An Algorithm to Find Words from Definitions

Dominique Dutoit*, Pierre Nugues†

* Memodata, 17, rue Dumont d'Urville, 14000 Caen, France
CRISCO, CNRS, Université de Caen, 14032 Caen Cedex, France
dutoit@info.unicaen.fr

† Computer Science Department,
Lund University, LTH, Box 118
221 00 Lund, Sweden.
Pierre.Nugues@cs.lth.se

Abstract

This paper presents a system to find automatically words from a definition or a paraphrase. The system uses a lexical database of French words that is comparable in its size to WordNet and an algorithm that evaluates distances in the semantic graph between hypernyms and hyponyms of the words in the definition. The paper first outlines the structure of the lexical network on which the method is based. It then describes the algorithm. Finally, it concludes with examples of results we have obtained.

1. Introduction

Le mot juste – the right word – consists in finding a word, and sometimes the only one, that describes the most precisely an object, a concept, an action, a feeling, or an idea. It is one of the most delicate aspects of writing. Generations of students, writers, or apprentice authors have probably experienced this. Unfortunately, we must too often content ourselves with approximations and circumlocutions.

The right word is also crucial to formulate accurately the elements of a problem and solve it. Naming a broken or defective part in a car or a bicycle is a challenge to any average driver when confronted with a mechanic. The right word is yet essential to find the part number in a database, order it, and have it replaced. This problem is even more acute when no human help is possible as for some e-commerce applications where access to information is completely automated.

When the adequate vocabulary escapes us, a common remedy is to employ a circumlocution, a description made of more general words. Examples of such circumlocutions are dictionary definitions that conform to the Aristotelian tradition as in *une personne qui vend des fleurs* (a person that sells flowers) to designate a *fleuriste* (a florist) or *la petite roue dentée au centre d'une roue de vélo* (the small toothed wheel in the center of a bicycle wheel) for *pignon* (sprocket-wheel).

This kind of definitions consists of two parts. A first one relates the object, the idea in question, to a *genus* to which the object, the idea belong, here *personne* (person). Then, the second part specifies it with a *differentia specifica*, a property that makes the object particular, here *qui vend des fleurs* (who sells flowers). A florist can thus be described as a species within the genus *personne*, with the *differentia specifica* "qui vend des fleurs."

The description of the florist corresponds closely to its definition in the French *Petit Robert* dictionary: *Personne qui fait le commerce des fleurs* (a person who trades in flowers). In the *Cambridge International Dictionary of English*, the definition is slightly more restrictive: *a person who works in a shop which sells cut flowers and plants for inside the house*. However, the correspondence between somebody's wording of a concept and the word definition in a dictionary is not always as straightforward.

2. The Lexical Database

2.1. The Integral Dictionary

The Integral Dictionary – TID – (see Dutoit (1992) and Dutoit (2000) for details) is a semantic network associated to a lexicon. It is available mainly for French and it is currently being adapted to other languages notably English and German. Its size is comparable to that of major lexical networks available in English such as WordNet (Fellbaum, 1998) or MindNet (Richardson et al. 1998).

A subset of the Integral Dictionary forms the core of the French lexicon in the EuroWordNet database (Vossen 1999). Although the structure of the Integral Dictionary and WordNet do not map exactly, it was possible to derive TID data and feed them in a WordNet compatible structure. In addition, the TID structure will be used in the Balkanet European project to merge the word nets for Balkan languages (Greek, Turkish, Bulgarian, Romanian, Czech, and Serbian) in a single database (Balkanet 2000).

2.2. The Structure of The Integral Dictionary

The Integral Dictionary organizes words into varieties of concepts and uses semantic lexical functions. Concept definitions are based on the componential semantic theory (Pottier 1974; Greimas 1986) and the lexical functions are inspired by the Meaning-Text theory (Mel'cuk 1992).

Both lexical functions and componential semantics can be accessed in the Integral Dictionary using a Java application programming interface (API). There are more than 30 API functions with parameters for seven languages: English, French, Spanish, Italian, Dutch, German, and Portuguese. In this article, we use five functions.

2.2.1. A Graph of Concepts

The basic component of the Integral Dictionary is the concept. The concept has a gloss of few words to identify its content. When the concept is entirely lexicalized, it gives the definition of a word (this case is marked with a particular kind of relation between the concept and the word: *Generic*). It happens that the concept may be only partially lexicalized.

This is a difference with WordNet where synsets group synonyms. In the Integral Dictionary, concepts group words that share a part of a meaning and they are not equivalent to synsets. A graph of concepts forms then a structure around which the words are organized: a kind of small world founded on an idea (Ferrer and Solé, 2001).

A starting `\` denotes a concept as in `\personne humaine` (human being) or `\animal à fourrure` (fur animal). Concepts are classified into categories. This paper describes only two main ones: the classes and the themes. Classes form a hierarchy and are annotated with their part of speech such as [\N] or [\V]. Themes are concepts that can serve as a predicate to the hierarchies of classes. They are denoted by a [T].

Words in the dictionary appear as terminal nodes in the hierarchical graph of concepts as shown in Figure 1 for the word *fleur* (*flower*). Relations annotate arcs between concepts – themes and classes – and between words and concepts. Major relations are hypernymy (Gen), hyponymy (Spec), various forms of synonymy, ToTheme, and ToClass.

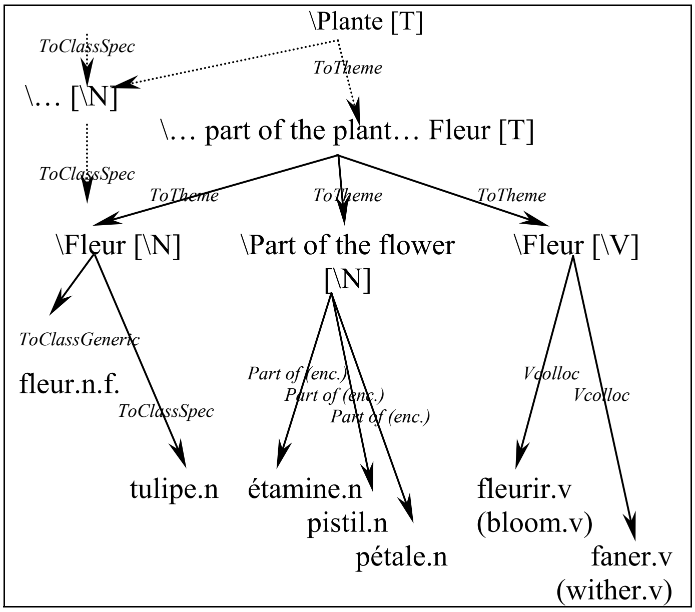


Figure 1 Graph of concepts for the word *fleur*.

The way to organize words and concepts in the Integral Dictionary is a crucial difference with WordNet. In WordNet, concepts are most of the time lexicalized under the form of synonym sets – synsets. They are then tied to the words of a specific language, in this case English.

In the Integral Dictionary, Themes and Classes don't depend on the words of a language and it is even possible to create a concept without any words. This is useful, for example, to build a node in the graph and share a semantic feature that is not entirely lexicalized.

A set of French adjectives shares the semantic feature *qui a cessé quelque chose: d'être, de subir, de devenir.* "Cease something: being, suffering, becoming" as the words *mort* "dead" that is no longer living*, démodé* "old-fashioned, outmoded", that is no longer modern. As it does not exist any French adjective, which means only *no longer*, it is not possible to create a WordNet synset. In effect, they are based on an optional gloss and an obligatory word that corresponds exactly to the meaning. In the Integral Dictionary, there is no such constrain and it is possible to create a class that is not lexicalized.

2.2.2. The Size of the Integral Dictionary

The Integral Dictionary contains approximately 16,000 themes, 25,000 classes, the equivalent of 12,000 WordNet synsets (with more than one term in the content), and, for French 190,000 words. There is a total of 389,000 arcs in the graph. Table 1 shows the word breakdown according to their category.

Part-of-speech	Number
Nouns	138,658
Adjectives	20,981
Verbs	21,956
Adverbs	4,287

Table 1 Size of the lexicon broken down by category.

2.2.3. Componential Semantics

Componential semantics corresponds to the decomposition of the words into a set of smaller units of meaning: the semes (Greimas, 1986; Pottier, 1974).

The term 'seme' is not very common in English although this concept can prove very effective and instrumental in the construction of a semantic network. English-speaking linguists prefer the phrases semantic feature or semantic component, which are not exactly equivalent.

Following the French semantic tradition, the interpretation of a text is made possible by the semes distributed amongst the words (Greimas, 1986; Eco, 1979).

The repetition of semes in a text ensures its homogeneity and coherence and forms an isotopy.

One problem raised by the semic approach is the choice of primitives. Although, there is no consensus on this, a well-shared idea is that the primitives should be a small set of symbolic and atomic terms.

This viewpoint may prove too restrictive and misleading in many cases. In effect, there are multiple ways to decompose a word that correspond to its possible paraphrases and to different contexts as for *fleuriste*/florist:

Semes(fleuriste) = [personne/person] [vendre/sell]
[fleur/flower]
Semes(fleuriste) = [vendeur/seller] [fleur/flower]
Semes(fleuriste) = [personne/person] [travailler/work]
[magasin/shop] [vendre/sell] [fleur/flower]

The Integral Dictionary adopts a componentional viewpoint but the decomposition is not limited to a handful of primitives. Any concept is a potential primitive and the possible semes of a word corresponds to the whole set of concepts connected to this word. Word semes can easily be retrieved from the graph of themes and classes. This approach gives more flexibility to the decomposition while retaining the possibility to restrain the seme set to specific concepts (Figure 2).

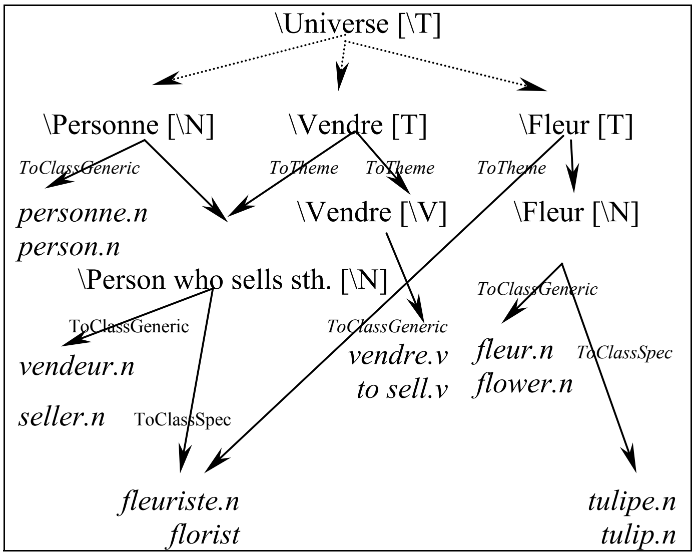


Figure 2 A part of the semantic decomposition of *fleuriste*.

2.2.4. Lexical Semantic Functions

Lexical semantic functions generate word senses from another word sense given as an input. Functions are divided into subsets. Amongst the most significant ones, a subset, S0, S1, and S2, carries out semantic derivations of verbs. These functions could be compared to nominalization in derivational morphology but they operate in the semantic domain and are applied to a specific verb case:

- S0(*acheter*/buy.v) = *achat* (morphological nominalization)
- S1(*acheter*) = *acheteur*/buyer (subject nominalization),
- S2(*acheter*) = {*achat*, *marchandise*, *service*}/{purchase, goods, service} (object nominalization),

The Integral Dictionary has implemented 66 lexical functions in total. It corresponds to 96,000 links between the words. The links between adjectives and nouns are amongst the most productive ones in the French part of the Integral Dictionary.

3. An Algorithm to Find Words from Definitions

The algorithm searches words using two main mechanisms. The first one extracts sets of words from the database that delimit the search space. In the definition "a person who sells marguerites", the algorithm extracts all the sets of persons.

The second mechanism computes a semantic distance between each candidate word in the person sets and the definition. This distance is asymmetric and is based on the structure of the *differentia specifica* and the semantic topology of TID.

As we can imagine, such sets can be very large. The sets corresponding to *person* cover more than 10,000 words in TID. When needed, a third mechanism prunes rapidly the search space (Bertholon 1998).

3.1. The Semantic Network

The Integral Dictionary superimposes two graphs. A first one forms an acyclic graph whose terminal nodes are the words, the other nodes are concepts, and where the arcs correspond to relations. A second one connects the words using and lexical functions. Figure 3 shows a simplified picture of this structure. The distance between two words or phrases is derived from the first graph.

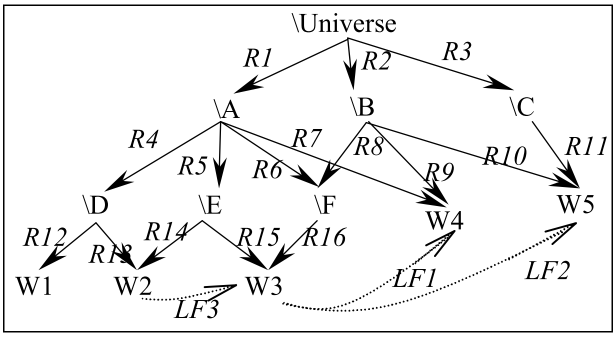


Figure 3. The graph of concepts, words, relations, and lexical functions.

In Figure 3, nodes beginning with a backslash '\' are concepts while W1, W2, W3, etc. are words. The root node of the graph is the \Universe label, which is the ancestor of all the concepts. It has a three children respectively \A, \B, and \C, which can either classes or themes.

Arc labels Rn are relations linking the concepts and LFn are lexical functions. In Figure 3, *W3* has two parents connected by arcs representing two different relations: R14(\E) = W3 and R15(\F) = W3. LF1 is a lexical function linking W3 to W4: LF1(W3) = W4. Inverse relations and lexical functions are implemented so that a parent can be found from its child.

The average number of parents of a word or a concept in the Integral Dictionary is 2.1. The average depth of the graph from the root is 15. From these numbers, we can evaluate the average number of concepts a word can be member of: $15^{2.1}$ = 294.

3.2. A Semantic Distance

The distance between two words or phrases is derived from the graph topology as shaped by the relations. It is the sum of two terms that we call respectively the semantic activation distance and the semantic proximity distance. We describe here a simplified version of this distance.

3.2.1. The Semantic Activation

The semantic activation of two words, M and N, is defined by their set of least common ancestors (LCA) in the graph (Aho et al, 1973). The semantic activation paths correspond to paths linking both words M and N through each node in the set of least common ancestors.

In Figure 3, we have LCA(*W*2, *W*3) = {\E} and LCA(W3, W4) = {\A, \B}. The activation path between *W*2 and *W*3 consists of the nodes *W*2 \E *W*3 with the functions R14^{-1} and R15. The path between *W*3 and *W*4 consists of *W*3 \E \A *W*4 and *W*3 \F \B *W*4.

We define the semantic activation distance as the number of arcs in theses paths divided by the number of paths. We denote it d^. In Figure 3:

$$d\hat{}(W2, W3) = (1 + 1) / 1 = 2$$
$$d\hat{}(W3, W4) = ((2 + 1) + (2 + 1)) / 2 = 3$$

Conceptually, the least common ancestors delimit small concept sets – small worlds – and provide a convenient access mode to them. They enable to extract a search space of potential semes together with a metric.

3.2.2. The Semantic Proximity

The semantic proximity between two words, M and N, uses sets of asymmetric ancestors that we call the Least Asymmetric Ancestors, LAA. LAA(M, N) is the set of nodes that are common ancestors of both words, that are not member of the least common ancestor set (LCA), and where M has a child, which is an ancestor of M and not an ancestor of N. Most of the time, the sets LAA(M, N) and LAA(N, M) are different. This is an essential feature of this metric that reflects a semantic difference.

In Figure 3, the set of the ancestors common to *W*2 and *W*3 that are not in the LCA set is {\A, \Universe}. \A has a child \D that is an ancestor of *W*2, which is not an ancestor of W3, hence LAA(*W2*, *W3*) = {\A}. The set LAA(*W3*, *W2*) = {\A, \Universe} because \F and \B are children of respectively \A and \Universe and ancestors of \W3 but not of \W2.

The semantic difference is the sum of distances of M to all the members of both LAA sets and N to all the members too:

$$SD(M,N) = \frac{\sum_{E \in LAA(M,N) \cup LAA(N,M)} d(M,E) + d(N,E)}{Card(LAA)}$$

We have:

$$SD(W2, W3) = (2+2) / 1 = 4$$
$$SD(W3, W2) = ((2 + 2) + (3 + 3)) / 2 = 5$$

Finally, the distance we use is the sum of the semantic activation and the semantic proximity, d = d^ + SD:

$$D(W2, W3) = (2 + 4) / 2 = 3$$
$$D(W3, W2) = (2 + 5) / 2 = 3.5$$

3.2.3. Examples of Semantic Activation and Semantic Proximity

In this section, we take the words florist (noun) and flower (noun) to illustrate with concrete examples what the LCA and LAA sets are. The results enable to outline the componential structure of the dictionary and show understandable outputs in terms of semes.

Although the words are entered in French, the concepts are roughly equivalent in English:

- LCA(florist.n, flower.n) = {\Flower [T], \RootOfTheNoun [\Grammar]}
- LAA(florist.n, flower.n) = {\TheWorldOfTheLiving [T], \HumansAndSociety [T], X_i [T], …}

where X_i [T] denotes the remaining members of the LAA set.

Often, the LAA set contains the root of the dictionary. In our case, we obtain 107 LAA from *florist.n* to *flower.n*. An automatic examination of the results shows that most of the LAA concepts are obtained through a very small number of classes. To find these classes, we traverse the graph using the LAA (\Universe [T] in Fig.2) down to the first class above *florist.n*. We then find where the difference between *florist* and *flower* originates:

- \Person [\N], which mean that flower.n is not a person.
- \Seller [\N], which means that flower.n has no link with business.

A study of the LCA is also interesting. In our case, we obtain:

- \Flower [\T], which means that both flower.n and florist.n have this seme.
- \GRAMMAR:NOUN [\GrammarN], which means that flower.n and florist.n share the part of speech “noun”.

In conclusion, this means that *florist* and *flower* are both nouns and that they both belong to the world of flowers. The difference between *florist* and *flower* is that a *florist* is a person, and that this person has the activity of *selling something*.

These results show that LCA and LAA are powerful tools to derive common sense meaning and that they can be used to compare words.

3.3. Finding the Right Word

The algorithm finds words from definitions using two main mechanisms. The first one extracts the sets of words from the database that delimit the search space. In the definition “a person who sells marguerites”, the algorithm extracts the hyponyms of person: the set of all the persons (more than 10,000 nouns in the Integral Dictionary).

The second mechanism computes the distance between each candidate word in the person sets and the words in the *differentia specifica*. To accelerate the algorithm, for large concepts like \Person [\N], a preliminary task attempts to reduce the search space to subset of it. See Berthelon (1998) for details.

3.3.1. Extracting a Set of Hyponyms

The sets of words are extracted using a function of a given word that finds all the hyponyms of one of the word’s hypernyms. This extraction requires a composition of relations slightly more complex in the Integral Dictionary than in WordNet. Figure 4 shows the hyponymy relationships of *flower.n* in both lexical networks.

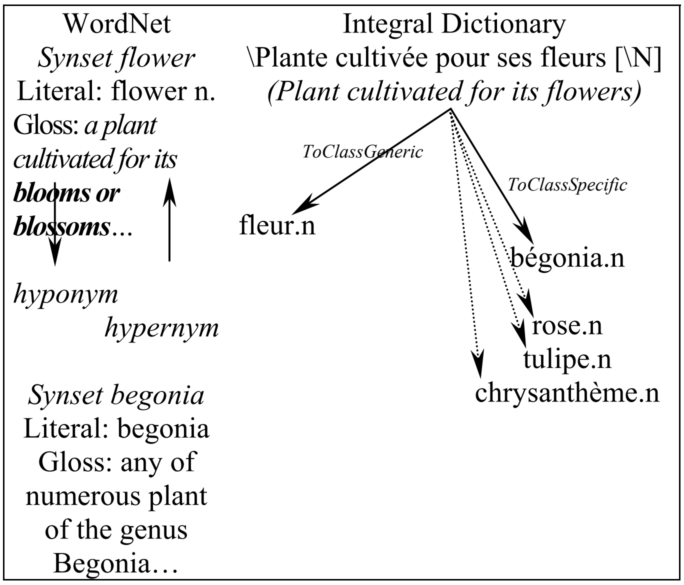

WordNet
Synset flower
Literal: flower n.
Gloss: *a plant cultivated for its* ***blooms or blossoms****...*
hyponym
hypernym
Synset begonia
Literal: begonia
Gloss: any of numerous plant of the genus Begonia…
Integral Dictionary
\Plante cultivée pour ses fleurs [\N]
(Plant cultivated for its flowers)
ToClassGeneric
ToClassSpecific
fleur.n
bégonia.n
rose.n
tulipe.n
chrysanthème.n

Figure 4 Hyponyms/hypernyms links in WordNet and in the Integral Dictionary.

Figure 4 shows that *begonia* is linked to *flower* by one single link in WordNet while the Integral Dictionary requires two symmetric links. The first one connects *fleur*.n to the class *\plante cultivée pour ses fleurs [\N]*. A second one connects this class to *bégonia.n*. This feature makes the search more complex but adds more flexibility to describe the lexicon. In the end, it is possible to extract sets of related words using a composition of hypernymy and hyponymy functions in both networks. In the Integral dictionary, it corresponds to the ToClassSpecific and ToClassGeneric functions.

ToClassSpecific ° ToClassGeneric (fleur.n) = {bégonia.n, rose.n, tulipe.n, …}

We call this composition Specific.

3.3.2. Ranking the Extracted Words

Then, to find answers from the query *vendeur de fleurs (seller of flowers),* we first extract all the words corresponding to salespeople in the lexical database:

Specific(vendeur.n) = {*vendeur, boulanger, boucher, papetier, fleuriste, bouquetière,* etc.}[1]

Using the measure of the semantic proximity, we compute:

1. The semantic proximity between *vendeur de fleurs* and the extracted words: d(vendeur de fleurs, X)
2. The semantic proximity between the extracted words and the phrase *vendeur de fleurs*: d(X, vendeur de fleurs*)*

where X is the extracted word for *specific*(*vendeur).*

Let's show why we need both measures..

Let's suppose that we only use the first asymmetric difference. As in the query *vendeur de fleurs,* we have no seme about the gender of the seller, it will be impossible to make a difference between the two specifics: *fleuriste* and *bouquetière.* These two specific terms saturate all the semes of the query *vendeur de fleurs.* To differentiate between *fleuriste* and *bouquetière,* we need to produce the two measures (2): *d(fleuriste, vendeur de fleurs)* and *d(bouquetière, vendeur de fleurs).* With this second measure, we obtain that *bouquetière* has a seme (feminine gender) non saturated by the query. It is not the same for *fleuriste*, which has no non-saturated seme with this query. To solve our problem (to distinguish the specifics *bouquetière* and *fleuriste* from *seller of flower)*, the measure 2 was needed.

As, in the answers *seller* or *shop assistant* we have no seme about what is selling, the measure (2) will say that any semes of *seller* or *shop assistant* are saturated by the queries *seller of flowers.* To differentiate between *fleuriste* and *one of these specifics,* we need to produce the measure (1): d(*vendeur de fleurs, fleuriste),* d(*vendeur de fleurs, seller),* d(*vendeur de fleurs, shop assistant).* ?? Il faudrait mieux mettre les mots en francais ??

If we compute these measures, we can observe that only *florist* saturates the query *vendeur de fleurs.* To solve our problem (to distinguish the specifics *seller* and *fleuriste* from *seller of flowers)*, the measure 1 was needed.

The figure 5 summarizes these results.

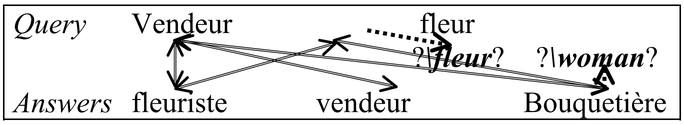


Figure 5. The measure top to down (2) sees the distinctive seme *flower* and eliminates *vendeur,* the measure down to top (1) sees the distinctive seme *woman* and eliminates *bouquetière.*

3.3.3. Complex Queries

In the previous sections, we have described queries using one term to distinguish between the specific terms. We have used in the last section some virtual points of the dictionary by adding the seme of different words (for example, a virtual point based on the points of *seller* and *flower*). Then, the calculation has compared this virtual point *seller + flower* to a lexical point, by example *florist.* It's possible to use the same process with number of words in the query. Sometime, a horizon effect may appear. In this case, we use graph techniques to limit the effect. These techniques that use together LAA, LCA and offsets of the words will be not describe in the paper. We will suppose that it's possible to add numerous semes in a semantic phrases to represent queries as complex as *small-toothed wheel in the center of a bicycle wheel.*

4. Results

Table 2 shows the words found by the algorithm for the phrase given in the introduction: *Personne qui vend des fleurs*.

[1] {*shop assistant, baker, butcher, stationer, florist, female flower seller,* etc.}

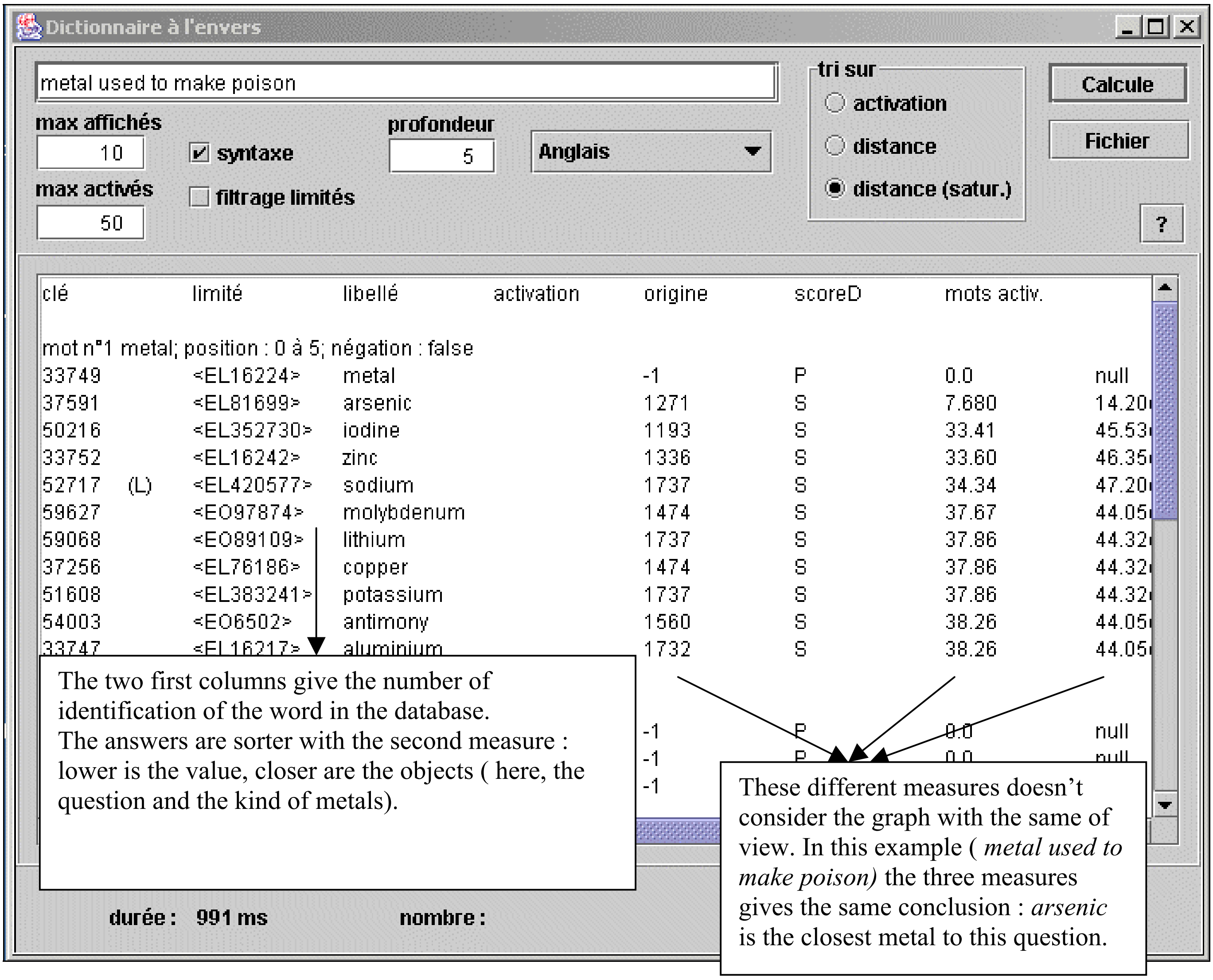
The probability to select the good answer in our example (florist) is 1/10,000. As we can see, the algorithm provides other words close to the definition: flower grower, flower seller, horticulturist, etc. These terms are ranked by the proximity as indicator of relevance: the lower, the closer. Proximity is the average of the two measures given above.

Rank	Word	English translation	Proximity
1	Fleuriste	Florist	1.34
2	Floriculteur	Flower grower	1.57
3	Vendeur	Person who sells something	1.77
4	Bouquetière	Flower seller in a street	1.84
5	Horticulteur	Horticulturist	2.21
6	Rosiériste	Rose grower	2.35
7	marchand	Tradesman	2.57
8	Maraîcher	Market gardener	2.71
9	Paysagiste	Landscape painter	3.12
10	Fruiticulteur	Fruit farmer	3.93

Table 2 Words corresponding to the phrase *Personne qui vend des fleurs*.

The next picture shows the interface of this API made by the five basic mechanisms.

In this screen, the question is given in English, and the answers are selected in English.



Dictionnaire à l'envers

metal used to make poison

max affichés 10

☑ syntaxe

profondeur 5

Anglais

max activés 50

□ filtrage limités

tri sur
○ activation
○ distance
● distance (satur.)

Calcule

Fichier

?

```
clé          limité      libellé       activation    origine    scoreD     mots activ.

mot n°1 metal; position : 0 à 5; négation : false
33749            <EL16224>    metal                      -1         P          0.0        null
37591            <EL81699>    arsenic                    1271       S          7.680      14.20
50216            <EL352730>   iodine                     1193       S          33.41      45.53
33752            <EL16242>    zinc                       1336       S          33.60      46.35
52717    (L)     <EL420577>   sodium                     1737       S          34.34      47.20
59627            <EO97874>    molybdenum                 1474       S          37.67      44.05
59068            <EO89109>    lithium                    1737       S          37.86      44.32
37256            <EL76186>    copper                     1474       S          37.86      44.32
51608            <EL383241>   potassium                  1737       S          37.86      44.32
54003            <EO6502>     antimony                   1560       S          38.26      44.05
33747            <EL16217>    aluminium                  1732       S          38.26      44.05
                                                         -1         P          0.0        null
                                                         -1         P          0.0        null
                                                         -1
```

durée : 991 ms   nombre :

The two first columns give the number of identification of the word in the database.
The answers are sorter with the second measure : lower is the value, closer are the objects (here, the question and the kind of metals).

These different measures doesn't consider the graph with the same of view. In this example (*metal used to make poison)* the three measures gives the same conclusion : *arsenic* is the closest metal to this question.

Figure 6. Finding words from definition API

The figure 7 shows the main APIs of the complete system.

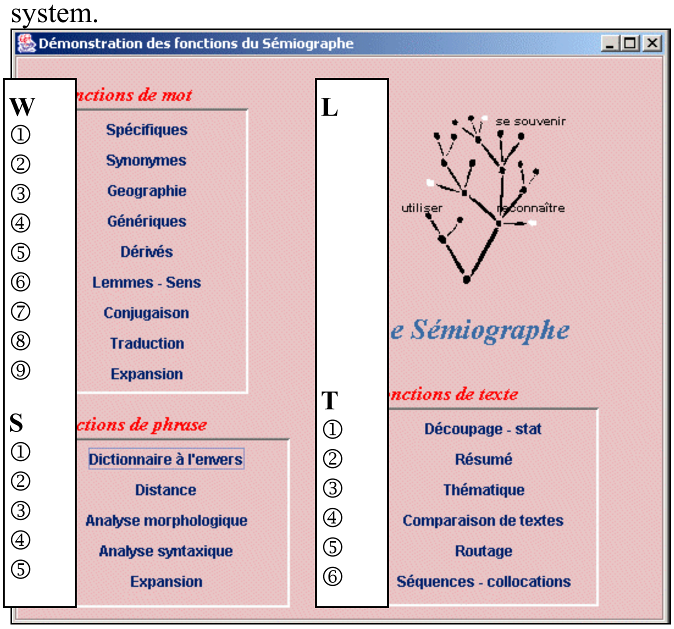


Figure 7. The main menu of the Semiograph

We give details to the APIs showed in French in the figure 7. The APIs with the one * was used in this paper. **(*)**

- **W** : Group the APIs based on the words.
  - ① : to obtain all the specifics (hyponyms) of a word or a word-meaning **(*)**
  - ② : to obtain synonyms of a word or a word-meaning **(*)**
  - ③ : to obtain information about the topological and political geography
  - ④ : to obtain all the generics (hypernyms) of a word or a word-meaning
  - ⑤ : to obtain derived forms of a word or a word-meaning. There is more than 60 lexical functions in the dictionary.
  - ⑥ : to obtain lemmas
  - ⑦ : to obtain flexional form
  - ⑧ : to obtain translation
  - ⑨ : to make a batch with the previous APIs.
- **S** : Group the APIs based on the sentence (more than one word, and less than a complete text).
  - ① : the APIs if this article : finding words from definition
  - ② : various semantic distances **(*)**
  - ③ : morphological analysis **(*)**
  - ④ : syntactical analysis (POS disambiguation) **(*)**
  - ⑤ : the same of W9, but with the integration of syntactical constrain on a sentence (Query expansion).
- **L** : The tree gives the access to many technical information about the system
- **T** : Group the APIs based on a complete text
  - ① : to obtain statistics on a text
  - ② : to obtain automatic summarisation
  - ③ : to obtain APIs to check the consistency of topics in a particular text
  - ④ : to compare two texts
  - ⑤ : to mail texts (push) to particular destination defined by the topics or syntactical patterns
  - ⑥ : to produce cluster information from a lot of dimension of a text (morphological, syntactical or semantic dimensions

To conclude this presentation, we give some example of the results in the table 3.

Query	Result
Crier pour un dindon Cry of a male turkey	Glouglouter Goggles
Vendeur de fleurs/magnolia/plantes Seller of flowers/magnolia/plants	Fleuriste Florist
Métal jaune Yellow metal	Or/soufre Gold/Sulfur
Métal de la finance Metal of the finance	Or/argent Gold/silver
Métal qui provoque des maladies Metal which induces disease	Plomb/arsenic Lead/arsenic
Petite roue dentée au centre d'une roue de bicyclette Small-toothed wheel in the center of a bicycle wheel.	Pignon Sprocket-wheel

Table 3 Other results. The table shows only the word ranked first.

5. Discussion and Perspectives

We have described a lexical database and an algorithm to find words from definitions. We have presented examples of the results we obtained. The core of the algorithm rests on two functions, LCA and LAA, that query the database to find sets of semes describing similarities and differences between two words. In addition to finding words from definition, the LCA and LAA functions help us to check the consistency of the lexical network. These functions should report semes corresponding to word differences and similarities. When the semes don't correspond, this generally indicates some faulty link in the network.

Currently, the algorithm has been applied mainly to the French part of the TID, but the functionalities are available for other languages, as the example of the figure 6 shows it.

The industrial application of these algorithms are numerous :

- accessing to key-words or descriptor terms
- finding answers from questions. In this algorithm, the searched things are words. But it's easy to consider that, for one application, searched things are small texts. In this case, we have only to declare the texts as specific things of a generic defined by the application.
- In the same approach, finding pictures from there description is possible : in this case, we have only to index the description of the pictures in the graph, and the whole picture as a searched thing of this kind of application.

But the most important is not in these industrial applications. Our point of view is that the problem of finding word from definitions will be very important in the future of natural language processing for two main reasons. The first reason is that, for this application, the results are good or bad. It's not the same in WSD for example. The second reason is that this application provide us a formidable tool to check the consistency of every electronic dictionary.

6. References

V. Aho, J. E. Hopcroft, J. D. Ullman, 'On computing least common ancestors in trees', *Proc. 5th Annual ACM Symposium on Theory of Computing, 1973,* pp. 253--265.

'Balkanet Project', *IST-2000-29388, 2000.*

Raphaël Berthelon, 'Le Semiographe', *Rapport de stage, ISMRA, Caen, 1998.*

Dominique Dutoit, 'Quelques opérations sens→texte et texte→sens utilisant une sémantique linguistique universaliste a priori', *PhD thesis, Université de Caen, 2000.*

Dominique Dutoit, 'A Set-Theoretic Approach to Lexical Semantics', *Proceedings of COLING, 1992.*

Umberto Eco, 'Lector in fabula', *Bompiani, 1979.*

Christiane Fellbaum (ed), 'WordNet: An electronic lexical database,' *MIT Press, 1998*

Ramon Ferrer I Cancho, Ricard V. Solé, 'The small-world of human language', *Proceedings of the Royal Society of London, B 268, 2261-22661, 2001.*

Algirdas Julien Greimas, 'Sémantique structurale', *Coll. Champs sémiotiques, PUF, 1986.*

Igor Mel'cuk, 'Dictionnaire Explicatif et Combinatoire du français contemporain (DEC), Recherche Lexico-sémantiques III'*, Presses de l'Université de Montréal, Québec, 1992.*

Bernard Pottier, 'Linguistique générale, Théorie et description', *Klincksieck, 1974.*

James Pustejovsky, 'The Generative Lexicon', *MIT Press, 1995*

Stephen D Richardson, William B. Dolan, Lucy Vanderwende, 'MindNet: acquiring and structuring semantic information from text', *Proceeding of COLING'98, 1998.*

Piet Vossen, 'EuroWordNet, Final Report', *D041, LE2-4003, LE4-8328, 1999.*