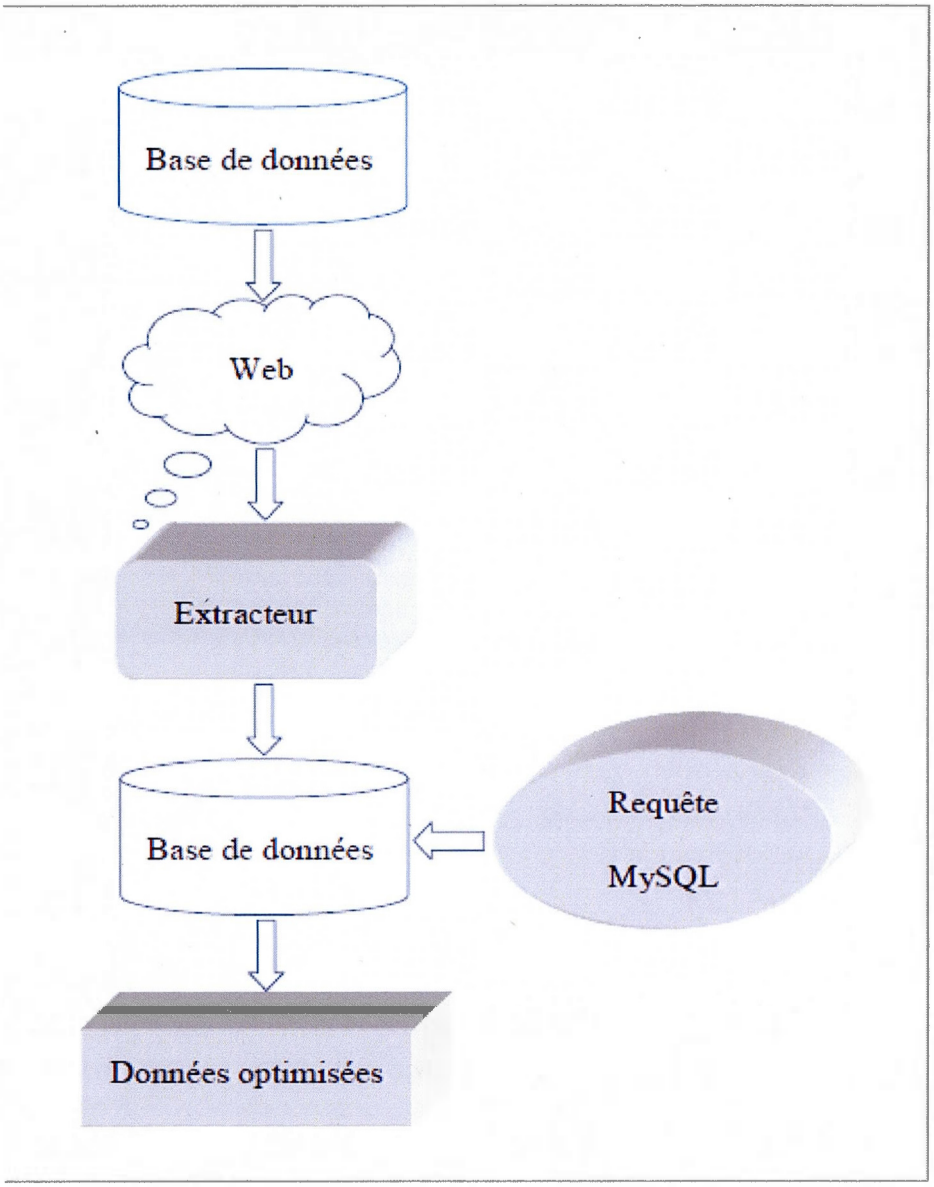


Figure 3.14 Processus d'extraction à partir d'une source Web

CHAPITRE IIV

EXPÉRIMENTATION

Dans ce chapitre, nous allons discuter les résultats de notre algorithme d'extraction. Afin de valider notre implémentation, nous avons fait une expérimentation sur les sites Web les plus populaires et qui ont été abordés dans la majorité des travaux. Nous nous sommes concentrés sur les articles traitant et publiant l'algorithme et l'extraction de STALKER.

4.1 Matériel utilisé

Pour l'implémentation de notre algorithme d'extraction, nous avons utilisé un mini-ordinateur portable comportant 230 GO d'espace disque dur, 2 GHZ de fréquence du microprocesseur double corps, et 4 GO de mémoire RAM. Le système d'exploitation utilisé est Windows Vista 32 bits version familiale.

4.2 Présentation des résultats

À cet effet, nous avons choisi 5 sites (OKRA[16], BIGBOOK[17], INTERNET ADRESSFINDER[18], LAWEEKLY[19] RESTAURANT, et QUOTE SERVER[20]) et on a testé notre algorithme sur chacun d'eux. Nous avons présenté les résultats dans le tableau 3.1. Ce tableau de résultats est réparti en colonne, la première colonne représente le nombre de feuilles dans chaque tuple, la deuxième représente le nombre des exemples d'entraînement utilisés pour chacun des sites Web, la troisième montre le temps d'apprentissage pour chaque site Web, la quatrième représente le nombre de données correctement extraites, la cinquième

[16] http://www.isi.edu/info-agents/RISE/wOKRA/__Source__.html
[17] http://www.isi.edu/info-agents/RISE/wBigBook/__Source__.html
[18] http://www.isi.edu/info-agents/RISE/wIAF/__Source__.html
[19] http://www.isi.edu/info-agents/RISE/LAW/__Source__.html
[20] http://www.isi.edu/info-agents/RISE/wQS/__Source__.html

montre le nombre de données non extraites, la sixième montre le temps d'extraction d'informations de chaque page Web du site, et la 7eme et la dernière représente le pourcentage d'extraction d'information.

Pour l'évaluation des résultats, nous avons utilisé le pourcentage des informations correctement extraites par rapport aux nombre total d'informations. Supposant que l'extraction d'une page contenant 50 informations à extraire, et que les résultats de l'adaptateur ont trouvé 50 informations dont 40 sont corrects et le reste représente des informations non extraites. La précision de l'extraction de la page sera alors 40 x 100 / 50 ce qui donne 80%. La précision d'extraction de la source Web représente la moyenne générale des précisions de chacune de page Web de la source.

$$\text{Précision (page web)} = \frac{Nombre\ d'informations correctement\ extraites}{Nombre\ d'informations totales}$$

$$\text{Précision (source web)} = \frac{\sum Précision\ page\ web}{Nombre\ pages\ web}$$

4.3 Choix des bons exemples d'entrainement

Pour chacun des sites Web et dans le but d'obtenir des meilleurs résultats, nous avons essayé de trouver les meilleurs exemples durant la phase d'apprentissage. Les résultats ont montré que la phase d'apprentissage était très difficile pour quelques sites Web. Par exemple, pour le site Web QUOTE SERVER, nous avons bien remarqué que la mise en forme des pages Web change fréquemment d'une page à l'autre. Ce changement a rendu la phase d'apprentissage ou plus précisément l'étape du choix des exemples d'entraînement très difficile à achever. Ce qui nous a obligés à refaire l'apprentissage à chaque fois que le choix des exemples d'entraînement a changé. Pour ce site, nous avons conclu que pour avoir une bonne précision d'extraction il nous faut 5 exemples d'entraînement provenant de 5 pages Web différentes.

Par contre l'apprentissage des règles d'extraction pour les sites Web OKRA, et BIGBOOK s'est fait en utilisant seulement 3 exemples d'entraînement. Les résultats ont montré que la précision de l'extraction est à 100%. Ainsi pour les sites Web INTERNET 90

ADRESS FINDER, et LAWEEKLY RESTAURANT le calcul des règles d'extraction était très simple suivant le nombre réduit des exemples d'entraînement choisis.

La difficulté rencontrée au moment de l'apprentissage est due souvent à la structure interne des pages Web. Cependant, nous avons des erreurs de frappe ou des informations mal saisies dans la page Web. Le nombre d'informations peut changer d'une page à l'autre ce qui mène à un manque d'informations et automatiquement va donner des résultats d'extraction non valides. Tous ces problèmes nous obligent à refaire l'apprentissage en utilisant des exemples d'entraînement beaucoup plus exhaustifs, et ceci pour augmenter la précision de l'extraction.

Bien que le temps d'apprentissage des règles est lié au nombre des exemples d'entraînement utilisés, il s'accroit chaque fois que le nombre des exemples augmente.

D'après les résultats obtenus, nous avons constaté que pour avoir une bonne précision d'extraction, il faut bien choisir les exemples d'entraînement. Toutefois, le fait d'utiliser un grand nombre d'exemples durant l'apprentissage ne suffit pas pour avoir une meilleure extraction. D'après notre expérience, on peut dire que la difficulté de l'algorithme se situe dans le choix des exemples d'entraînement car ce choix doit se faire d'une façon intelligente par un expert de domaine.

QUOTE SERVER							
La source	Temps d'pprentissage (sec)	Temps d'extraction (sec)	Nbre de tuple	Nbre total de données	Nbre de données fausses	Nbre de données correctes	Pourcentage
s29_01	1909,954999 (5 exemples)	0,013	2	36	6	30	83,33
s29_02		0,187	2	36	13	23	63,89
s29_03		0,219999	2	36	6	30	83,33
s29_04		0,242999	3	54	8	46	85,19
s29_05		0,197	3	54	21	33	61,11
s29_06		0,006999	2	36	2	34	94,44
s29_07		0,157	1	18	5	13	72,22
s29_08		0,013	2	36	4	32	88,89
s29_09		0,018	4	72	10	62	86,11
s29_10		0,012	4	72	23	49	68,06
Récap			25	450	98	352	78,22
BIGBOOK							
Récap	0,665999 (3 exemples)						100,00
OKRA							
Récap	0,195 (3 exemple)						100,00
LAWEEKLY							
eclectic.law	0,271779 (5 exemples)	0,00999	1	5	2	3	60,00
seafood.law		0,017999	3	15	0	15	100,00
thai law		0,018	3	15	1	14	93,33
polynesian.law		0,011	1	5	0	5	100,00
mexican.law		0,056999	9	45	0	45	100,00
mediterranean.law		0,010999	2	10	2	8	80,00
latin.law		0,005	8	40	0	40	100,00
japaneese.law		0,013999	2	10	0	10	100,00
italian.law		0,039	6	30	0	30	100,00
indian.law		0,015999	2	10	0	10	100,00
healthy.law		0,009	1	5	2	3	60,00
global.law		0,059999	11	55	6	49	89,09
french.law		0,042999	7	35	0	35	100,00
cuban.law		0,014999	2	10	0	10	100,00
coffee.law		0,006	1	5	0	5	100,00
chineese.law		0,026	4	20	4	16	80,00
carribean.law		0,016	2	10	0	10	100,00
cajun.law		0,023	6	30	0	30	100,00
american.law		0,054999	9	45	3	42	93,33
west_la.law		0,036999	6	30	4	26	86,67
west_holywood.law		0,061999	11	55	4	51	92,73
vlley.law		0,052999	9	45	0	45	100,00
la.law		0,12	24	120	4	116	96,67
hollywood.law		0,032999	5	25	3	22	88,00
east_la.law		0,010999	1	5	1	4	80,00
burbank_glendale		0,022	3	15	0	15	100,00
beverly_hils		0,031	6	30	1	29	96,67
beach_cities		0,082999	15	75	0	75	100,00
Récap			160	800	37	763	95,375
INTERNET ADRESS FINDER							
s11-1	0,433 (3 exemples)	0,013	10	60	0	60	100
s11-2		0,023999	8	48	0	48	100
s11-3		0,029	10	60	0	60	100
s11-4		0,009	2	12	0	12	100
s11-5		0,029	10	60	0	60	100
s11-6		0,476	10	60	0	60	100
s11-7		0,027999	10	60	0	60	100
s11-8		0,026999	10	60	0	60	100
s11-9		0,671	10	60	1	59	98,33333333
s11-10		0,017	10	60	0	60	100
Récap			90	540	1	539	99,81481481

Tableau 4.1 Résultat d'apprentissage et extraction des sites Web

Nous avons aussi testé notre approche sur le site Web RENTAL[21]. Le tableau 4.2 ci-dessous illustre les résultats trouvés durant les deux phases d'extraction. D'après le tableau, on peut facilement remarquer que le temps d'apprentissage pour le site RENTAL est très réduit, l'utilisation de la notion du contexte d'information permet de trouver facilement et directement la bonne règle d'extraction. Ce contexte d'information permet de déduire la règle d'extraction même si la mise en forme des pages a changé car la règle d'extraction est liée au type de données et pas au format du site Web. Cet avantage nous permet de ne pas prendre en considération la reconstruction de l'adaptateur une fois la forme des pages a changé. Le même adaptateur peut fonctionner correctement sur les nouvelles pages.

RENTAL							
source	0.6 (5 exemples)	405	3	1215	50	1165	0,958847737
Récap	0.6 (5 exemples)	405	3	405	50	1165	0,958847737

Tableau 4.2 Résultat d'apprentissage et extraction du site Web RENTAL

[21] http://www.isi.edu/info-agents/RISE/Rentals/__Source__.html

CONCLUSION

Dans ce mémoire, nous présentons une méthode d'extraction de données à partir du web. Nous avons présenté un adaptateur capable d'extraire des données provenant de pages web semi-structurées. Notre méthode s'inspire du système STALKER dans la représentation de la page web. Nous avons intégré la notion de séparateur du système SOFTMEALY afin de donner plus d'expressivité au délimiteur STALKER. Dans la première partie de ce document, nous avons présenté une introduction générale sur le principe et l'utilité de l'extraction des données. Par la suite, nous avons consacré la deuxième partie de ce mémoire à la revue de littérature. Nous avons commencé par présenter les différentes approches et travaux qui précédent notre recherche en se basant sur une classification de quatre catégories : méthodes manuelles, méthodes semi-automatiques supervisées, méthodes semi-automatiques semi-supervisées et méthodes automatiques non supervisées. Nous avons clôturé ce chapitre par une comparaison des approches étudiées. Dans la troisième partie, nous avons décrit en détail notre algorithme et son principe de fonctionnement. Nous avons présenté les différents processus de notre adaptateur : la construction de l'arbre hiérarchique ECT, l'apprentissage et l'extraction. Ensuite, dans la quatrième partie, nous avons expérimenté notre algorithme en expliquant les difficultés rencontrées durant le processus de l'extraction. Finalement, la dernière partie de ce mémoire présente l'implémentation de notre algorithme. Nous avons commencé par définir les outils et technologies utilisés et le pourquoi du choix de ces outils. Ensuite, nous avons présenté en détail le déroulement de notre prototype en se basant sur un exemple de source Web. À la fin de ce manuscrit, nous avons illustré notre approche hybride (STALKER, SOFTMEALY) à travers une application de médiation de la source de données kijiji.

Notre approche a montré de meilleurs résultats dans la précision de l'extraction ainsi qu'au niveau performance d'apprentissage des règles d'extraction. L'introduction du type de la valeur de l'attribut à extraire en plus de délimiteur utilisé dans l'approche STALKER, a

permis la reconnaissance des informations qui n'ont pas été reconnues par le système STALKER. Notre approche se distingue également par le fait qu'elle est basée sur le contenu informationnel et non pas sur les aspects de mise en forme. Ce qui permet sa réutilisation lorsque des sites Web changent de mise en forme fréquemment. Cette approche peut éventuellement être améliorée pour qu'elle soit applicable non seulement sur des documents HTML semi-structurés mais aussi sur des documents bruts non formatés.

RÉFÉRENCES

[A. Leipzig, 2008] A. Leipzig. A comparison of HTML-aware tools for Web Data extraction, September, 2008.

[A. Saiiuguet et F. Azavant, 2001] A. Saiiuguet et F. Azavant. Building Intelligent Web Applications Using Lightweight Wrappers, Data and Knowledge Eng., vol. 36, no. 3, 283-316, 2001.

[B. Adelberg, 1998] B. Adelberg. NoDoSE : A Tool for Semiautomatically Extracting Structured and Semistructured Data from Text Documents," SIGMOD Record, vol. 27, no. 2, pp. 283-294, 1998.

[B. HABEGGER, 2004] Benjamin HABEGGER. Extraction d'informations à partir du Web, Décembre 2004.

[B. Liu, 2005] B. Liu. DATA MINING, Exploring Hyperlinks, Contents, and Usage Data : Bing Liu, 2005.

[C.H. Chang et S. Kuo, 2004] C.H. Chang et S. Kuo. OLERA : A Semi-supervised Approach for Web Data Extraction with Visual Support, 2004.

[C.H. Chang et S. Lui, 2001] C.H. Chang et S.C. Lui. IEPAD : Information Extraction Based on Pattern Discovery, Proc. 10th Int'l Conf. World Wide Web (WWW), pp. 223-231, 2001.

[C. Hsu, 1998] C. Hsu. Initial Results On Wrapper Semistructured Web Pages With Finite-State Tranducers And Contextuels Rules, 1998.

[C.A. Knoblock et al., 1998] C.A. Knoblock, S. Minton, J.L. Ambite, N. Ashish, P.J. Modi, I. Muslea, A. Philpot, et S. Tejada. Modeling web sources for information integration. In Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI), 998

[C.H. Chang et al., 2006], C.H. Chang, M. Kayed, M.R. Girgis, et K.F. Shaalan. A Survey of Web Information Extraction Systems, 2006.

[D.W. Embley et al., 2006], D.W. Embley, D.M. Campbell, Y.S. Jiang, S.W. Liddle, Y.K. Ng, D. Quass, et R.D. Smith. Conceptual-Model-Based Data Extraction from Multiple-Record Web Pages, Data and Knowledge, 1999.

[H. Alberto et al., 2004] H. Alberto, A. RibeiroNeto, L. Berthier, S. Teixeira, et A.S. da Silva Juliana. A Brief Survey of Web Data Extraction Tools, 2004.

[I. Muslea et al., 1998] I. Muslea, S. Minton, et C.A. Knoblock. Learning Extraction Rules For Semistructured Web-based Information Sources, 1998.

[I. Muslea et al., 1998] I. Muslea, S. Minton, et C.A. Knoblock. Hierarchical Wrapper Induction For Semistructured Information Sources, 1998.

[I. Muslea et al., 1999] I. Muslea, S. Minton, et C.A. Knoblock. Active Learning for Hierarchical Wrapper Induction, 1999.

[I. Muslea et al., 2000] I. Muslea, S. Minton et C.A. Knoblock. Selective SamplingWith Naive Co-Testing: Preliminary Results, 2000.

[I. Muslea et al., 2001] I. Muslea, S. Minton et C.A. Knoblock. Selective Sampling With Redundant Views, 2001.

[J. Hammer et al., 1997] J. Hammer, H. Garcia-Molina, J. Cho, R. Aranha, et A. Crespo. Extracting Semistructured Information from the Web. In Proceedings of the Workshop on Management of Semistructured Data. Tucson, Arizona, May 1997.

[J. Hammer et al., 1997] J. Hammer, J. McHugh, et H. Garcia-Molina. SemistructuredData: The TSIMMIS Experience, Proc. First East-European Symp.Advances in Databases and Information Systems (ADBIS), pp. 1-8, 1997.

[J. Wang et F.H. Lochovsky, 2002] J. Wang et F.H. Lochovsky. DELA : Wrapper Induction Based on Nested Pattern Discovery, Technical Report HKUST-CS-27-02, Dept. of Computer Science, Hong Kong, Univ. of Science & Technology, 2002.

[M. Michelson et C.A. Knoblock, 2006] M. Michelson et C.A. Knoblock. Phoebus: A System for Extracting and Integrating Data from Unstructured and Ungrammatical Sources, 2006.

[M. Michelson et C.A. Knoblock, 2004] M. Michelson et C.A. Knoblock. Semantic Annotation of Unstructured and Ungrammatical Text, 2004.

[M. Kifer et al., May 1995] M. Kifer, Georg Lause, et James Wu. F- Logic : Logical Foundations of ObjectOriented and FrameBased Languages, May 1995.

[M. Rowe, 2005] M. Rowe. Wrapper Implementation Form Information Extraction From House Music Web Sources, May 2005.

[N. Kushmerick et al., 1997] Nicholas Kushmerick, S.W. Daniel, R. Doorenbos. Wrapper Induction For Information Extraction, 1997.

[N. Kushmerick, 1997] Nicholas Kushmerick. Wrapper Induction For Information Extraction, 1997.

[N. Kushmerick, 2003] Nicholas Kushmerick. Learning to invoke Web forms and services, 2003, Computer Science Department, University College Dublin, Ireland, 2003.

[S. Soderland, 1999] S. Soderland. "Learning Information Extraction Rules for Semi-Structured and Free Text," J. Machine Learning, vol. 34, nos. 1-3, pp. 233-272, 1999.

[T. Kirk et al., 1995] T. Kirk; A.Y. Lev.; Y. Sagiv et D. Srivastava. The information manifold. In Working Notes of the AAAI Spring Symposium on Information Gathering in Heterogeneous, Distributed Environments, 1995.

[V. Crescenzi et G. Mecca, 1998] V. Crescenzi and G. Mecca. Grammars Have Exceptions, Information Systems, vol. 23, no. 8, pp. 539-565, 1998.

[V. Crescenzi et al., 2001] V. Crescenzi, G. Mecca, et P. Merialdo. "RoadRunner: Towards- Automatic Data Extraction from Large Web Sites," Proc. the 26th Int'l Conf. Very Large Database Systems (VLDB), pp. 109-118, 2001.

[W. COHEN, 1998] W. COHEN. Data Integration Using Similarity Joins and a Word-Based Information Representation Language, 1998.

[W. May, G. Lausen, 2004] W. May, G. Lausen. A uniform framework for integration of information from the web, 2004.

[X. Gao et al., 2004] X. Gao, P. Andreae, et R. Collins. Approximately Repetitive Structure Detection for Wrapper Induction. In PRICAI, pages 585.594, 2004.

http://www.isi.edu/info-agents/RISE/repository.html.

http://www.isi.edu/info-agents/RISE/wOKRA/__Source__.html.

http://www.isi.edu/info-agents/RISE/LAW/__Source__.html.

http://www.isi.edu/info-agents/RISE/wBigBook/__Source__.html.

http://www.isi.edu/info-agents/RISE/wIAF/__Source__.html.

http://www.isi.edu/info-agents/RISE/wQS/__Source__.html.