WEB DATA EXTRACTION AND ESTABLISHMENT OF A META SOCIAL NETWORK

CHARA AZIZ
Faculty of Science Ben M'sik, University Hassan II, Casablanca, Morocco

BENLAHMAR HABIB
Faculty of Science Ben M'sik, University Hassan II, Casablanca, Morocco

ABDERRAHIM TRAGHA
Faculty of Science Ben M'sik, University Hassan II, Casablanca, Morocco

ELGUEMMAT KAMAL
Faculty of Science Ben M'sik, University Hassan II, Casablanca, Morocco

RACHDI MOHAMED
Faculty of Science Ben M'sik, University Hassan II, Casablanca, Morocco

Abstract: Currently, Internet is considered as the main information source, this information is not directly exploitable by computers, that's what make the importance of the problem of information extraction from the web. Our article is about the “gene/clone” approach that is an approach for extracting information from the web, it consist on generating a gene from a set of examples instances entered by the user. The gene designates the smallest repetitive structure containing all values of examples instances. The search of the clones of gene allows to retrieve the other instances of the relationship to extract.

The objective of this paper is to improve the accuracy of the Gene / Clone approach, so we propose new concepts namely the structural prefix and structural suffix. The use of structural prefixes and structural suffixes allows to increase the accuracy of the gene and so improve the accuracy of data extraction. Also in this article the, we give an evaluation of these two methods on several data sources. At the end of the article, we propose a direct use of our approach, in fact, we have developed an application for extracting and gathering data from several social networks in order to generate a meta social network.

Keywords: Data extraction, gene, accuracy, structural prefix, structural suffix, meta social network.

1. INTRODUCTION

Today, the web contains many online data sources such as electronic trading sites, electronic directories, search engines, etc. Data from these sources are not directly usable by computers. However, some formatting regularities of the web pages can be exploited to extract from these pages the structured data needed by the user.

The objective of extracting information is to allow a computer to access to data from web sources. The problem of information extraction from the web consist on building programs (wrappers) or patterns or gene able to extract from a web page a part of its content and to put it into a structured format.

Depending on cases, a wrapper, can be built to be applied to a set of documents coming from the same source, a set of documents belonging to the same domain, or at any document of the web. Such wrapper takes often the form of a generic program applying a set of extraction rules to a web page.

The problem of extracting data web represents a strong challenge for the coming years. The development of a system of extracting data web becomes a great need for companies that aim to control events (news about the trade market) and comparison of markets or marketing. Also, the development of a system of extracting data web is necessary for the generation of meta search engines, price comparators, catalogs of libraries.

The objective of this paper is to propose an improvement of the “gene/clone” method. In this paper, we give a formalism of notions of structural prefix and structural suffix (new concepts), and we expose the formalism of a new method called “gene/clone with structural prefixes and suffixes”; this method is an improvement of the “Gene/clone” method.

Also in this article the, we give an evaluation of these two methods on several data sources to prove that the method of Gene /clone with structural prefixes and suffixes is better than the Gene /clone method in terms of accuracy of the extraction results.

At the end of the article, we give the description of an application that we developed, this is an application for extracting and gathering data from several social networks.

2. DIFFERENT APPROACHES FOR INFORMATION EXTRACTION FROM THE WEB

In the literature, there are several approaches enabling to obtain either a wrapper, patterns or a gene.
We find the inductive approach \([1,2,3]\) that consists on building a wrapper from a set of labeled examples pages introduced by the user, the structural approach \([4,5]\) which allows to build patterns by analyzing the structure of html pages, the knowledge-based approach \([6,7]\) that consists on building a wrapper for a domain using the knowledge of the concerned domain.

There is also the approach of extracting relations \([8]\) that enables to build patterns from a set example instances of a given relation.

Finally there is the “gene/clone” approach that consists on generating a gene from a set of example instances introduced by the user. The gene designates the smallest repetitive structure containing all values of example instances. The search of the clones of the gene enables to recover the other instances of the relation to extract.

3. THE “GENE/CLONE” APPROACH

3.1. PRESENTATION OF THE “GENE/CLONE” APPROACH

The “gene/clone” approach proposed by Benlahmer El Habib, Doukkali Sdiguizi Aziz, Asmaa Elouerkhaoui in \([9]\) is used to extract information from the web, it consists of using a set of examples instances and working on the structure of the source documents.

In this method, it is necessary to transform the html pages, from which we want to extract the relevant information into xhtml or xml pages in order to benefit of the tree structure of xml documents.

To know in advance the relevant information to extract, the user uses some examples in order to express his need in terms of relevant information to extract. Each example is called instance or example instance.

Definition 1 : An instance is an n-tuple composed of n values such that this n-tuple is an instance of the relation to extract.

The objective of the “gene/clone” approach is to generate the gene that is the smallest repetitive substructure containing all examples instances values. Since this structure is repetitive, the search of its clones allows recovering the other instances of the relation to extract. Instead of searching the relevant information we search the relevant nodes.

Definition 2 : A relevant node is a tag which the textual content contains the relevant information.

A relevant node could have one or many occurrence which does not contain the relevant information. So, knowing the node does not solve the problem. Therefore, we must know the xpath of the node.

A value of an instance is not necessary the whole textual content of a node, it may presents only a part of textual content. For example, the telephone on the online telephone directory may sometimes appear just after the string ”Tel.”.

We define the textual suffix (respectively prefix) of a value \(t_i\) as the textual string which is just after(resp before) \(t_i\) and that must verify the following condition :

Condition 1: A textual suffix (resp. prefix) should not contain a value \(t_j\) of an instance \(t_i\) if this is the case the textual suffix (resp. prefix) will be the textual prefix (resp. suffix) of the value \(t_j\) in the initial textual suffix (resp. prefix).

3.2. THE PROCESS OF THE “GENE/CLONE” APPROACH

We take a document \(D\) from a Web data source \(S\).

The extraction rules construction process is the following: First of all, the html document \(D\) which the user wishes to extract the structural data is transformed to xhtml document.

Next, to express his needs in terms of information to extract, the user enters some examples instances.

Then we seek the values of examples instances in the document and we construct the gene which is the smallest structure containing all the values of examples instances. Finally to find all clones of the gene and recover the other instances of the relation to extract, we build an xslt file that contains all the structural information of the gene and we apply this file to all documents of the source \(S\).

3.3. CONSTRUCTION OF THE GENE AND THE EXTRACTION RULES OF CLONES

Definition 3 : (the value context -the textual context-the structural context)

We define the value context of a value \(t_i\) by a triplet formed by the textual prefix of \(t_i\), the textual suffix of \(t_i\) and the xpath of the relevant node \(N_i\) whose the textual content contains \(t_i\).

The textual prefix and the textual suffix of a value \(t_i\) represent the textual context of the value \(t_i\) and the xpath of \(N_i\) represents the structural context of \(t_i\).

Definition 4 : (the context of an instance )

We define the context of an instance \(t=(t_1,t_2,\ldots,t_n)\) as the smallest structure that contains all values of the instance \(t\). It is also the xpath of the parent of the instance, it is the node that contains all nodes where appear values \(t_i\) such that there is no son of the node that contains all the values \(t_i\).

Definition 5 : (the global context )

We take two instances \(t=(t_1,t_2,\ldots,t_n)\) and \(t'=(t'_1,t'_2,\ldots,t'_{n})\),

We define the global context as the smallest structure containing all the values of instances \(t\) and \(t'\), formally it is the xpath of the parent of the two instances.

We suppose that a user has introduced two examples instances \(t=(t_1,t_2,\ldots,t_n)\) and \(t'=(t'_1,t'_2,\ldots,t'_{n})\)

To generate the gene, we seek the context of value for each value \(t_i\) and \(t'_i\), next, from structural contexts of values \(t_i\) and \(t'_i\), we construct the context of each instance. Then from the contexts of instances we build the global context.

We define the gene as the union of the global context, context of instances, and contexts of value.

In order to extract data of the clones, we construct an xslt file that contains all the structural information of the gene and we apply this file to the all documents of the source from which the user wants to extract.

In fact, from the global context we can locate the target structure, that is to say the substructure that contains all relevant information, then, from the context of instance we can access to each substructure containing a block of relevant information, and for each block we access, through
the contexts of value, to the relevant information one by one.

**Note:** generally, it is possible to generate a gene from a single instance; in this case, the gene is the union of the context of the instance and the value contexts. It is also the smallest structure containing all the values of the instance.

### 3.4. THE LIMITATIONS OF THE “GENE/CLONE” APPROACH

In the “gene/clone” approach, the use of textual context (textual prefix and textual suffix) allows to increase the accuracy and reliability of the gene. In fact, the structural context may not be sufficient to generate a reliable gene.

For example: it could be that two values \( t_i \) and \( t_j \) of an instance \( t \) have the same structural context; In this case, if each of these two values both hasn’t neither a textual prefix nor a textual suffix, then these two values will be returned by the Gene / clone method in a single block.

It is also possible that a value \( t_i \) of an instance \( t \) and an irrelevant value \( v \), have the same structural context; in this case, if the value \( t_i \) hasn’t neither a textual prefix nor a textual suffix, then the two values will likely be returned by the method in a single block.

Thus we can conclude that for the sources where there is absence of textual prefixes and textual suffixes, the “Gene / clone” approach can generate an unreliable gene.

### 3.5. SOLUTION

In order to remedy to all limitations of the “gene/clone” approach we will introduce new concepts that do not exist in the literature, namely the notions of structural prefix and structural suffix.

Thanks to structural prefixes and structural suffixes we can significantly improve the accuracy and reliability of the gene, especially in the case of the absence of textual prefixes and textual suffixes.

In fact, since there is a generic structure of presentation of structured information of a source, then it is likely that the gene and its clones are surrounded by the same structures, thus these structures can be considered as labels for the gene, these structures will be called structural prefixes and structural suffixes of a gene.

### 4. THE STRUCTURAL PREFIX AND THE STRUCTURAL SUFFIX

In this part we will formalize the new notions: the structural prefix and the structural suffix, which do not exist in the literature. Thanks to these 2 notions we can considerably improve both accuracy and reliability of the gene in order to avoid extracting the irrelevant information.

#### 4.1. ORDERED TREES

**Definition 6:** (ordered tree)

An ordered tree is a rooted tree (naturally oriented) in which all son of each node are ordered. So there is a new order 'horizontal' (left to right) in addition to the vertical order determined from the root.

Example: A tree representing an XML code is an ordered tree, because the order of the child tags of a tag is the order of appearance of these children at the xml code.

#### 4.2. LEFT BROTHER AND RIGHT BROTHER OF A NODE IN AN ORDERED TREE

**Definition 7:** (Left brother and right brother of a node)

Let \( D \) an ordered tree, and let for each node \( x \) of \( D \), \( O(x) \) is the order of \( x \) in the list of his father's sons.

The left brother of a node \( y \) is its brother that has \( O(y) \)-1 as order.
The right brother of a node \( y \) is its brother that has \( O(y)+1 \) as order.

Graphically, the left brother of \( y \) is the brother of \( y \) which is just to the left of \( y \), the right brother of \( y \) is the brother of \( y \) which is just to the right of \( y \).

#### 4.3. STRUCTURAL PREFIX AND STRUCTURAL SUFFIX OF A NODE

Recall of some notions about rooted trees:

Let \( D \) a rooted tree (naturally oriented), and \( x, y, z \) nodes of \( D \).

we say that \( y \) is a descendant of \( x \) if there exists a path from \( x \) to \( y \).
The sub tree rooted at \( z \) is the sub tree of \( D \) composed of \( z \) and its descendants.

**Structural prefix and structural suffix of a node in an xml or xhtml tree:**

**Definition 8:** (structural prefix and structural suffix)

Let \( D \) a xml or xhtml tree and \( x \) a node of \( D \).
The structural prefix of a node \( x \) is the sub tree rooted at the left brother of \( x \).
The structural suffix of a node \( x \) is the sub tree rooted at the right brother of \( x \).

**Note:**
The structural prefix and the textual prefix are two different concepts, in fact, the structural prefix of a node \( <x> \) is a xml structure while the textual prefix of a value \( t_i \) is a textual string.

**Example:**

Let the following xml structure: `"<table> <tr> <td> Name : chara </td> <td> Tel: 0522359950 </td> </tr> </table>"`.
The textual prefix of the value 0522359950 is the textual string "Tel:".
The structural prefix of the second node `<td>` is the xml structure "<td Name: chara / </td>

**4.4. STRUCTURAL PREFIX AND STRUCTURAL SUFFIX OF A NODE OF THE GENE**

**Properties of structural prefix and structural suffix of a node of the gene:**

A node of the gene, and all equivalent of this node in the clones have the same structural prefix. (P1)
A node of the gene, and all equivalents of this node in the clones have the same structural suffix. (P2)

Certainly, the properties (P1) and (P2) sometimes are not checked, but often there are checked.
The valid structural prefixes and valid structural suffixes of the gene:

Definition 9: (valid structural prefix and valid structural suffix)

A valid structural prefix of a gene is a structural prefix of a node of gene that verifies property (P1). A valid structural suffix of a gene is a structural suffix of a node of gene that verifies property (P2).

To optimize and improve the search of clones, we should use the valid structural prefixes and the valid structural suffixes of the gene. These structural suffixes allow to increase the accuracy of the gene by increasing the accuracy of the relative xpaths of the gene.

5. THE PROCESS OF THE "GENE/CLONE WITH STRUCTURAL PREFIXES AND SUFFIXES" APPROACH

We take a document D from a Web data source. The process of construction of extraction rules is as follows:

The HTML document D which the user wishes to extract the structured data is transformed into XHTML document.

After the user enters k instances examples: t' = (t'_1, t'_2, ...., t'_n). t'' = (t''_1, t''_2, ...., t''_n), ...., t'' = (t''_1, t''_2, ...., t''_n), we seek the gene generated by each example instance t that we note g''(j) (the smallest repetitive substructure containing all the values of the instance t') , next, we determine the valid structural prefixes and the valid structural suffixes of the gene g''(j) (gene generated by the first instance) by comparing the structural prefixes and the structural suffixes of the nodes of the gene g''(j) with the structural prefixes and the structural suffixes of the nodes of other genes.

Finally, in order to find all clones of the gene and recover the other instances of the relation to be extracted, we use all structural information of the gene g''(j) and also the valid structural prefixes and the valid structural suffixes of this gene.

Please Note:

- of course we assume that genes have a similar structure that is to say that for each i belonging to [1, ...., n], the relative path of values t'i relatively to genes g''(j) are similar. This assumption is true in most cases because the data blocks of a web page are always presented in the same way.

- if there is a node of the gene g''(j) such that this one and its equivalents in the other genes have the same structural prefix(resp suffix) then this structural prefix(resp suffix) is considered as valid.

6. MATHEMATICAL FORMALISM FOR THE DETERMINATION OF THE VALID STRUCTURAL PREFIXES AND THE VALID STRUCTURAL SUFFIXES

Let t'=(t'_1, t'_2, ...., t'_n), t''=(t''_1, t''_2, ...., t''_n) ...., tk=(tk_1, tk_2, ...., tk_n) k instances.

Here is a mathematical formalism to determine the valid structural prefixes and the valid structural suffixes of the gene generated by the first instance t' (the smallest structure containing all the values of the instance t').

Let for each j belonging to [1, ...., k], g''(j) the gene generated by the instance t' (the smallest structure containing all the values of the instance t').

For each j belonging to [1, ...., k] and i belonging to [1, ...., n], we denote by xpath(i) the relative path of the value t'i relatively to the gene g''(j), that is to say the sequence of nodes to cover from the root node of the gene g''(j) to reach the value t'i.

We put xpath(i)=(x(i)_1, x(i)_2, ..., x(i)k) x(i)_1 is the root node of the gene g''(j), and x(i)k is the node whose the textual content contains the value t'i.

We put for each i belonging to [1, ...., n], j belonging to [1, ...., k] and k' belonging to [1, ...., n (i, j)] , Pk' the structural prefix of x(i)k' and Sk' the structural suffix of x(i)k'.

To determine the valid structural prefixes and the valid structural suffixes of the gene g''(j), we assume that all genes g''(j) have a similar structure, it means that for each i belongs to [1, ...., n] and j belonging to [2, ...., k], n(i, j)= n(i,1), and also for each i belonging to [1, ...., n] and j belonging to [2, ...., k] and k' belonging to [1, ...., n (i, j)],
tagname(x(i)k') = tagname(x(i)1) (that is to say that x(i)k and x(i)1 have the same tag name).

We can see that this assumption is true in most cases because the data blocks of a web page are always presented in the same way.

We compute the structural prefixes and the structural suffixes of the first gene g''(1), and we construct for each i belonging to [1, ...., n] two lists of structures P1,i and S1,i such as P1,i=[P1,i,l, P1,i,2, ... , P1,i,n(i,1)] and S1,i=[S1,i,1, S1,i,2, ... , S1,i,n(i,1)].
P1,i designate the list of structural prefixes of nodes of xpath1 and S1,i designate the list of structural suffixes of nodes of xpath1.

To determine the valid structural prefixes and the valid structural suffixes, we compare the structural prefixes of the nodes of genes g''(j) (j different to 1) with the structural prefixes of the nodes of the gene g''(1), also we compare the structural suffixes of the nodes of genes g''(j) (j different to 1) with the structural suffixes of the nodes of the gene g''(1).

For each j belonging to [2, ...., k], i belonging to [1, ...., n] and k' belonging to [1, ...., n (i, j)], we compare Pk,i with Pk,i,k' and Sk,i with Sk,i,k' .

For each i belonging to [1, ...., n], let IP (i) the stack allowing to recover the indexes of nodes of xpath1 that have a valid structural prefix and IS (i) the stack allowing to recover the indexes of nodes of xpath1 that have a valid structural suffix. We assume that the two stacks are initially empty.

For each i belonging to [1, ...., n], if there is k' belonging to [1, ...., n (i, 1)] such that for each j belonging to [2, ...., k], Pk,i = Pk,i,k' (T1) then the index k' is added at stack IP (i). For each i belonging to [1, ...., n], if there is k' belonging to [1, ...., n (i, 1)] such that for each j belonging to [2, ...., k], Sk,i = Sk,i,k' (T2) then the index k' is added at stack IS (i).

Note:

- Pk,i (resp Sk,i) equal to Pk,i,k' (resp Sk,i,k') means that they have the same names of tags and the same contents of tags (the attributes are ignored when comparing).
we consider that two empty structural prefixes are equal, also we consider that two empty structural suffixes are equal. In fact, if there is a node of the gene such that this one and its equivalents in the other genes, haven’t a structural prefix, then we should exploit this regularity in order to increase the accuracy of the gene. Thus, an empty structural prefix may be valid. Similarly, an empty structural suffix may be valid also.

After verifying tests (T1) and (T2), the valid structural prefixes of the gene \( g^{(1)} \) corresponding to the \( i \)th relevant information are the elements \( P_{i}[k'] \) such that \( k' \) belongs to \( \text{IP}(i) \). The valid structural suffixes of the gene \( g^{(1)} \) corresponding to the \( i \)th relevant information are the elements \( S_{i}[k'] \) such that \( k' \) belongs to \( \text{IS}(i) \).

### 7. VALUATION OF THE "GENE/CLONE" APPROACH AND THE "GENE/CLONE WITH STRUCTURAL PREFIXES AND SUFFIXES" APPROACH ON SEVERAL DATA SOURCES.

We interest in this part to the comparison between the “Gene/clone” approach and the “gene / clone with structural prefixes and suffixes” approach. In fact, we evaluated these two approaches on the same data sources, and we compared the extraction results in order to see if the use of structural prefixes and structural suffixes increased the precision of extraction results.

This valuation was effectuated thanks to a web application that we developed.

We used two evaluation criteria which are known in the domain of web data extraction, namely precision and recall. The rates of recall and precision use the usual formulas of the community of information extraction.

Let \( FN \) the number of instances to extract but they are not extracted, \( FP \) the number of incorrect instances and \( TP \) the number of instances correctly extracted, the recall rate is :\n
\[
R = \frac{TP}{TP + FN}
\]

and the accuracy rate is \( P = \frac{TP}{TP + FP} \).

Here is two table that includes the results obtained by the two approaches for different data sources chosen.

<table>
<thead>
<tr>
<th>source</th>
<th>“gene/clone with structural prefixes and suffixes” approach</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TP</td>
</tr>
<tr>
<td>google.fr</td>
<td>195</td>
</tr>
<tr>
<td>yahoo.fr</td>
<td>200</td>
</tr>
<tr>
<td>livrergratis.com</td>
<td>0</td>
</tr>
<tr>
<td>bibnum.education.fr</td>
<td>86</td>
</tr>
<tr>
<td>avendrealouer.fr</td>
<td>0</td>
</tr>
<tr>
<td>inilibroveritas.net</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 1. Results of the “gene/clone ” approach for the data sources chosen

We note according to this valuation that for some sources, the “Gene / clone” approach gives a zero accuracy rate, while the “Gene / clone with structural prefixes and suffixes” approach gives a high accuracy rate. Thus, we can conclude that for some sources, the “Gene/clone” approach can not extract data correctly, while the “Gene / clone with structural prefixes and suffixes” allows to extract correctly the data.

Therefore, the use of the valid structural prefixes and the valid structural suffixes allowed increase significantly the accuracy of the extraction results, in fact, these valid structural prefixes and valid structural suffixes allow to increase the accuracy on the relative xpaths of the gene, and consequently, optimize the search for clones.

### 8. THE ADVANTAGES OF THE “GENE /CLONE WITH STRUCTURAL PREFIXES AND SUFFIXES” APPROACH

The advantages of the “gene /clone with structural prefixes and suffixes” approach are :

- The precision rate and the recall rate is very high.
- This approach can be applied to any web data source whose the structure of data presentation is repetitive.
- No problem of tedious labeling.
- It is not necessary to analyze all pages of the web source (a single page is sufficient to generate a reliable gene).

### 9. THE INCONVENIANTS OF THE “GENE / CLONE WITH STRUCTURAL PREFIXES AND SUFFIXES” APPROACH

The limitations of the “gene /clone with structural prefixes and suffixes” approach are :

- This Approach requires an intervention from the user , in fact the user must enter at least two examples instances, and must ensure that the examples instances generate the genes that have a similar structure. (In the graphically interface that allows the user to enter of examples instances , there is a button that allows the user to view each gene generated by an instance. So the user can ensure that these genes have a similar structure).
10. GENERATING A META SOCIAL NETWORK BY GATHERING DATA FROM MULTIPLE SOCIAL NETWORKS

10.1. PRINCIPLE OF PROJECT

In order to apply our approach for extracting information from the web, we have developed an application for extracting and gathering data from multiple social networks to generate a meta social network.

More precisely, the idea is to extract and gather the information of companies contained in several social networks to consequently generate a single graphical interface that includes all companies information contained in the social networks used.

In fact, the social networks used does not contain the same base of companies, Thus, thanks to our project we were able to generate an interesting base of companies data, and also display it via a graphic interface.

For this project, we have used some professional social networks that provide reliable information on companies, which will be detailed in the next section.

10.2. THE PROFESSIONAL SOCIAL NETWORKS USED

In our project, we have used these professional social networks: LinkedIn, Xing, Yuppik. You will find below their descriptions.

LinkedIn:

LinkedIn is an online professional social network created in 2003 in Mountain View (California). It allows individuals to build their professional network. It is defined as a knowledge network that facilitates dialogue between professionals.

LinkedIn has several features, one can seek employment, contacts, companies or business opportunities, and be recommended by someone from our network of contacts etc..

Xing:

Xing is an online professional social network that allows each member to compose his own professional network. Xing facilitates communication between professionals.

Xing offers many features: searching a job, companies, people in specific activity sectors, or contacting someone not listed in the network.

Yuppik:

Yuppik is a social network that is addressed especially to students and young graduates. It ensures the relationship between young graduates and enterprises. Unlike traditional sites for job search, here it is the company who hunts the student.

10.3. DESCRIPTION OF THE PART "ENTERPRISES" OF EACH SOCIAL NETWORK

LinkedIn:

The social network LinkedIn allows a member to find companies according to a search keyword. LinkedIn gives the following information of companies: company name, company description, address, number of employees, and number of people registered in linkedin who are interested in the concerned company.

If we consider that the relationship to extract is: (name, description, address, number of employees, number of people who are interested to the company) then an extract of the block containing the gene is represented by the following figure:

In the figure above, the gene is the structure determined by the blue arrows.

From this figure, we see that the relative path li/dl/dd/span allows to access to the address, the number of people who are interested to the company and the string "|", hence the need to use the "gene/clone with structural prefixes and suffixes" approach to correctly extract instances of the relationship (in this case, the "gene / clone" approach can not extract data correctly).

Xing:

The social network Xing allows members to find companies by using a keyword of search or the “Detailed Search” function. By using the detailed search function, you can search for companies according to the following criteria: company name, zip code, city, country, company size, sector of activity. Xing gives the following information of companies: company name, address, country.

If we consider that the relationship to extract is: (name, address, country) then the block containing the gene is represented by the following figure:
In the figure above, the gene is the structure determined by the blue arrows.

We notice from this figure that the relative path li/div/br allows to access to the address and country, hence the obligation to use the “gene/clone with structural prefixes and suffixes” approach to retrieve correctly the instances of the relationship to extract (for this case, only the “gene / clone” approach can not retrieve data correctly).

Yupeek:

The social network Yupeek allows a member to find companies by using a keyword of search. Yupeek gives the following information: company name, sector of activity.

If we consider that the relationship to extract is : (name, sector of activity) then the block containing the gene is represented by the following figure:

In the figure above, the gene is the structure determined by the blue arrows.

In this figure, there is no repetition problem of the relative paths of the gene in the interior of the block containing the gene, and consequently the two approaches can correctly extract the instances of the relationship.

10.4. DESCRIPTION OF THE APPLICATION

The figure below shows the home page of the application, it allows the user to enter a keyword of search, select the social network, and enter the number of example instances for the relationship to extract. To submit the form, the user clicks on the button "SEARCH".

To see in the same page, the data of companies of all social networks used, the user should select the source: "Linkedin+xing+yupeek".

After submitting the form, the application opens a new page (see figure below) divided into two parts:

- The left part displays the web page corresponding to the selected social network and also the keyword of search entered by the user.
- The right part contains an input form that allows the user to enter examples instances with a simple copy / paste of a few values of blocks of results figuring in the left part. This form contains a submit button named "extract data" to display of companies data.

When the user clicks on "extract data" button, the application opens a new page containing the data of companies corresponding to selected social network and the keyword of search.

<table>
<thead>
<tr>
<th>Index</th>
<th>name of the information</th>
<th>value of the information</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>name</td>
<td>HEC Courtrai Business School AG</td>
</tr>
<tr>
<td></td>
<td>description</td>
<td>Labour Law</td>
</tr>
<tr>
<td>2</td>
<td>name</td>
<td>PMMA Industrie SpA</td>
</tr>
<tr>
<td></td>
<td>description</td>
<td>Labour Law</td>
</tr>
<tr>
<td>3</td>
<td>name</td>
<td>3156-2955 employees</td>
</tr>
<tr>
<td></td>
<td>description</td>
<td>Labour Law</td>
</tr>
</tbody>
</table>

Figure 4. Home page of the application

Figure 5. page of introduction of examples instances

Figure 6. page containing data of companies
If the user selects the source "Linkedin+xing+yupeek", the application opens a new page divided into two parts:
- The left part displays the web pages corresponding for each social network and the keyword of search entered by the user.
- The right part containing an input form allowing to enter examples instance for each social network. It also contains a submit button named "extract data" that display the data of companies for each social network corresponding to the keyword of search entered by the user.

NB:

We have used "gene/clone with structural prefixes and suffixes" approach to extract the data of companies because the "gene/clone" approach can not correctly extract the data of companies of these social networks.

This application has shown the benefits of using the "gene / clone with structural prefixes and suffixes" approach in terms in accuracy of results extraction.

11. CONCLUSION

For some sources, the "gene / clone" method remains insufficient to extract correctly the data, while "gene/clone with structural prefixes and suffixes" approach allows to extract correctly the data.

The high precision of the method of "gene/clone with structural prefixes and suffixes" is due to use a valid structural prefixes and valid structural suffixes of the gene; these structures increase the accuracy on the relative paths of the gene, consequently they improve the detection of clones and avoid extracting for irrelevant information.

References
