Proceedings of the Twenty-Third International Florida Artificial Intelligence Research Society Conference (FLAIRS 2010)

# Disambiguation of Textual Data Typification for the Purpose of Categorial Analysis

**Adam Joly, Ismail Biskri and Boubaker Hamrouni**

Laboratoire de Mathématiques et Informatique Appliquées (LAMIA)
Université du Québec à Trois Rivières
C.P. 500, Trois Rivières (Québec) G9A 5H7, Canada
{adam.joly; ismail.biskri}@uqtr.ca

**Abstract**

The operation of categorial type assignment is prior to a categorial analysis. The lexical units, which are entries in a dictionary, are commonly associated to one or more categorial types. Therefore, we need to determine for each unit in a sentence the correct categorial type to be assigned. Current research on Categorial Grammars is not paying attention enough to this issue. The assignment of categorial types is often done either manually, or with ad hoc heuristics. In this paper we present a method based on conditional probabilities.

## 1. Introduction

Categorial Grammars assign categorial types to lexical units so they act as an operator on or an operand of (or both) other lexical units. We consider two base categorial types, N and S, which respectively stand for nominal syntagm and sentence. More complex and oriented categorial types are built recursively using these base types, plus a right (/) or a left (\) construction operator. For instance, the categorial type of a transitive verb is (S\N)/N, because it acts on a nominal syntagm in object position (to the right) in order to construct a verbal syntagm that is itself applied to a nominal syntagm in subject position (to the left) so a complete sentence is formed.

In *table 1*, we present some of the common categorical types (although very incomplete) that can be assigned to a given syntactic category.

| Syntactic category | Categorial type | Example |
|---|---|---|
| Nominal syntagm | N | |
| Sentence | S | |
| Determinant | N/N | the: *the-book* |
| Adjective | N/N | good: *a-good-day* |
| | N\N | available: *the-time-available* |
| Transitive verb | S\N | runs: *Luke-runs* |
| Intransitive verb | (S\N)/N | loves: *Luke-loves-Lisa* |
| Adverb | (N\N)\(N\N) | very: *very-good* |
| | (S\N)\(S\N) | quickly: *he-speaks-quickly* |

**Table 1: Some categorial types**

A syntactic category can be of different categorial types. As an example, an adjective can operate on a nominal syntagm located to its right (N/N) or left (N\N). Moreover, two different syntactic categories can be of the same categorial type. This is the case of the determinant and the adjective, which share the categorial type N/N.

The model of the Applicative Combinatory Categorial Grammars (ACCG) (Desclés, Biskri, 1996 ; Biskri, Bensaber, 2008), as well as models such as the Combinatory Categorial Grammars (CCG) (Steedman, Baldridge, 2009), or its extension, the Multi-Modal Combinatory Categorial Grammars (MMCCG) (Baldridge, Kruijff, 2003), verifies the syntactic correctness of sentences by the means of standard forward and backward application rules and also type raising, functional composition and substitution rules derived from theorems of the Lambek calculus (Lambek, 1961) and from the combinatory logic of Curry (Curry, 1958) (Shaumyan, 1998). These rules are operated on the lexical units through a process of quasi-incremental left to right analysis, not only to prove the syntactic validity, but also to construct the underlying functional semantic interpretation of a sentence.

Even though further presentation of these models exceeds the scope of this paper, we would like however to show to the reader some of the combinatory rules and a short example of a categorial analysis in ACCG so he can have an illustrated overview of how the process works and a better appreciation about what we will be proposing afterward.

Application rules:

$$\frac{[X/Y : u_1] - [X : u_2]}{[X : u_1\, u_2]}> \qquad \frac{[Y : u_1] - [X\backslash Y : u_2]}{[X : u_2\, u_1]}<$$

Type raising rules:

$$\frac{[X : u_1]}{[Y/(Y\backslash X) : (C_*\, u_1)]}>T \qquad \frac{[X : u_1]}{[Y\backslash(Y/X) : (C_*\, u_1)]}<T$$

Functional composition rules:

$$\frac{[X/Y : u_1] – [Y/Z : u_2]}{[X/Z: (B\, u_1\, u_2)]}>B \qquad \frac{[Y\backslash Z : u_1] – [X\backslash Y : u_2]}{[X\backslash Z: (B\, u_2\, u_1)]}<B$$

The premise of each rule is a concatenation of lexical units with categorial types and the consequence is an applicative typed expression with the possible introduction of a combinator, i.e. $C_*$ for type raising and B for function composition. In order to assess the syntactic correctness of the sentence, we must obtain an applicative expression of type S while having considered all the lexical units. Then, using rules of combinator reduction, we get the functional semantic interpretation.

The incremental categorial analysis of the sentence “The cat eats a mouse” would go as following:

| | | |
|---|---|---|
| (1) | [N/N : the] [N : cat] [(S\N)/N : eats] [N/N : a] [N : mouse] | |
| (2) | [N : the cat] [(S\N)/N : eats] [N/N : a] [N : mouse] | > |
| (3) | [S/(S\N) : ($C_*$ (the cat))] [(S\N)/N : eats] [N/N : a] [N : mouse] | >T |
| (4) | [S/N : (B ($C_*$ (the cat)) eats)] [N/N : a] [N : mouse] | >B |
| (5) | [S/N : (B (B ($C_*$ (the cat)) eats) a)] [N : mouse] | >B |
| (6) | [S: ((B (B ($C_*$ (the cat)) eats) a) mouse)] | > |
| (7) | ((B (B ($C_*$ (the cat)) eats) a) mouse) | |
| (8) | ((B ($C_*$ (the cat)) eats) (a mouse)) | (B) |
| (9) | (($C_*$ (the cat)) (eats (a mouse))) | (B) |
| (10) | ((eats (a mouse)) (the cat)) | ($C_*$) |

Steps 1 to 6 lead successfully to the categorial type S. Steps 7 to 10 build the functional semantic interpretation “((eats (a mouse)) (the cat))”.

Such an analysis in ACCG, as in any categorial model, is possible only because categorial types are correctly pre-assigned to the lexical units. The major issue about these assignments concerns the fact that, as we mention earlier, to a same lexical unit can correspond more than one different categorial type so we cannot simply automate the process using a one-to-one association table. Thereby, in order to deal with these ambiguities, current strategies employ *ad-hoc* heuristics or manual typification only.

To address the problem of automatic textual typification, we propose in this paper a probabilistic approach for the French language. The following sections will describe the suggested model.

## 2. BDLex

BDLex is a project developed within the research group GDR-PRC CHM (Research Group – Concerted Research Program Man Machine Communication) (De Calmès, Pérennou, 1998 ; Pérennou, 1986).

It consists of a lexical database of approximately 440,000 inflected forms, generated from some 50,000 canonical words in French, for which many properties are specified, covering phonological and morphological aspects such as spelling, pronunciation and morpho syntax, intended to be used for automatic text and speech processing.

*Table 2* shows a sample of lexical entries of the dictionary.

| Spelling | Pronunciation | | Morpho syntax | | | |
|---|---|---|---|---|---|---|
| ORTHO | PHONO | FPH | CS | VS | M | LIEN |
| être *(being)* | E,tR | @ | N | MS | | = |
| être *((to) be)* | E,tR | @ | V | | inf | = |
| sont *((they) are)* | \|so~ | t” | V | 3P | pi | être |
| petites *(little)* | p@tit | @z” | J | FP | | petit |
| un *(a)* | 9~ | @ | d | MS | di | = |
| avion *(plane)* | avjo~ | | N | MS | | = |

**Table 2: Examples of lexical entries in BDLex[2]**

The first column contains the lexical units to which we add the English translation in parenthesis. The two following fields describe the phonological representation (PHONO) and the behavior of the phonological final (FPH) of the lexical unit. The morpho-syntactic fields correspond to the grammatical type (CS), syntactic information on gender and number (VS) and mode (M), and finally the written form of from which the inflected form derives (LIEN).

For our needs, we will only be concerned about the ORTHO and CS fields, although it could be interesting to consider more morpho-syntactic information, especially for languages for which different word orders are possible and the grammatical type alone is insufficient to avoid ambiguities. The next section will show how they are involved into the instantiation and the evolution of a categorial dictionary.

## 3. Construction of a Categorial Dictionary

In order to automatically predict the categorial types of the lexical units of a sentence, we must first build a categorial dictionary along with a Markov transition matrix. We will

[2] Retrieved from the IRIT Web site (www.irit.fr).

extract from it the most probable sequences, determined upon the categorial types associated to the lexical units of the sentence in the categorial dictionary and the known transitions of types. *Figure 1* illustrates the global process.

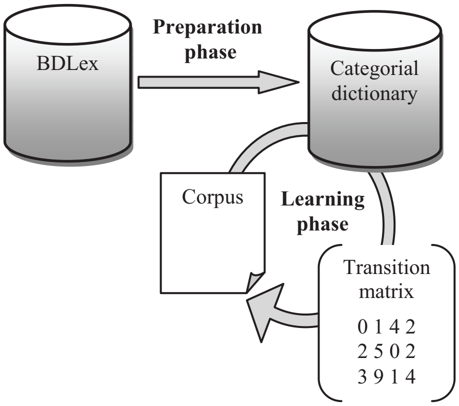

**Figure 1: Construction of a categorial dictionary**

## Preparation phase

The process of preparation consists to add lexical entries of the French language from BDLex into a database, named the categorial dictionary, for which one or more categorial types will be associated to each of the entries.

In order to do so, we applied grammatical to categorial type translation rules, partially shown in *table 3*, for each entry of BDLex.

| Grammatical type (CS value) in BDLex | Categorial type in the categorial dictionary |
|---|---|
| N | N |
| V | (S\N)/N |
| V | S\N |
| J | N/N |
| J | N\N |
| D | N/N |

**Table 3: Grammatical to categorial type translation rules**

Considering the entries of table 2 only and according with the rules presented in table 3, the following entries would be generated into the categorial dictionary:

| BDLex | |
|---|---|
| ORTHO | CS |
| être | N |
| être | V |
| sont | V |
| petites | J |
| un | D |
| avion | N |

| Categorial dictionary | | |
|---|---|---|
| Lexical unit | Categorial type | Frequency |
| être | N | 0 |
| être | (S\N)/N | 0 |
| être | (S\N) | 0 |
| sont | (S\N)/N | 0 |
| sont | (S\N) | 0 |
| petites | N/N | 0 |
| petites | N\N | 0 |
| un | N/N | 0 |
| avion | N | 0 |

**Table 4: Application of the translation rules**

The reader can observe a third column, the frequency, in the categorial dictionary. The purpose of this field will be explained further.

Now that the categorial dictionary has been filled up from the lexical database, we are ready for the initial learning phase.

## Learning phase

To that purpose, we used a corpus of 74 sentences of diverse syntactic structures. Each of them was previously typified and analyzed manually by an expert so we are assured of the syntactic correctness of the sentences and the categorial types associated to their lexical units. The rest of the process is automated.

Because we are interested to know the probability that a given categorial type is followed by another given type, each sentence is seen as a sequence of states, that is to say as a Markov chain, where states are in fact categorial types. The sum of the probabilities of transition for a state is 100%. We consider that every sentence is preceded and followed by the empty type. Consequently, the initial and final state of the Markov chain is the empty state.

Moreover, we need to store in a matrix the transitions between the categorial types in the sentences. In our context, because the lexical units of the corpus have 34 distinct categorial types and considering the "empty" type, that means that we must have a 35 X 35 Markov matrix, for which each type appears as a row and also as a column, instantiated to 0. This is illustrated by *table 5*.

| | Empty | Type 1 | Type 2 | … | Type 34 |
|---|---|---|---|---|---|
| Empty | 0 | 0 | 0 | 0 | 0 |
| Type 1 | 0 | 0 | 0 | 0 | 0 |
| Type 2 | 0 | 0 | 0 | 0 | 0 |
| … | 0 | 0 | 0 | 0 | 0 |
| Type 34 | 0 | 0 | 0 | 0 | 0 |

**Table 5: Initial transition matrix of categorial types**

The next step involves the processing of the parsed sentences one by one. For example, let us consider the sequence of the categorial types of a sentence we used previously: "the cat eats a mouse".

*(Empty, N/N, N, (S\N)/N, N/N, N, Empty)*

The list of transition goes as following:

*Empty* → *N/N*
*N/N* → *N*
*N* → *(S\N)/N*
*(S\N)/N* → *N/N*
*N/N* → *N*
*N* → *Empty*

Once all the transitions of the sentence have been processed, the resulting matrix is as described in *table 6*.

|  | Empty | N | N/N | (S\N)/N | … |
|---|---|---|---|---|---|
| Empty | 0 | 0 | 1 | 0 | … |
| N | 1 | 0 | 0 | 1 | … |
| N/N | 0 | 2 | 0 | 0 | … |
| (S\N)/N | 0 | 0 | 1 | 0 | … |
| … | … | … | … | … | … |

**Table 6: Matrix after processing "The cat eats a mouse"**

At the same time, we search for each pair of lexical unit and categorial type in the categorial dictionary and we increment the frequency of the corresponding entry by one.

If ever a new categorial type *t* occurs for a given lexical unit, we add it into the categorial dictionary and then we use the following algorithm:

```
for each grammatical type g of lexical unit $u_1$ in BDLex
  for each lexical unit $u_2$ with g in BDLex
    if not existing, add the entry ($u_2$, t, 0) into the categorial dictionary
  end for
end for
```

If we come across a new lexical unit – this situation is more likely to happen with named entity – we simply add the entry *(u, t, 1)* into the categorial dictionary.

For our prototype, we processed the 74 sentences of the corpus that way. However, the matrix we obtained is not considered as being a normalized Markov matrix. Indeed, the sum of the values for each row must be equal to 1 (100%). In order that the values represent probabilities, we must divide the value of each cell by the sum of all cells of its row and copy these probabilities into another matrix of the same dimensions, named the transition matrix.

Instead of presenting our 35 X 35 transition matrix, let us simply consider for illustration that there are only three categorial types, X, Y and Z, and that we have a corpus of 50 sentences. After the processing, we suppose that the matrix of occurrences of transitions goes like this:

|  | Empty | X | Y | Z | *Total* |
|---|---|---|---|---|---|
| Empty | 0 | 23 | 0 | 27 | *50* |
| X | 39 | 19 | 0 | 28 | *79* |
| Y | 0 | 21 | 22 | 17 | *60* |
| Z | 11 | 23 | 38 | 18 | *90* |

**Table 7: Example of a matrix of categorial types**

The normalized matrix would be as shown in *table 8*.

|  | Empty | X | Y | Z | *Total* |
|---|---|---|---|---|---|
| Empty | 0 | 0.460 | 0 | 0.540 | *1.000* |
| X | 0.494 | 0.241 | 0 | 0.355 | *1.000* |
| Y | 0 | 0.350 | 0.367 | 0.283 | *1.000* |
| Z | 0.122 | 0.256 | 0.422 | 0.200 | *1.000* |

**Table 8: Transition matrix**

In a transition matrix, P(j|i) represents the probability that a type i transit to a type j, that is to say the probability to have the type i, then the type j. In the previous Markov matrix, there would be 46% of probabilities that the categorial type of the first lexical unit of a given sentence is X and 54% that it is Z.

## Automatic Typification, Validation and Reinforcement

We now have a starting transition matrix based on the categorial types of a few hundred lexical units so the next step consists to test and validate the process of automatic typification, as the same time as to reinforce learning. This process is described below.

First of all, through a graphic interface of our prototype, we ask a user to enter a sentence. Then, from the categorial dictionary, we retrieve all the possible categorial types for each lexical unit of the sentence. Taking the French sentence "Jean court lentement", which means "Jean runs slowly", we suppose the following candidates as categorial types for "Jean", "court" and "lentement":

| Jean | court | lentement |
|---|---|---|
| Type 1.1 | Type 2.1 | Type 3.1 |
| Type 1.2 | Type 2.2 | Type 3.2 |
| … | … | … |
| Type 1.i | Type 2.j | Type 3.k |

**Table 9: Potential categorial types for the lexical units of "Jean court lentement"**

The possible sequences of categorial types can be represented with the help of an oriented graph, as in *figure 2*. Among them, we are interested to find the most probable ones.

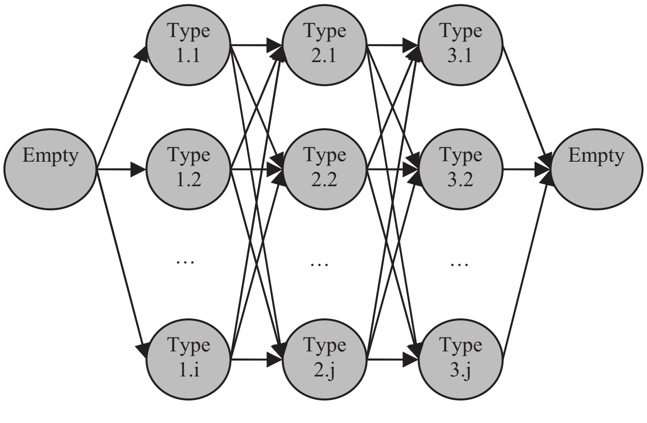

**Figure 2: Graph of possible sequences of categorial types**

Thanks to the transition matrix, we can calculate the relative probability for a given path. For example, if

*P(type 1.1 | empty) = 0.14*,
*P(type 2.1 | type 1.1) = 0.37*,
*P(type 3.1 | type 2.1) = 0.18* and
*P(type 3.1 | empty) = 0.31*,

then the relative probability of the sequence (Empty, Type 1.1, Type 2.1, Type 3.1, Empty) is

*P = P(type 1.1 | empty) * P(type 2.1 | type 1.1) * P(type 3.1 | type 2.1) * P(empty | type 3.1)*
*= 0.14 * 0.37 * 0.18 * 0.31*
*= 0.00289.*

We are interested to find the most probable sequences by calculating the probabilities of every path. In the prototype, the most probable sequence is then displayed to the expert. His role is to validate or not the attribution of the categorial types to the lexical units. If the typification is correct, the process ends. Else, the second most probable sequence is proposed, and so on, until the expert accepts the distribution of the categorial types.

Finally, once validated, we update the transition matrix and the categorial dictionary the same way we described in the learning phase. Therefore, we dynamically improve the quality of the Markov matrix for the succeeding sentences as new ones are validated by the expert.

## 4. Evaluation

We tested 58 sentences of different syntactic structures with our prototype. Some of them were taken from the base corpus of 74 sentences. Please note that we maintain the list of the sentences of the corpus so we can test them at free will, but they will not serve once again for reinforcement.

Here are the outputs (translated in English) for 3 sentences entered through the prototype.

```
Sentence 1: “Jean se déplace souvent à pied”

Highest probability of sequence: 0.00000160267
Sequence: [Empty, N, N, (S\N)/N,
((S\N)/N)\((S\N)/N), N, N, Empty]
Is this sequence correct? Yes
```

```
Sentence 2: “Ava marche lentement et avec
beaucoup de grâce”

Highest probability of sequence: 0.00000773714
Sequence: [Empty, N, N, (S\N)\(S\N), &,
(((S\N)\(S\N))/N), (N/N), (N/N), N, Empty]
Is this sequence correct? No

2nd Highest probability of sequence: 0.00000773714
Sequence: [Empty, N, (S\N), (S\N)\(S\N), &,
(((S\N)\(S\N))/N), (N/N), (N/N), N, Empty]
Is this sequence correct? Yes
```

```
Sentence 3: “L’homme foule allègrement bitume et
trottoirs”

Highest probability of sequence: 0.00000639264
Sequence: [Empty, (N/N), N, ((S\N)/N), (((S\N)/N)\
((S\N)/N)), N, &, N, Empty]
Is this sequence correct? No

2nd Highest probability of sequence: 0.00000099255
Sequence: [Empty, (N/N), N, ((S\N)/N), (((S\N)/N)\
((S\N)/N)), N, &, (N\N), Empty]
Is this sequence correct? No

3rd Highest probability of sequence: 0.00000033746
Sequence: [Empty, (N/N), N, ((S\N)/N), (((S\N)/N)\
((S\N)/N)), N, *, (N\N), Empty]
Is this sequence correct? Yes
```

Below, *table 10* shows the results for the 58 sentences.

| Sequence number validated by the expert | Number of sentences validated at the sequence number | Percentage |
|---|---|---|
| **1** | **43** | **74.14%** |
| 2 | 6 | 10.34% |
| 3 | 6 | 10.34% |
| 4 | 1 | 1.72% |
| 5 | 1 | 1.72% |
| 9 | 1 | 1.72% |

**Table 10: Test results for 58 benchmark sentences**

We notice that almost all sentences (55 on 58; 94.82%) were properly typified within 3 attempts, of which 43 (74.14% of the 58 sentences) were correctly guessed on the first try (the most probable sequence) by the system. These results are very encouraging and demonstrate clearly the potential of our model, considering that the successful rate should improve as we train the system more.

## 5. Conclusion

The model we have presented in this paper is a work in progress and we are aware that many things can be done to improve it.

Our main concern involved the algorithm used for the calculation of the probabilities of the sequences, which is not efficient at all in terms of time and memory consuming. As a matter of fact, for a sentence of 10 lexical units having 5 potential categorial types each, $5^{10}$ (near 10 millions) probabilities would need to be calculated. Hopefully, there are strategies we can implement in order to reduce drastically the number of operations.

The first one would be the pruning of the categorial types in the categorial dictionary. More specifically, we could use the total frequency of all the categorial types of a given lexical unit after it has reached a certain number of occurrences and delete the entries for which the frequency is under a determined threshold. For example, we consider the French word "la" ("the") for which we suppose that the occurrences are distributed between the categorial types as in *table 11*.

| Lexical unit | Categorial type | Frequency |
|---|---|---|
| la | N | 4 |
| la | N/N | 195 |
| la | N\N | 1 |
| la | (S\N)/N | 0 |
| la | S\N | 0 |

**Table 11: Pruning of the categorial types of "avare"**

Given that the value of the threshold is 3, we would delete the last three entries of "la" (though the transition matrix would stay intact).

As a consequence, let us say we have 3 categorial types for each of the 10 lexical units instead of 5, the number of calculated sequences would fall from 10 million to 59 049.

Another strategy would be to use a Dijkstra-like algorithm in order to find very quickly the optimal sequence. As a replacement for searching for the shortest path, we would rather search for the "longest" path using multiplication instead of addition. Since the most probable sequence is the good one near 3 times of 4, we could first use this strategy, and then calculate the probabilities of all paths only if it is necessary.

Ultimately, we want to graft the model to a categorial analysis tool so it can be fully automatic and self-trained. In the context of analysis, if the syntactic correctness cannot be proved using the most probable sequence of categorial types, we could always test again the sentence with the sequence in second position, and so on, until it is validated or the probability falls below a given threshold.

## References

BALDRIDGE, J. AND KRUIJFF, G.J. (2003). "Multi-Modal Combinatory Categorial Grammar", in *Proceedings of EACL 2003*. 211-218.

BISKRI, I. AND BENSABER, B.A. (2008). "The Categorial Annotation of Coordination in Arabic", in *Proceedings of the Twenty-First International Florida Artificial Intelligence Research Society Conference*, 462-467.

DE CALMÈS, M. AND PÉRENNOU, G. (1998). "BDLEX : a Lexicon for Spoken and Written French", in *Proceedings of First International Conference on Langage Resources and Evaluation (LREC1998)*, Grenade, Paris, ELRA, 1129-1136.

CURRY, B.H. AND FEYS, R. (1958). *Combinatory Logic Vol. 1*, North-Holland.

DESCLÉS, J.P. AND BISKRI, I. (1996). "Logique Combinatoire et Linguistique: Grammaire Catégorielle Combinatoire Applicative", *Mathématiques et sciences humaines (No. 132),* 39-68.

LAMBEK, J. (1961). "On the Calculus Syntactic Types", in *Proceedings of Symposia in Applied Mathematics (Vol. XII)*, 166-178.

PERENNOU, G. (1986). "B.D.L.E.X: A data and cognition base of spoken French"*, IEEE International Conference on Acoustics, Speech, and Signal Processing'86*, 324-328.

SHAUMYAN S.K. (1998). Two Paradigms of Linguistics : The Semiotic Versus Non-Semiotic Paradigm. *Web Journal of Formal, Computational and Cognitive Linguistics*.

STEEDMAN, M. AND BALDRIDGE, J. (2009). "Combinatory categorial Grammar", R. Borsley and K. Borjars (eds) *Non-Transformational Syntax: A Guide toCurrent Models*, Blackwell, Oxford.