Mining Conceptual Graphs for Knowledge Acquisition*

Milagros Fernández
Dept. of Computer Science
University of Corunna
Campus de Elviña s/n
15174 A Coruña, Spain
mfgavilanes@udc.es

Eric de la Clergerie
Inria
Domaine de Voluceau,
Rocquencourt, B.P. 105
78153 Le Chesnay Cedex,
France
Eric.De_La_Clergerie@inria.fr

Manuel Vilares
Dept. of Computer Science
University of Vigo
Campus As Lagoas, s/n
32004 Ourense, Spain
vilares@uvigo.es

ABSTRACT

This work addresses the use of computational linguistic analysis techniques for conceptual graphs learning from unstructured texts. A technique including both content mining and interpretation, as well as clustering and data cleaning, is introduced. Our proposal exploits sentence structure in order to generate concept hypothese, rank them according to plausibility and select the most credible ones. It enables the knowledge acquisition task to be performed without supervision, minimizing the possibility of failing to retrieve information contained in the document, in order to extract non-taxonomic relations.

Categories and Subject Descriptors

I.2.6 [**Learning**]: Knowledge acquisition; I.2.7 [**Natural Language Processing**]: Language parsing and understanding

General Terms

Management

Keywords

Classification,Clustering, Knowledge synthesis and visualization, Text Mining

1. INTRODUCTION

Even though research on the conceptual querying concept [10] dates from the early days of *information retrieval* (IR) research, it is surprising that, nowadays, most of practical IR systems are still based on the classic *bag of words* proposal. In effect, given that text retrieval [11] is a *natural language processing* (NLP) task, the most sensible thing would be to incorporate some of the user's knowledge and reasoning capabilities in order to improve precision in query processing.

Retrieval at the conceptual level should contribute to overcoming these limitations, leading us to more complex IR approaches. However, we should first consider an automated tool for identifying concepts from raw text, which implies having efficient techniques available for dealing with both the inherent ambiguity and flexibility of the natural language.

With things as they are, the automatic construction of practical structures from text has become an active research topic; and the reduction of both the time and effort in their development process, a tremendous need. Also, given that most of human knowledge is available in textual format, the consideration of *natural language processing* (NLP) techniques to extract the latent semantics from texts seems to be an adequate starting point to cope with the problem.

To deal with the exploitation of the linguistic structure in a text without requiring predefined knowledge of the specific domain analyzed, most authors consider a combination of robust parsing, allowing semantic relations to emerge from the text, and some kind of statistical and/or heuristic strategies in order to select the most relevant of these. On the robust parsing side, it is often argued that complex text processing is impractical on real corpus [7]. So, a popular technique is *association rule learning*, applied to retrieving term associations satisfying a minimum level of support and confidence through linguistic patterns. Formally based on a deterministic finite automaton architecture, it is the simplest and computationally most efficient strategy, which allows it to deal with very large text collections. However, it first results in excessively general parsers, which can lead to a failure to identify less commonly found grammatical structures.Moreover, its deterministic condition can mean that the system discards useful interpretations, when all the available information should be translated and considered later in a specific filtering-out task, once complementary data is available and ambiguities could effectively be solved.

Concerning the statistical/heuristic task, these methods are often applied as a complement of the parsing one with a semantic clustering purpose. The goal is to simplify the initial set of semantic links proposed by the parse, eliminating ambiguous interpretations as far as possible. Given that these techniques are based on a distributional analysis intended to be applied on large corpora, time and space

*This work was partially supported by the Spanish Government from research project HUM2007-66607-C04-02, and by the Autonomous Government of Galicia from research projects 05PXIC30501PN, 07SIN005206PR, the Galician Network for NLP and IR, and 'axuda para la consolidación e estruturación de grupos de investigación'.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
iNEWS'08, October 30, 2008, Napa Valley, California, USA.
Copyright 2008 ACM 978-1-60558-253-5/08/10 ...$5.00.

$$P(\text{denticulées:adj})_{\text{local}(0)} = \frac{\#\text{deriv-node(denticulées:adj)}}{\#\text{deriv-cluster}} \quad (1)$$

$$P(\text{denticulées:adj})_{\text{global}(n+1)} = \frac{\Sigma_{i=1}^{n} P(\text{denticulées:adj})_{\text{local(i)}}}{\#\text{occ-denticulées}} \quad (2)$$

$$P(\text{denticulées:adj})_{\text{local}(n+1)} = \frac{P(\text{denticulées:adj})_{\text{local(n)}} P(\text{denticulées:adj})_{\text{global}(n+1)}}{\Sigma_X \quad P(\text{denticulées:X})_{\text{local(n)}} P(\text{denticulées:X})_{\text{global}(n+1)}} \quad (3)$$

Table 1: Lexical categories for the word *"denticulées"*

complexity become essential factors in their design. So, although the final system should be able to compute the frequency of n-grams of words for arbitrary values of n, most authors choose to work with bigrams using association measures known from collocation discovery, such as χ^2, pointwise mutual information or log-likelihood-ratios. In addition to this primary limitation on the value for n, these formula may also provide different or even inappropriate estimations. In effect, mutual information tends to overestimate low-frequency data while log-likelihood-ratios and χ^2 assume we are dealing with normal distributions, which is not realistic when working on texts, in which rare phenomena are common.

An alternative consists of considering a statistical model allowing an *a priori* unlimited number of states defining the probabilistic dependency. Here the classic reference is the *hidden Markov model* (HMM) that does not seem to give complete satisfaction to our requirements either. So, although HMMs can apply on arbitrary n-grams, the value of n is fixed. This firstly implies that we are assuming *stationary time*[1] and *limited history*[2] hypothesis which, for example, would not enable us to deal with long distance semantic dependencies nor recursive structures. Although we could get round this problem by considering a sufficiently high value for n, this would involve prohibitive time and space performances.

In this context, our contribution can be summarized as a proposal whose aim is to produce practical understandable results by allowing the unsupervised integration of background knowledge from complex document representations, promoting the use of *conceptual graphs* (CGs) produced automatically from text, exploiting the linguistic information available in the corpus and a sophisticated grammatical formalism, extracting the latent semantics. More in detail, we introduce a text mining strategy where primary knowledge acquisition is performed through a robust parser working on a *tree-adjoining grammar* (TAG) generated from a source *meta-grammar* (MG) [5]. Working on such a mildly context-sensitive formalism we significantly increase descriptive power in relation to *context-free grammars* (CFGs), while time and space bounds are polynomial. On the other hand, the acquisition knowledge process is not considered as deterministic, but allows us to deal with different interpretations simultaneously, integrating all viable alternatives in the final knowledge representation, in such a way that documents are represented by a structure of terms enriched by relations.

For clustering purposes, we adapt an iterative algorithm inspired by an error-mining strategy [9] developed to locate and diagnose parse and lexical errors in NLP applications. This technique enables the retrieval from a large corpora of missing, incorrect or incomplete linguistic descriptions using the frequency of n-grams of words for arbitrary values of n.

[1]probabilistic dependencies value does not change with time.
[2]probabilistic dependencies are limited to n states.

2. THE RUNNING CORPUS

We introduce our proposal from a botanic corpus describing West African flora. We concentrate on the work *"Flore du Cameroun"*, published between 1963 and 2001, which is composed of about 40 volumes in French, each volume running to about 300 pages, organized as a sequence of sections, each one dedicated to one species and following a systematic structural schema. So, sections include a descriptive part enumerating morphological aspects such as color, texture or form. This implies the presence of noun phrases, adjectives and also adverbs to express frequency and intensity, and named entities to denote dimensions.

The corpus[3] describes concepts that are related both taxonomically, for example hypernymy or `"is a"` relations, and non-taxonomically. The collection also possesses a vocabulary that is shared by most text based on this matter and is of sufficient size for our purposes.

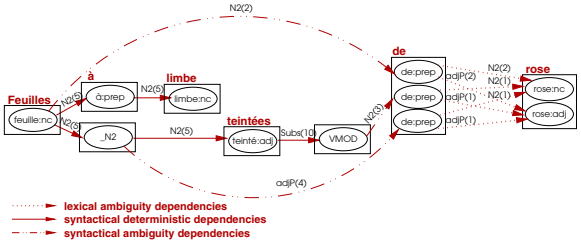

Figure 1: Parse dependencies

The present paper forms part of BIOTIM [8], a research initiative on the integral management of botanic corpus including conceptual acquisition and text mining tasks. Here, we disregard initial phases, related to the transfer from textual to electronic format [9] and also the capture of the logical structure of the text.Our linguistic starting point will be a grammar of large coverage for French and the tagged corpus. [4]

3. THE PARSING FRAME

We choose to work with TAGs [6], a mildly context-sensitive grammatical formalism that has given rise to a lot of interest in the modeling of syntax in NLP. Basically, TAGs are

[3]provided by the French Institute of Research for Cooperative Development.
[4]http://mgkit.gforge.inria.fr/

Properties	Lemmas
color	verdâtre, violacé, noirâtre, violet, jaunâtre, orange, roux, rose
form	obconique, oblancéolé, oblong, bifolié, crateriforme, punctiforme, périgone, concave, oblongoïde, ovoïde
size	moyen, petit, double, épais, inégal, entier, longue
texture	hispide, bifide, globuleux, coriace, velutineux, gélatineux, barbu
position	antérieur, dessus, voisin, seul, latéral, transversal
others	dur, bifide, frais, fréquent, jeune

Table 2: File of properties

somewhat similar to classic CFGs, but the elementary unit of rewriting is the tree rather than the symbol, allowing *extended domains of locality* (EDLs) to be specified as compared to those over which lexical constraints can be stated.

3.1 Mildly context-sensitive parsing

Any grammar formalism defines a domain of locality, that is, a domain over which various dependencies, syntactic and semantic, can be specified. This issue is related to the use of constrained systems adequate for modeling various aspects of language. In this context, the principle of EDL means that TAGs possess certain properties that make them more powerful than CFGs in terms of generative capacity. So, it allows constraints to be defined in more than one level of the parsing tree as compared to context-free rules.

Altogether, these properties lead us to conjecture that TAGs are powerful enough to model natural language while remaining efficiently parseable, but in order to fully exploit them we need an adequate operational framework. Our choice is DyALog [4], a parsing environment for a variety of grammatical formalisms, including TAGs, that returns total or partial parses. Our aim is to avoid the elimination of any parsing branch until we are sure it is not used for knowledge acquisition. In the particular case of TAGs, the system implements a parse scheme [1] verifying the VPP, which assures the best time behavior.

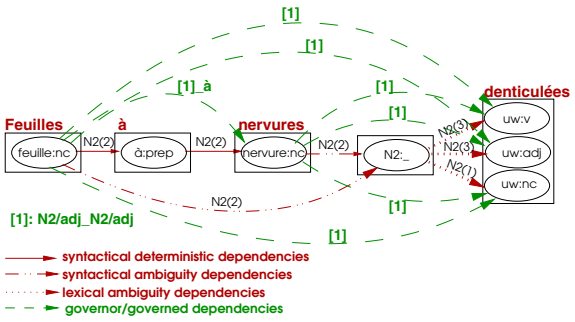

Figure 2: Another example of parse dependencies

The introduction of a high degree of abstraction in the design of the analyzer is achieved on the basis of the MG concept [5], by involving elementary constraints re-grouped in classes, these themselves inserted in a hierarchy of multiple heritage. This allows descriptions to be progressively refined, which is of particular interest when we are describing complex linguistic behavior.

3.2 Parse dependencies

The parse is resumed in a *graph of syntactic dependencies*, as is shown in Fig. 1 for the sentence *"feuilles à limbe teintées de rose"* (`"rose-tinted laminar leaves"`). Here, arrows represent binary dependencies between nodes through some syntactic construction. The parse labels each *node*, represented by an ellipse, with the tag of the corresponding lexical form, including its lemma. Rectangular shapes represent *clusters*, that is, structures referring to a position in the input string and all the possible nodes assigned by the parse at that position. So, we introduce ambiguities from both lexical and syntactic points of view. The former corresponds to clusters containing different nodes and are indicated by dotted dependencies, while syntactic ones correspond to different dependencies marked by broken dotted lines and relating to the same node.

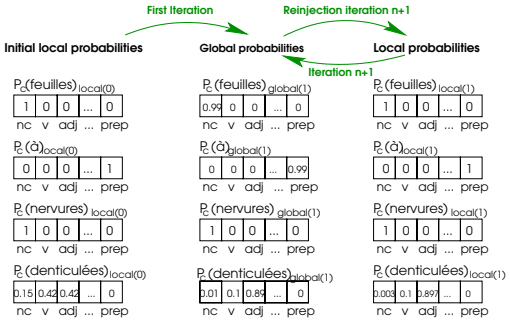

Figure 3: Computing lexical categories

This structure constitutes the starting point for detecting related knowledge and composes an initial *graph of governor/ governed dependencies*, reflecting the corresponding syntactic relationship between the nodes involved. Formally, the head of a syntagm governs its modifiers, as is shown in Fig. 2 by broken lines going from the governor node to the governed one, and labeled by functors. We shall later transform these dependencies into semantic ones.

3.2.1 Lexical ambiguities

Formally, the morpho-syntactic phase consists of a pipeline named Sxpipe [8] that concatenates a number of different tasks such as chunking, entity recognition and tagging.

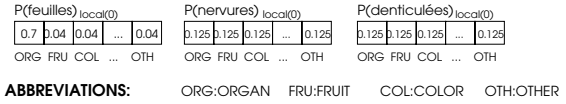

Figure 4: List of semantic weights

Tagging is often a non-deterministic and even incomplete task, especially when dealing with an encyclopedic corpus with a high degree of unknown words, that is, words whose

Word	Position	Class	Word	Position	Class	Word	Position	Class
teinte	[2]	color	couleur	[2]	color	tache	[2]	color
teinté	[2]	color	texture	[2]	texture	position	[2]	position
taille	[1,2]	size	diamètre	[1]	size	épaisseur	[1]	size
longueur	[1]	size	hauteur	[1]	size	largeur	[1]	size
altitude	[2]	size	atteindre	[2]	size	dépassant	[2]	size
atteindre	[1]	organ/fruit	forme	[2]	form	bord	[1]	organ/fruit

Table 3: File of linguistic markers

lemma is not included in the lexicon of the tagger and whose corresponding tag can only be suggested through the parser from the grammar considered. This is shown in Fig. 2, where *"denticulées"* (`"dentate"`) is labeled as an unknown word (`uw`) with three possible associated lexical categories: verb (`v`), adjective (`adj`) and common noun (`nc`). In order to avoid discarding useful interpretations, we should translate these ambiguities, which we cannot solve at a lexical level, to the syntactic phase.

This is the case of the sentence *"feuilles à limbe teintées de rose"*, that we could interpret as `"rose's tinted laminar leaves"`, as `"rose-tinted laminar leaves"` or as `"tinted laminar rose leaves"`. In the first case, *"rose"* would be a noun related to *"feuilles"* (`"leaves"`), while in the other ones it would be an adjective related to *"teintées* (`"tinted"`), as is shown in Fig. 1.

3.2.2 Syntactic ambiguities

Parsing in NLP is an incomplete task and, therefore, a source of ambiguities because it usually deals with shallow/partial strategies providing a lightweight analysis focused on identifying dependencies between nodes that are more or less close together in the text, such as noun sentences, as in *"feuilles à nervures denticulées"*, that we could locally translate in two ways: `"leaves with dentate veins"` or, alternatively, `"dentate leaves with veins"`. It here becomes impossible to establish if *"denticulées"* (`"dentate"`) relates to *"feuilles"* (`"leaves"`) or to *"nervures"* (`"veins"`), as shown in Fig. 2.

In terms of dependencies, the ambiguities, that are caused by local non-determinism, can be translated into a governor node having more than one governed node. As a consequence, to solve these ambiguities involves applying a simple syntactic constraint, namely, that a governed node should have only one governor. So, for example, in the sentence of Fig. 2, *"denticulées"* (`"dentate"`) is governed by *"feuilles"* (`"leaves"`), but also by *"nervures"* (`"veins"`) and, in consequence, we should give priority to one of these dependencies.

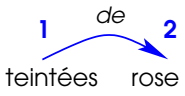

Figure 5: An example of a marked structure

No other topological restrictions are considered and, in consequence, a governor node can have more than one governed one; as in the second interpretation of Fig. 2 (`"dentate leaves with veins"`), where *"feuilles"* (`"leaves"`) is the governor for *"nervures"* (`"veins"`) and *"denticulées"* (`"dentate"`). Also, one node could be governor and governed at the same time, as in the first interpretation of Fig. 2 where *"nervures"* (`"veins"`) is the governor of *"denticulées"* (`"dentate"`), but is also governed by *"feuilles"* (`"leaves"`).

Given that ambiguous sentences require a greater use of increasing constraints to solve them, the idea behind the strategy we consider applying consists in identifying the concept that best matches the area and analyzing it, as well as its relationships with the context. This implies exploring mining techniques for discovering hidden knowledge.

4. KNOWLEDGE ACQUISITION

Once these primary syntactic dependencies have been established, probably including a number of lexical and syntactic ambiguities, our goal is to effectively extract the meaning of the corpus. This firstly implies initializing the set of classes to be considered and the dependencies between them. Later, the process will continue by compiling additional information and ranking these dependencies in order to detect those that are less plausible. Given that we are assuming our corpus is large enough, we should be able to recover this information by exploring it progressively in depth. That is, to solve the problem we only need the information we are looking for, which leads us to consider an iterative learning process in order to attain our goal.

We can illustrate this on our running corpus. So, the lexical ambiguity described in Fig. 1 should usually be decided in favor of the first alternative (`"rose's tinted laminar leaves"`), because most of us have the intuitive certainty that plants with rose colored leaves do not exist. However, this is not the case here since the rose is not a botanic species of the west Africa flora and the corpus never talks about this plant. In fact, the correct alternative is the second one (`"rose-tinted laminar leaves"`), where the word (`"tinted"`) indicates that what goes after is a color.

However, the process we are going to describe does not necessarily imply the determination of the conceptual graph. In effect, even assuming that we are working on a large corpus, we cannot be sure that a given dependency can be useless on the basis of its low weight on the graph. Perhaps the problem simply consists of the fact that some aspect of the knowledge domain is not yet fully developed in the corpus. This is a realistic hypothesis in dealing, for example, with our running botanic document collection. So, although the third alternative in Fig. 1 (`"tinted laminar rose leaves"`) is highly improbable, we should not eliminate it, but simply consider this interpretation having a low probability since we cannot discard that in the future a new variety will be unequivocally described as having rose leaves.

In this sense, our approach is inspired by an error-mining proposal originally designed to identify missing and erroneous information in parsing systems [9], combining two complementary iterative processes. For a given iteration, the first one computes, for each governor/governed pair in a

$$P(\text{feuilles:uc}, [\text{à-1}], \text{nervures:uc})_{\text{local(0)}} = \frac{\begin{array}{c} P_c(\text{feuilles:uc})_{\text{local}}\ P_c(\text{nervures:uc})_{\text{local}} \\ P_{\text{dep ini}}(\text{feuilles:uc}, [\text{à-1}], \text{nervures:uc}) \end{array}}{\Sigma_{X,Y,Z} P_c(\text{Z:X})_{\text{local}}\ P_c(\text{nervures:Y})_{\text{local}}} \quad (4)$$

$$P(\text{feuilles:uc}, [\text{à-1}], \text{nervures:uc})_{\text{global(n+1)}} = \frac{\Sigma_{i=1}^{n} P(\text{feuilles:uc}, [\text{à-1}], \text{nervures:uc})_{\text{local(i)}}}{\#\text{dep}_{\text{local(n)}}} \quad (5)$$

$$P(\text{feuilles:uc}, [\text{à-1}], \text{nervures:uc})_{\text{local(n+1)}} = \frac{\begin{array}{c} P(\text{feuilles:uc}, [\text{à-1}], \text{nervures:uc})_{\text{local(n)}} \\ P(\text{feuilles:uc}, [\text{à-1}], \text{nervures:uc})_{\text{global(n+1)}} \end{array}}{\Sigma_{X,Y,Z,T} \begin{array}{l} P(\text{Z:X}, T, \text{nervures:Y})_{\text{local(n)}} \\ P(\text{Z:X}, T, \text{nervures:Y})_{\text{global(n+1)}} \end{array}} \quad (6)$$

Table 4: Extraction of dependencies for *"feuilles à nervures denticulées"*

sentence, the probability of the corresponding dependency. The second process computes, from the former, the most probable semantic class to be assigned to terms involved in the dependency. So, in each iteration we look for both semantic and syntactic disambiguation, each benefiting from the other. A fixed point assures the convergence [9].

4.1 Starting the process

The first step consists in estimating, from the graph of syntactic dependencies, how a set of initial values for classes and instances can be established through the set of nodes in order to begin the iterative learning process.

Taking as our reference a list of identifiers provided by the programmer for naming the classes we are going to consider, our goal is to extract from the corpus a minimal set of nodes associated to each one. For example, in our running botanic corpus, entities could be distinguished in this list as `organ` or `fruit`, but also as properties of the type `color`, `form`, `size`, `texture` or `position`. The programmer attaches to each class a sequence of initial lemmas whose some values can be seen in Tables 2.

In future, no distinction will be considered between the terms weight, probability and preference, assuming they refer to the same statistical concept. At this point, a weight is assigned to nodes in clusters relating them with each lexical category initially associated by the parse. In order to solve ambiguities at this level, we compute them as shown in Table 1, taking as our example the word *"denticulées"*:

(1). To begin with, we compute the initial local probability for each tagged lemma of a node in a cluster, which is a simple ratio between the number of parses involving this node (`#deriv-node`) and the total number of these involving the cluster (`#deriv-cluster`). In our example, the number of parses involving the node *"denticulées"*, whose tag is an adjective (`uw:adj`), is (`3`), as shown with the label (`N2`) of the dependency (`N2(3)`). There are seven parses involving the corresponding cluster (`(N2(3), N2(3), N2(1)`).

(2). We re-introduce the local probabilities into the whole corpus in order to re-compute the weight of all tagged lemmas, after which we then globally estimate the most probable ones. Normalization is given by the number of occurrences of the lemma (`#occ-denticulées`), possible on different nodes.

(3). The local value in the new iteration should take into account both the global preferences and the local injection of these in the cluster, reinforcing the local probability. Normalization is given by local and global weights for the lemma, involving all possible tags (`X`) associated to the cluster considered.

We illustrate in Fig. 3 this calculus for our example in three columns, one for each step introduced. An element in these columns is a *property list of tag weights* including all tagging alternatives for the corresponding lexical form. More in detail the left-most column is the estimation of the initial local probabilities. The center one refers to the computation for the global probability, and the right-most column represents the re-injection of this in the next iteration. As we can see, in the case of *"feuilles"* the initial probability is the same as the result obtained after the first iteration because, in this sentence, *"feuilles"* has only one possible tag.

The system then associates to local occurrences of each node in the graph of governor/governed dependencies a *property list of semantic weights* reflecting the probability of the governor word being an instance of a class and the governed word to be an instance of another class, as shown in Fig. 4. Positions in one of these lists refer to probabilities for class assignment, whose sum must be equal to one, and their computation relates to the detection of particular syntactic and/or lexical patterns involving nodes with lemmas liable to be one of those instances. When this last condition is satisfied, we assign a fixed weight[5] to the corresponding entry in the property list and we equitably distribute preferences between the rest of classes in that list. For example, in the case of the word *"feuilles"*, that fixed value is put in the first position of the list of semantic weights because it is considered to be an `organ`.

With regard to the pattern recognition, we should take into account the particular characteristics of each document collection. So, in the case of our running corpus, we can assume we are dealing with a descriptive text whose kernel is composed of definitions and, therefore, dependencies related to syntactic patterns should revolve around parse structures involving nouns and/or adjectives.

On the lexical side, we take advantage of linguistic markers in order to set semantic knowledge in a context. These relations involve more explicit physical information, such as *"en forme de X"* (`"in form of X"`) or *"de couleur X"* (`"of color X"`). So, they presumably provide the most reliable information on both classes and dependencies, concentrating the vocabulary around them. We accordingly refer to the nodes and dependencies so located as *pivot nodes* and *strong dependencies*. The result serves to acquire simple

[5]in our case, this weight is 0'7.

$$P(\text{feuilles:uc:org, [à-1], nervures:uc:org})_{\text{local(0)}} = \frac{\begin{matrix}P(\text{feuilles:uc, [à-1], nervures:uc})_{\text{local(0)}} \\ P(\text{feuilles:uc:org})_{\text{local(0)}} \\ P(\text{nervures:uc:org})_{\text{local(0)}}\end{matrix}}{\Sigma_{X,Y} P(\text{feuilles:uc:X})_{\text{local(0)}}\ P(\text{nervures:uc:Y})_{\text{local(0)}}} \quad (7)$$

$$P(\text{feuilles:uc:org, [à-1]}, X)_{\text{global(n+1)}} = \frac{\Sigma_X P(\text{feuilles:uc:org,[à-1]},X)_{\text{local(n)}}}{\#\text{dep}_{\text{local(n)}}(\text{feuilles})} \quad (8.1)$$

$$P(Y, \text{[à-1], nervures:uc:org})_{\text{global(n+1)}} = \frac{\Sigma_Y P(Y,\text{[à-1], nervures:uc:org})_{\text{local(n)}}}{\#\text{dep}_{\text{local(n)}}(\text{nervures})} \quad (8.2) \quad (8)$$

$$P(\text{feuilles:uc:org, [à-1], nervures:uc:org})_{\text{global(n+1)}} = \begin{matrix}P(\text{feuilles:uc:org, [à-1]}, X)_{\text{global(n+1)}} \\ P(Y, \text{[à-1], nervures:uc:org})_{\text{global(n+1)}}\end{matrix} \quad (8.3)$$

$$P(\text{feuilles:uc:org, [à-1], nervures:uc:org})_{\text{local(n+1)}} = \frac{\begin{matrix}P(\text{feuilles:uc:org, [à-1], nervures:uc:org})_{\text{local(n)}} \\ P(\text{feuilles:uc:org, [à-1], nervures:uc:org})_{\text{global(n+1)}}\end{matrix}}{\Sigma_{X,Y} \quad \begin{matrix}P(\text{feuilles:uc:X, [à-1], nervures:uc:Y})_{\text{local(n)}} \\ P(\text{feuilles:uc:X, [à-1], nervures:uc:Y})_{\text{global(n+1)}}\end{matrix}} \quad (9)$$

Table 5: Extraction of classes for *"feuilles à nervures denticulées"*

concepts such as the value of the properties referred, or to detect enumerations that can propagate some of these values. Formally, as is shown in Table 3, we consider triples to represent markers, where the first element is the lemma playing the role of linguistic marker and the second indicates the position on the syntagm for the lemma marked by the first one. The last element is the class that will be considered as being the most probable for that marked structure.

This is illustrated that in Fig. 5 for the syntagm *"teintées de rose"* (`"rose-tinted"`), taken from the sentence *"feuilles à limbe teintées de rose"* (`"rose-tinted laminar leaves"`), where the presence of the marker *"teinté"* (`"tinted"`) indicates that the lemma *"rose"* (`"rose"`) can be embedded in the class *color*, as indicated in Table 3. As is shown in Fig. 5, the number 2 represents the position of the lemma once semantic dependencies have been extracted, as can be seen in Fig. 1, following a given syntactic pattern. Thus, *"teintées"* is considered to be in the first position, while *"rose"* is placed in the second one and both are related through a dependency labeled by the preposition *"de"*.

Once the initial process has finished, all the property lists of semantic weights associated to nodes in the graph of syntactic dependencies have been initialized, and the first assumptions on semantic dependencies between classes can be made by extending the corresponding syntactic dependencies involving nodes into these classes. The system is ready to begin with the iterative learning task, which we shall illustrate on the syntactic dependency labeled `[à_1]` relating *"feuilles"* (`"leaves"`) and *"nervures"* (`"veins"`) in Fig. 2.

4.2 Ranking of dependencies

In dealing with the ranking of dependencies, the sequence of steps to be applied by the learning process is shown in Table 4. Our aim is to associate to a probability each dependency in the graph of syntactic dependencies, denoted by `P(word1:c1,[label],word2:c2)`; with `word1` being the governor node, `c1` the lexical category of the `word1`, `label` the tag of the dependency, `word2` the governed node and `c2` the lexical category of the `word2`. More formally, we have that:

(4). We compute the local probability of the dependency in each sentence. To start the process, first tag assumptions ($\mathtt{P_C}$) are provided by the error-mining algorithm [9], whose process was described in Fig. 3. We also take into account the initial probability for the dependency considered ($\mathtt{P_{dep\ ini}}$), a simple ratio on all possible dependencies involving the nodes concerned. Normalization is given by the choice for the possible lexical categories, denoted by `X` and `Y`, involving each of the clusters considered as governor, expressed by `Z`.

(5). We re-introduce the local probabilities into the whole corpus in order to re-compute the weights of all possible dependencies, after which we then globally estimate the most probable ones. Normalization is given by the number of dependencies connecting the nodes considered (`#dep`).

(6). The local value in the new iteration should take into account both the global preferences and the local injection of them in the sentences, reinforcing the local probabilities. Normalization is given by previous local and global weights for the dependency, whose label is represented by `T`, involving all possible lexical categories, denoted by `X` and `Y`, associated to each of the clusters considered as governor, represented by `Z`.

4.3 Semantic class assignment

Concerning this, the sequence of steps is shown in Table 5, illustrating the computation of the probability that *"feuilles"* (`"leaves"`) and *"nervures"* (`"veins"`) are both organs, taking again the dependency labeled `[à_1]` in Fig. 2:

(7). In each sentence, we compute the local probability of this dependency if *"feuilles"* (`"leaves"`) and *"nervures"* (`"veins"`) are both organs (`org`). We start from the local weight computed in Table 4, and the initial preferences of the nodes involved in relation to class assignment as in Fig. 4. Normalization is given by the probabilities for the possible classes involving each one of the nodes considered represented by `X` and `Y`.

(8). We then calculate this preference at global level, by re-introducing it into the whole corpus in order to recompute the weights of all the possible classes in the sentence. In order to obtain it, we first compute the probability in the whole corpus (8.1 and 8.2) for each

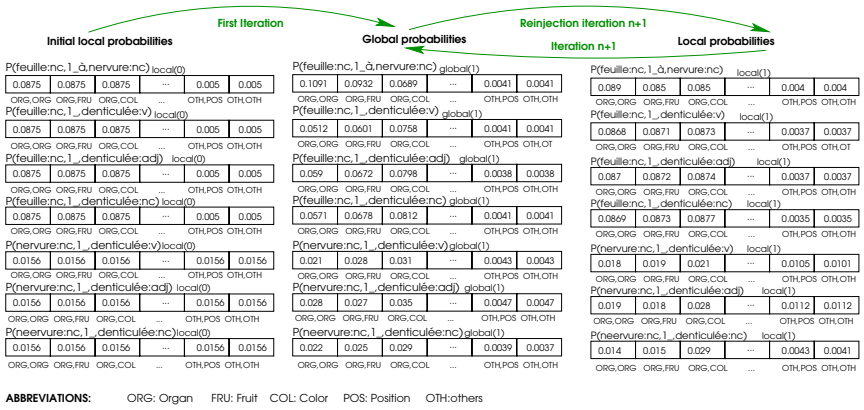

Figure 6: Computing semantic categories

node and semantic class, disregarding the right and left context, represented by `X` and `Y`. The final probability (8.3) is a combination of the two previous ones.

(9). After each iteration, we re-inject the previous global weight to obtain a new local one, by reinforcing the local probabilities. Normalization is done by the addition of the preferences corresponding to the nodes and classes involved in the dependency, for all the possible semantic classes considered.

We illustrate in Fig. 6 this calculus for our example in three columns, one for each step. An element in these columns is a *property list of semantic weights*. More in detail, the first column is the estimation of the initial local probabilities. So, the word *"feuilles"* (`nc`), related to the word *"nervures"* (`nc`) through a dependency labeled as `[1_à]`, has a property list where the first entry refers to the probability that *"feuilles"* and *"nervures"* could be an `organ`. The second column refers to the computation for the global probability, and the last represents the re-injection of this in the next iteration.

5. EXPERIMENTAL RESULTS

We now describe some preliminary tests on our proposal. A major drawback here, derived of the range of the corpus and the novelty and unusual of its content, is the difficulty to develop a systematic work of validation for these results, that must be done by a group of experts on west Africa flora. So, the urgency to estimate the viability of the proposal, takes us to consider a representative sampling we consider it is sufficient to provide guiding and reliable data on efficiency. We formally justify this on the uniformity of both the syntactic structure commented as well as the lexical distribution, that we show in Fig. 7. In this way, we have randomly chosen a collection of samples, each one composed by 100 sentences taken from our running corpus. In order to facilitate understanding, we focus on three of these samples, whose behavior summarizes the results obtained in all the collection.

Relating to the data compiled, when measuring the quality of an automatically created conceptual structure, the typical measures are Recall (10), Precision (11) and F-measure (12). Intuitively, *Recall* shows how much of the existing knowledge is extracted, and it is computed by

$$\text{Recall} = \frac{\#\text{correctly selected entities}}{\#\text{domain entities}} \qquad (10)$$

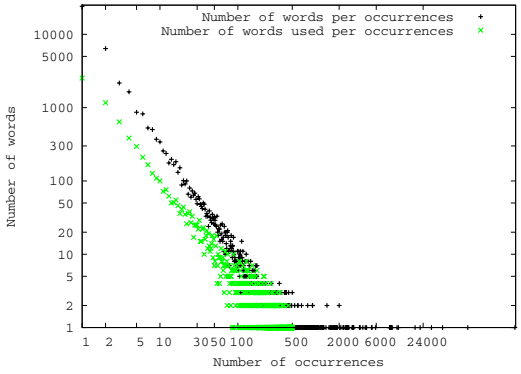

Figure 7: Word distribution in the corpus

while *Precision* specifies to which extent the knowledge is extracted correctly, and it is given by

$$\text{Precision} = \frac{\#\text{correctly selected entities}}{\#\text{total selected entities}} \qquad (11)$$

Related to the *F-measure*, it provides the weighted harmonic mean of precision and recall, summarizing the global performance of the selection process. It is computed as follows

$$\text{F-measure} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \qquad (12)$$

We concentrate our attention on this last measure. Whatever is the case, these tests are performed in function of the number of iteration passes, once fixed two thresholds:

- The probability (`% success`) that a node be embedded in a class at a given moment of the process, in order to evaluate the effect of lexical non-determinism. We consider two values for testing: 20% and 90%. The former refers to a testing frame highly tolerant to this phenomena, on the contrary of the second case.

- The probability (`prob`) of a dependency when it refers to nodes shared between parses, which allows us to estimate the capability to discriminate between different interpretations, illustrating the impact of syntactic ambiguities on the learning task. We consider as values for this threshold 0'2 and 1. The former value groups dependencies on which the learning process should to

continue. The second one refers to dependencies fully identified by the algorithm, that is, considered as deterministic and, therefore, there is no reason to continue the learning process on them.

Between all the possible combinations for this set of thresholds, we focus on two extreme cases, as shown in Fig. 8. The first one illustrates a maximum level of non-determinism on both lexical and syntactic points of view. The second one compiles results for low lexical and syntactic ambiguity. Relating to the samples of sentences considered, which are divided in two groups, we baptize them as *Sample 1*, *Sample 2* and *Sample 3* in these figures for each case.

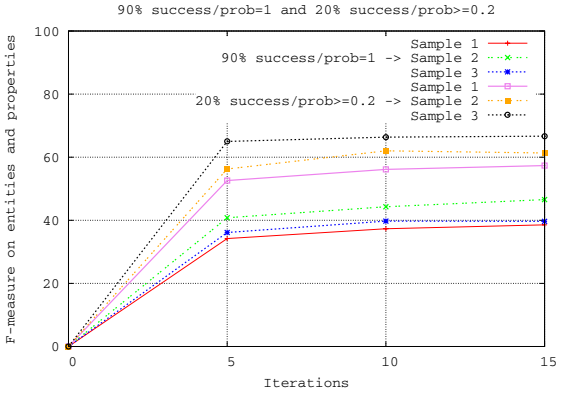

Figure 8: Maximum and minimum non-determinism

A simple analysis puts into evidence a similar qualitative behavior in the evolution of the knowledge acquisition task, with a high speed of learning that allows the process to become stabilized after only five iterations. Just as was hoped, the increasing value for the precision indexes strongly impacts the F-measure for a deterministic context as that considered in Fig. 8, for the first three samples. So, the number of detected instances of classes in this case results to be much poor in comparison with the more flexible context represented in the second three samples. In effect, intuitively, more confidence is the information relative to dependencies and node identification, and more strict are the constraints on the knowledge acquisition process. A similar reasoning can be applied in relation to the speed observing in the learning process, directly associated to the tangent of each graph, which is more steep in the second three samples.

6. CONCLUSIONS

The increasing amount of information in textual format currently available on-line is changing the way of building knowledge-based systems. On the one hand, it is inconceivable to capture it manually and, on the other, it is not possible to directly consider automatic management facilities, which has created a growing need for effective concept learning strategies. Without this kind of tools, access to relevant information runs the risk of become a frustrating and inefficient task. Our claim is that it is possible to consider the unsupervised generation of practical conceptual graphs from an unstructured corpus of sufficient size.

Our proposal attempts to reconcile quantitative and qualitative aspects on both knowledge acquisition and clustering phases. We seek to dynamically compile local, and also global, context information during the learning process. In contrast to previous works that entrust lightweight deterministic parsers with the primary knowledge acquisition task, assuming that high redundancy over a large corpus will enable mis-interpretation phenomena to be overcome; we consider an efficient non-deterministic analyzer that nips the problem in the bud. Once the corpus has been parsed and we can be sure that, sooner or later, any relevant information in the corpus will have been analyzed, do we consider filtering out useless semantic links. At this stage, we have also proved that a statistical measure based on the frequency of word sequences of arbitrary length can be used in practice to deal with semantic clustering even on very large corpora.

As a whole, preliminary experimental results seem to corroborate a promising approach as an unsupervised alternative to classic ones, but also as a possible response to solving under-specification and uncertainty problems in dealing with knowledge acquisition on unexplored domains, which could be a significant advantage for redeploying the system when no external resources are yet available.

7. REFERENCES

[1] M. Alonso, D. Cabrero, E. de la Clergerie, and M. Vilares. Tabular algorithms for TAG parsing. In *Proc. of the 9th Conf. of the European Chapter of the* ACL, pages 150–157. ACL, 1999.

[2] N. Aussenac-Gilles and J. Mothe. Ontologies as background knowledge to explore document collections. In RIAO *2004 , Avignon*, 2004.

[3] D. Bourigault, N. Aussenac-Gilles, and J. Charlet. Construction de ressources terminologiques ou ontologiques à partir de textes : un cadre unificateur pour trois études de cas. *Revue d'Intelligence Artificielle (*RIA*) M. Slodzian (Ed.)*, 18:87–110, 2004.

[4] E. de la Clergerie. DyALog: a tabular logic programming based environment for NLP. In *Proc. of 2nd Int. Workshop on Constraint Solving and Language Processing*, Barcelona, Spain, 2005.

[5] E. de la Clergerie. From metagrammars to factorized TAG/TIG parsers. In *Proc. of* IWPT'05, pages 190–191, Vancouver, Canada, Oct. 2005.

[6] A. Joshi. An introduction to Tree Adjoining Grammar. In A. Manaster-Ramer, editor, *Mathematics of Language*, pages 87–114. John Benjamins Company, 1987.

[7] Kavalec, Maedche, Svatek. Discovery of lexical entries for non-taxonomic relations in ontology learning.

[8] G. Rousse and E. de la Clergerie. Analyse automatique de documents botaniques: le project Biotim. In *Proc.* TIA'95, pages 95–104, Rouen, France, Apr. 2005.

[9] B. Sagot and E. de la Clergerie. Error mining in parsing results. In *Proc. of the 21st Int. Conf. on Computational Linguistics and 44th Annual Meeting of the* ACL, 2006.

[10] R.C. Schank, J.L. Kolodner and G. DeJong. Conceptual information retrieval. In SIGIR *pages 94-116*, 1980.

[11] E.M. Voorhees. Natural language processing and information retrieval. In *Maria Teresa Pazienza editor,* SCIE*, pages 32-48, Lecturers Notes in Computer Science, Springer*, 1999.