Handling Idioms in Symbolic Multilingual Natural Language Generation

Michaelle Dubé **François Lareau**
OLST, Université de Montréal
C.P. 6128 succ. Centre-Ville, Montréal QC, H3C 3J7, Canada
{michaelle.dube, francois.lareau}@umontreal.ca

Abstract

While idioms are usually very rigid in their expression, they sometimes allow a certain level of freedom in their usage, with modifiers or complements splitting them or being syntactically attached to internal nodes rather than to the root (e.g., *take something with a big grain of salt*). This means that they cannot always be handled as ready-made strings in rule-based natural language generation systems. Having access to the internal syntactic structure of an idiom allows for more subtle processing. We propose a way to enumerate all possible language-independent n-node trees and to map particular idioms of a language onto these generic syntactic patterns. Using this method, we integrate the idioms from the French Lexical Network (LN_{fr}) into GenDR, a multilingual realizer. Our implementation covers nearly 98% of LN_{fr}'s idioms with high precision, and can easily be extended or ported to other languages.

Keywords: idioms, multilingual natural language generation, lexicalization

1. Introduction

Idioms are notoriously difficult for natural language processing (NLP) (Sag et al., 2002; Constant et al., 2017). In this paper, we will focus on the task of rule-based natural language generation (NLG), and on the most prototypical type of idioms, which have namely been called *fixed expressions* by Sag et al. (2002) or *full idioms* by Mel'čuk (2012). To put it in a nutshell, such idioms can be defined as non-compositional multiword expressions (MWEs) where each word has been emptied of its meaning and rendered non-referential. They show a high degree of syntactic cohesion that typically forbids alteration. For example, UNDER THE WEATHER doesn't refer to any weather at all but means 'sick', and the noun WEATHER here cannot be modified without breaking the idiomatic interpretation of the whole. This is not to say that idioms cannot have complements or modifiers, but these are normally attached to the syntactic head of the phrase (here, UNDER), and they complement or modify the whole expression, not one of its internal words.

Because idioms behave somewhat like simple words from a syntactic point of view, they can very well be processed as ready-made blocks in symbolic NLG. For example, if a system is able to produce *Mary felt a bit sick that day*, it is trivial to replace the string `"sick"` with `"under the weather"` somewhere in the process and produce *Mary felt a bit under the weather that day*. Indeed, this has been the prevalent approach so far (cf. §2). However, there are several ways idioms can wreak havoc in an NLG system:

1. An idiom can be split by its modifier. For example, in French, DONNER SA LANGUE AU CHAT ('give up guessing', lit. 'give one's tongue to the cat'), when combined with ENCORE ('again') yields *donner encore sa langue au chat*.
2. Modifiers do not always attach to the syntactic head of an idiom. For example, to intensify TAKE (y) WITH A GRAIN OF SALT, you can modify the noun GRAIN instead of the head TAKE: *Take it with a big grain of salt*.
3. An idiom can be split by its complement, as in *You have to take whatever he says with a grain of salt*.
4. Complements do not always attach to the syntactic head of an idiom. This is particularly common with (but not exclusive to) idioms that contain body part words. For example, the second actant of PULL (y'S) LEG is expressed as a syntactic complement of the noun LEG, not as an object of the verb: *He's just pulling your leg*.
5. Inflection can be messy. This is especially true of nominal idioms in languages where agreement exists, because an inflected head triggers the inflection of internal determiners and adjectives. The problem is further exacerbated in languages with grammatical case. For example, in Lithuanian, LIETUVOS APELIACINIS TEISMAS ('court of appeal of Lithuania', lit. 'appellate court of Lithuania') has a nominal head TEISMAS 'court' with an adjective APELIACINIS 'appellate', and when the noun varies in case or number, so does the adjective. This requires access to individual words within the idiom (Dubinskaitė, 2017).
6. Idioms can sometimes "loosen up" and allow some syntactic freedom, with component words becoming referential, as in *It was a pretty big bullet to bite*, where BULLET acts as if it actually meant something like 'situation', although there is no such sense for that word in any other context.

Solving these problems elegantly in a symbolic NLG system requires access to the internal syntactic structure of idioms. In this paper, we propose a solution to represent that internal structure, which addresses the first five issues above. It is language-independent and designed for multilingual natural language generation

Proceedings of the 18th Workshop on Multiword Expressions (MWE 2022) n@LREC2022, pages 118–126
Marseille, 25 June 2022.

(MNLG), but it requires detailed lexical resources that we had only for French. Therefore, the discussion will draw from French data. As for the sixth issue, it has been explored in detail by Pausé (2017) from a theoretical point of view, but we have no elegant solution for it in the context of MNLG.

This paper is structured as follows. First, we will make a distinction between superficial and deep realizers in NLG and discuss briefly how idioms have been handled in existing systems (§2). Then, we will present the lexical data on which we rely and explain Pausé's (2017) idiom classification, which is central to our solution (§3). The main section will present our implementation (§4), which will be followed by an evaluation (§5) and a conclusion (§6).

2. Idioms in Linguistic Realizers

We should emphasize that in this paper we will only discuss the problem of idioms in rule-based realizers. Statistical and neuronal language models typically reproduce MWEs with relative ease, since they are very good at capturing recurrent patterns in a corpus. Yet, they are rambling machines that are very hard to harness. Thus, for many practical NLG applications where high precision and full control are needed, symbolic realizers are still the way to go.

While NLG refers to the whole pipeline from data collection to text delivery, realizers focus on the linguistic part of the process. Most realizers expect an input where both lexical choice and syntactic structure have already been computed, leaving the user with two particularly complex tasks. This is the case for FUF/SURGE (Elhadad, 1993; Elhadad and Robin, 1996), RealPro (Lavoie and Rambow, 1997; CoGenTex, 1998), SimpleNLG (Gatt and Reiter, 2009), its bilingual version, SimpleNLG-EnFr (Vaudry and Lapalme, 2013) and its Spanish version, SimpleNLG-ES (Ramos-Soto et al., 2017), JSReal (Daoust and Lapalme, 2015) and its bilingual version, JSRealB (Molins and Lapalme, 2015; Lapalme, 2020), as well as ATML3 (Weißgraeber and Madsack, 2017). KPML (Bateman, 1996) and OpenCCG (White, 2008) both start from a more abstract representation of the text's meaning, but they tend to focus on the grammar more than the lexicon, resulting in well-formed sentences that somehow lack lexical flexibility. The same goes for the bilingual (French/English) realizer FLAUBERT (Meunier and Danlos, 1998; Danlos, 2000)—not the be confused with the language model FlauBERT (Le et al., 2020). More recently, statistical approaches have been applied to text generation from logical forms (Basile, 2015) or semantic structures (Mille, 2014), but again lexical choice is rather rigid.

MARQUIS (Wanner et al., 2010) was a multilingual data-to-text generator used to produce air quality bulletins. It was designed to have a reusable text realization component that takes as input semantic representations, thus taking charge of lexical choice. Its lexicalization module was designed to produce natural-sounding collocations and to be as generic as possible (Lareau and Wanner, 2007; Wanner and Lareau, 2009). However, it handled full idioms as blocks of text, lacking the flexibility required for the cases discussed in §1. Its successor FORGe (Mille and Wanner, 2017) significantly improved the lexical coverage of MARQUIS, but also takes a rigid approach to idioms. Another successor, GenDR (Lareau et al., 2018) significantly expanded the range of collocation patterns it can handle (Lambrey and Lareau, 2015), but it also treats idioms as blocks with no internal structure.

To sum up, as far as we know, there is no generic, largish-scale deep realizer that takes idioms for what they are: premade phrases with internal syntactic structure. Hence, our goal is to incorporate such a functionality into a deep realizer. We picked GenDR for that purpose, because it already had a strong focus on non-trivial lexicalizations, in particular collocations. We will explain in this paper how we extended its lexicalization module to handle idioms in a way that reflects both their non-compositional semantic nature and their internal syntactic structure.

3. Lexical Data

We take our lexical data from the LN_{fr} (Polguère, 2009; Polguère, 2014; Ollinger and Polguère, 2020), a rich, open resource based on the principles of Explicative Combinatorial Lexicology (ECL) (Mel'čuk et al., 1995; Mel'čuk, 1995; Apresjan, 2000; Mel'čuk, 2006). Since GenDR itself is based on Meaning-Text Theory (MTT) (Žolkovskij and Mel'čuk, 1965; Kahane, 2003; Mel'čuk, 2016), an ECL-based resource was an obvious choice for our purposes. Each of LN_{fr}'s ~20k entries corresponds to a specific word sense that has its own lexicographic record with morphological, semantic and syntactic information, examples, and relations with other lexical units via lexical functions (LFs) (Wanner, 1996; Apresjan et al., 2002).

Our work is based on **Pausé's (2017) idiom classification**, in which she proposed linear syntactic patterns for French idioms. To avoid any confusion with our own generic patterns, we will henceforth use the term **linguistic patterns** to refer to them. These patterns are sequences of part of speech (POS) tags that represent each word of an idiom. For example, the idiom JOINDRE LES DEUX BOUTS ('make ends meet', lit. 'join the two ends') is assigned the pattern `V Det Num N`.

If necessary, function markers are used to distinguish patterns that have the same POS sequence but different syntactic structures. For example, while CRACHER DANS LA SOUPE ('bite the hand that feeds you', lit. 'spit in the soup') and BATTRE DE L'AILE ('be on the skids', lit. 'flap from the wing') both correspond to the sequence `V Prep Det N`, they don't have the same syntactic structure because the preposition is a circumstantial in the former but an oblique in the latter, so they have been assigned, respectively,

`V Prep.circ Det N` and `V Prep.obl Det N`. Obviously, this is tied to an underlying syntactic analysis, as determined by the lexicographers.

These patterns can further specify the syntactic position of an idiom's complements, which is useful when they do not attach to the syntactic head of the idiom, but to an arbitrary node within the idiom. For example, in PARLER DANS LE DOS (DE y) ('talk behind (y's) back'), the second complement of the idiom is expressed as a complement of the noun DOS ('back'), not as a complement of the head PARLER ('talk'). This is encoded as `V Prep Det N (Prep_$2)`, where `(Prep_$2)` refers to the second complement and its preposition.

Pausé's classification was incorporated into LN_{fr}, which contained 2919 idioms classified between 514 different patterns when we conducted our study. Table 1 gives the frequency of the most common patterns, with an example for each. Note the zipfian distribution, with only eight patterns accounting for half of the data.

Idiom pattern	**Example**	**#**	**%**
`N Prep N`	TÊTE DE MULE	409	14%
`N Adj`	TERRE FERME	377	13%
`Prep N`	DE JUSTESSE	222	8%
`N Prep.circ N`	CORPS À CORPS	138	5%
`Adj N`	JOLI CŒUR	100	4%
`Prep Det N`	DANS LE VENT	98	3%
`V Det N`	LEVER LE PIED	79	3%
`N Prep Det N`	ART DE LA TABLE	79	3%
Others	...	1417	49%
Total		**2919**	**100%**

Table 1: Most frequent idiom patterns in LN_{fr}

4. Implementation

As said in §2, we implemented our solution in the multilingual deep realizer GenDR (Lareau et al., 2018), which follows the principle of resource sharing across languages (Bateman et al., 2005). This realizer is built on top of the graph transducer MATE (Bohnet and Wanner, 2010). It is based on MTT and only handles the semantics-syntax interface: it takes as input a graph-based semantic representation (SemR) (Mel'čuk, 2012) and produces first an abstract dependency tree called a deep syntactic representation (DSyntR), from which it then derives a full-dependency tree called a surface syntactic representation (SSyntR) (Mel'čuk, 1988). A DSyntR is roughly similar to a Universal Dependency tree (de Marneffe et al., 2021), without functional words and with idioms collapsed into single nodes, while a SSyntR is analogous to a Surface-syntactic Universal Dependency (SUD) tree (Gerdes et al., 2018). Only the second transduction is relevant to us, since idioms are represented as single nodes in the DSyntR (thus, the SemR⇒DSyntR mapping is trivial), but as multiple nodes in the SSyntR. Figure 1 presents an input example (SemR) and a sample of possible outputs (SSyntR), in which the meaning 'courtiser' ('to court') can be lexicalized as the lexeme COURTISER or as the idiom FAIRE LA COUR (lit. 'do the court').

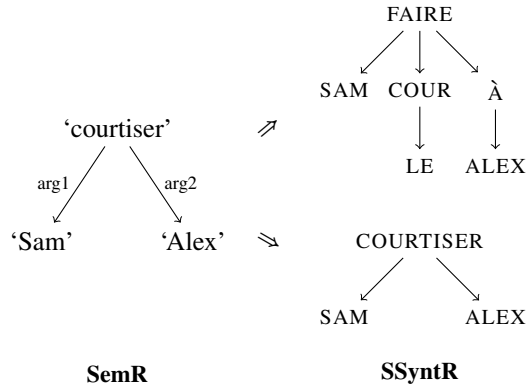

Figure 1: Alternative outputs for a simple SemR

4.1. Template Lexicalization Rules

The process of mapping a single DSyntR node onto multiple SSyntR nodes is called **template lexicalization** within GenDR (Lareau et al., 2018). Figure 2 is an example of such a rule for JOINDRE LES DEUX BOUTS ('make ends meet', lit. 'join the two ends').[1]

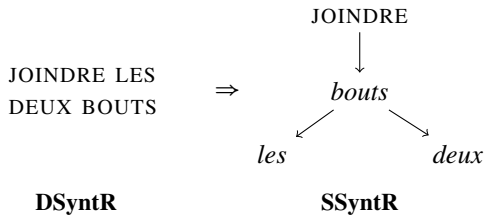

Figure 2: A simple template lexicalization rule

A full grammar would require rules like this one for each idiom in a given language. Obviously, a great number of these rules would resemble each other, thus the goal is to generalize them. Our solution is to create a set of template lexicalization rules that generalize Pausé's linguistic patterns (cf. §3). Each of these rules describes a generic pattern of syntactic tree with placeholders that are filled with lexical stock from our dictionary. The latter is derived from LN_{fr} and enhanced with our own data, as explained below. The idea behind these rules is to generalize linguistic patterns into more generic, language-independent patterns defined by the number of nodes in an idiom's subtree.

[1]Note that we omit relation names from our discussion. The choice of relation names is beyond the scope of our work, since they are language-specific. These decisions are thus left to the lexicographers working on LN_{fr}.

4.2. Generic Tree Patterns

We grouped idioms that had identical structures. For example, four-nodes idioms like JOINDRE LES DEUX BOUTS, ENFONCER UNE PORTE OUVERTE ('state the obvious', lit. 'kick an open door') and DANS DE BEAUX DRAPS ('in trouble', lit. 'in some nice bedsheets') have different linguistic patterns, but share the same structure, as illustrated in Figure 3.

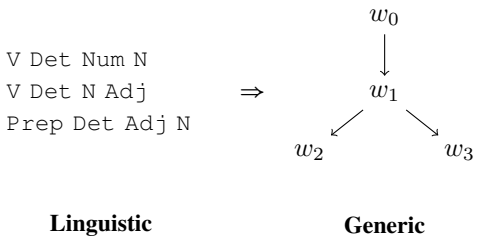

Figure 3: Linguistic patterns with the same structure

This pattern is not the only possibility for a four-node tree. Giving SSyntRs only represent hierarchy and not word order, two trees that differ solely by word order are equivalent. Therefore, there are four theoretically possible patterns for a four-node tree, to which we assigned IDs `4_01` to `4_04`, as shown in Figure 4.

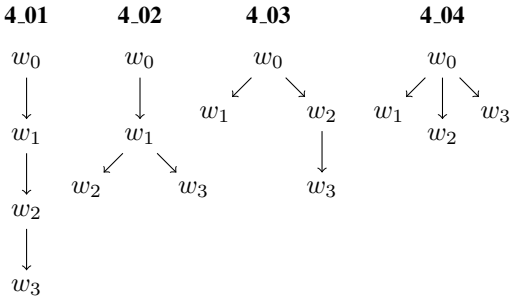

Figure 4: Generic patterns for a four-node tree

To systematically enumerate all possible generic patterns, we rely on number theory, which defines an integer's **partition** as its decomposition into a sum of positive integers (Andrews, 1998). For example, 3 has three different partitions: 3, 2 + 1, and 1 + 1 + 1. Each summand of the partition is called a **part**. The integer partitions thus correspond to all possible configurations of a root's (direct and indirect) dependents in a tree, with the number of parts in a partition corresponding to the number of direct dependents of the root.

To illustrate this, consider a three-node tree. It comprises a root and two dependents. The root's position is fixed, but there are two ways we can configure the other two nodes: either one depends on the root, and the other depends on the first (forming a chain), or both depend directly on the root. This corresponds to the two partitions of the integer 2: as 2 (a two-node subtree is attached to the root) or 1 + 1 (two single-node subtrees are attached to the root).

	Partitions	#Trees
2 nodes	1	1
		= 1
3 nodes	2	1
	1 + 1	1
		= 2
4 nodes	3	2
	2 + 1	1
	1 + 1 + 1	1
		= 4
5 nodes	4	4
	3 + 1	2
	2 + 2	1
	2 + 1 + 1	1
	1 + 1 + 1 + 1	1
		= 9
6 nodes	5	9
	4 + 1	4
	3 + 2	2
	3 + 1 + 1	2
	2 + 2 + 1	1
	2 + 1 + 1 + 1	1
	1 + 1 + 1 + 1 + 1	1
		= 20
7 nodes	6	20
	5 + 1	9
	4 + 2	4
	4 + 1 + 1	4
	3 + 3	3
	3 + 2 + 1	2
	3 + 1 + 1 + 1	2
	2 + 2 + 2	1
	2 + 2 + 1 + 1	1
	2 + 1 + 1 + 1 + 1	1
	1 + 1 + 1 + 1 + 1 + 1	1
		= 48

Table 2: Number of different n-node trees

As one can see, the enumeration of all node configurations for a tree of size n boils down to enumerating the partitions of the integer $n - 1$. Hence, the nodes of a four-node tree can be configured according to the partitions of 3, since its root has three (direct or indirect) dependents:

- a root linked to a three-node subtree (3);
- a root linked to a one-node subtree and a two-node subtree (2 + 1);
- a root linked to three one-node subtrees (1 + 1 + 1).

We have established above that a three-node subtree can be configured in two ways, thus yielding a total of four different configurations for a four-node tree. Now that we have computed the configurations for a four-node tree, we can compute those of a five-node tree, and so on, recursively. Table 2 gives the partitions and corresponding number of configurations for trees with up to seven nodes.

Let us pay special attention to the 3 + 3 partition for the

dependents of a seven-node tree, highlighted in Table 2. This partition is composed of two three-node subtrees. As seen previously, with $n = 3$, there are two possible trees for this part. Thus, one might expect to get 2 + 2 trees for a 3 + 3 partition. However, this is not the case. To demonstrate this, let us identify the two variants of a three-node tree as A and B. If you have two of them, the possible combinations are AA, AB, BA and BB. However, since our trees are unordered, AB and BA are actually identical. Therefore, there are only three possible configurations for a 3 + 3 partition.

4.3. Mapping Linguistic Patterns onto Generic Patterns

Since our generic patterns are essentially empty trees, they must be filled with lexical information specific to each language, which we retrieved from LN_{fr} in this case. Therefore, we need to map each linguistic pattern used for French idioms (cf. §3) onto one of our generic patterns. This pattern mapping involves establishing each idiom's SSyntR; therefore, it requires good knowledge of the formalism and high precision. Consequently, it was performed manually. For this purpose, we differentiated each generic pattern with a unique ID, as seen in Figure 4. In addition, we annotated each linguistic pattern with a word-to-node mapping code that describes the position in the tree of all the words of an idiom's pattern, as in Table 3.

For example, the four-node idiom JOINDRE LES DEUX BOUTS ('make ends meet', lit. 'join the two ends') follows the generic tree pattern `4_02`. Using its linguistic pattern `V Det Num N`, we established a mapping between the idioms' words and the nodes of the tree, which we express as a code: `0231`. Figure 5 illustrates the procedure we followed.

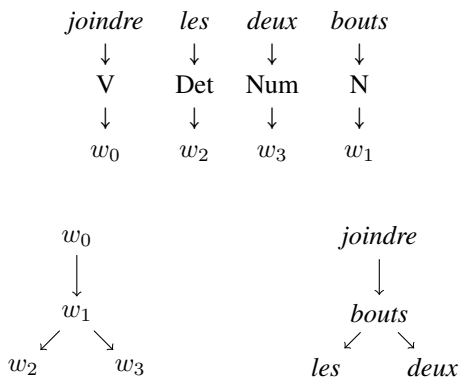

Figure 5: Example of an idiom's word mapping

The mapping code tells our realizer which word to assign to each of the tree's nodes during lexicalization. Each node in a tree is identified by an ID (w_0, w_1, w_2, etc.). Furthermore, we refer to an idiom's words by their linear order in the citation form. In sum, the mapping code takes this ordinal numbering and rearranges it according to the words' position in the tree.

Idiom pattern	Tree pattern	Word mapping
`Adj N`	`2`	`10`
`N Prep N`	`3_01`	`012`
`Prep Det N`	`3_01`	`021`
`V Det Num N`	`4_02`	`0231`

Table 3: Pattern mapping

Thus, in this case, `0231` means that w_0 will be filled by `"joindre"`, node w_2 by `"les"`, node w_3 by `"deux"` and node w_1 by `"bouts"`.

Following this procedure, we determined the mapping codes for each pair of patterns. This had to be done manually for each of the 514 linguistic patterns. Table 3 gives a few examples.

The mapping between specific idioms and one of Pausé's patterns is already given in the LN_{fr}, as this is part of the dictionary's structure. Hence, each of the 2919 French idioms are mapped onto one of 514 patterns. Table 4 presents some examples.

Idiom	Idiom pattern
JOLI CŒUR	`Adj N`
FEUILLE DE MATCH	`N Prep N`
DANS LE VENT	`Prep Det N`
JOINDRE LES DEUX BOUTS	`V Det Num N`

Table 4: Idiom pattern mappings from LN_{fr}

All this information was compiled into a dictionary format compatible with GenDR (Lareau and Lambrey, 2016). The process was relatively straightforward, except in the case of amalgams (such as *des=de+les*), as they are single words but correspond to two nodes in the SSyntR. Other forms that required special attention were reflexive pronouns, compounds and linguistic patterns containing embedded idioms.

5. Evaluation

The evaluation of our implementation focuses on the surface lexicalization of idioms in GenDR. The assessment is based on two criteria. First, we evaluate the coverage of the implementation, i.e., the percentage of LN_{fr}'s idioms that we can regenerate. Second, we evaluate the precision of the implementation, i.e., the proportion of generated structures that are correctly formed.

5.1. Coverage

The coverage of our implementation is measured by calculating the number of idioms that we process out of the total number of idioms associated with a linguistic pattern in LN_{fr}. Our dataset was composed of 2919 idioms from LN_{fr}, classified between 514 linguistic patterns (cf. §3). Most of the data (93%) were nominal (48%), prepositional (22%) and verbal (22%) idioms.

	Coverage	# total	%
Idioms	2846	2919	97,5%
Linguitic patterns	452	514	87,9%
Generic patterns	29	36	80,6%

Table 5: Coverage against LN_{fr}

Our implementation is currently limited to idioms of six words or fewer, which corresponds to a **97.5%** coverage of the idioms in LN_{fr}, i.e., 2846 idioms. As seen in Table 5, only 73 idioms (divided into 62 patterns) are not covered by our implementation and overall 29 of our 36 generic patterns are exploited by LN_{fr}'s idioms.

Table 6 lists the coverage of LN_{fr} idioms classified by POS. We notice a high coverage of all POS, except for clausal idioms (67%), such as CE N'EST PAS LA MER À BOIRE ('it's no big deal', lit. 'it is not the sea to drink'). This is due to their length, which tends to be greater than other idioms, thus often exceeding our limit of six.

POS	Coverage	# idioms	%
Nominal	1409	1414	99,6%
Prepositional	650	655	22%
Verbal	601	646	93%
Conjunctive	84	87	97%
Clausal	28	42	67%
Adjectival	35	35	100%
Adverbial	21	21	100%
Propositional	7	8	88%
Numeral	5	5	100%
Interjectional	4	4	100%
Pronominal	2	2	100%
Total	2856	2919	97,5%

Table 6: Coverage by part of speech against LN_{fr}

The decision to limit our coverage to six-node idioms was based on two factors. First, the number of possible trees (or generic patterns) grows exponentially with the number of nodes. Figure 6 shows the relationship between the number of trees and the frequency in LN_{fr} for idioms of different sizes. Notice that the number of possible trees quickly becomes higher than the number of idioms in the dictionary.

Secondly, we observe a recursion among the generic patterns. A tree being an intrinsically recursive structure, a subtree is itself a tree. Thus, we can compare the internal structure of idioms to Russian dolls, one embedded in the other. As a result, we can group an idiom's words into clusters that do not necessarily correspond to anything from a lexicological point of view, but that can operate as a string from a computational point of view. In other words, it is possible to describe a long idiom as a combination of smaller idioms corresponding to implemented generic patterns. For example, METTRE LES PETITS PLATS DANS LES GRANDS ('put on a big spread', lit. 'put the small dishes in the large') has seven nodes in its structure. In order to simplify this tree, we can reconfigure it into a three-node tree where *les petits plats* and *dans les grands* themselves form two three-node subtrees that depend on the root METTRE (Figure 7).

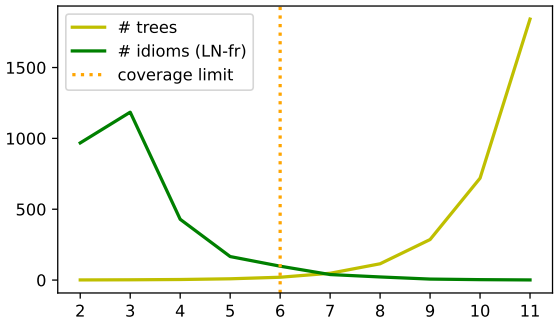

Figure 6: Number of trees vs. number of actual idioms in LN_{fr} (y-axis) with n nodes (x-axis), shown with our coverage cutoff

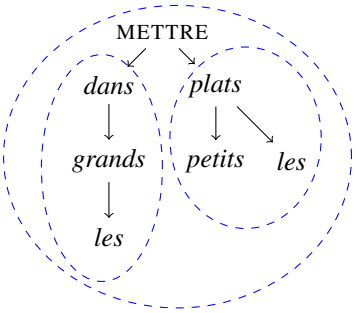

Figure 7: METTRE LES PETITS PLATS DANS LES GRANDS ('put on a big spread', lit. 'put the small dishes in the large') seen as a recursive structure

The verb (METTRE) is the only element of this idiom that can be freely inflected, hence the only one that needs to be isolated from the others. Note that this reconfiguration is not implemented for the moment. However, this solution will allow us to reuse our work as a design basis for processing longer idioms.

5.2. Precision

We automatically generated DSyntRs in MATE format for each of the 2846 idioms in LN_{fr}, together with placeholders for their complements. We then randomly selected three samples without overlap for human evaluation by two annotators: samples 1 and 2 each contained 100 structures (3.5% of the data) and were respectively evaluated by annotators A and B; sample 3 had 300 structures (10.6% of the data) and underwent double evaluation. The annotators were graduate students in linguistics with extensive training in GenDR and MTT. They used our grammar rules to process the DSyntRs and evaluated the resulting SSyntRs. An out-

Sample	n	Judge A	Judge B	κ
Sample 1	100	98 (98%)		
Sample 2	100		97 (97%)	
Sample 3	300	295 (98.3%)	294 (98%)	0.91

Table 7: Precision evaluation results

put structure was considered correct if all the nodes of the idiom were present and attached to the correct governor. Table 7 summarizes the number of correct outputs for each part of the evaluation.

Overall, **97.8%** of the SSyntRs had the expected configuration, with only 11 structures deemed problematic out of all 500. Cohen's κ (Cohen, 1960) was 0.91, which indicates near perfect inter-annotator agreement; only one structure was not agreed upon.

The problems we encountered stemmed from possibly too vague linguistic patterns (5), weaknesses in our implementation (4) and annotation errors (2).

The problems identified derive mainly from pattern mapping. As we have seen earlier, mapping requires a matching number of items on both patterns. Although LN_{fr}'s linguistic patterns are supposed to represent a single syntactic tree, some describe idioms containing a varying number of constituents.

The first explanation for this is embedded idioms. LN_{fr} does not specify the POS of idioms when they are embedded in another idiom. For example, `V Det N_Idiom` describes both MANGER LA FEUILLE DE MATCH ('fail to score a goal that should have resulted in victory', lit. 'eat the game sheet') and FAIRE LE JOLI CŒUR. Both FEUILLE DE MATCH and JOLI CŒUR are nominal idioms, but the former has three nodes and the latter only has two. The linguistic pattern might thus be too vague.

The second is our handling of amalgams (*du*, *des*, *aux*, etc.). The token *des* is ambiguous in French: it can be a determiner (the plural of UN 'a') or an amalgam of a preposition and a determiner (*des=de+les* 'of the'). Our handling of idioms failed to take this difference into account. This problem is also a consequence of node inflection. For example, the idiom ALLER AUX FRAISES ('make out in the bushes', lit. 'go to the strawberries') contains two inflected nodes (*aux* and *fraises*). The nodes in SSyntR are usually not word-forms; rather, they are lexemes with attached grammatical features specifying the desired inflection. This allows lexical information to be consolidated into entries corresponding to the lexical unit (FRAISE) rather than the word-form (*fraises*). Although the inflection of articles can be quickly processed, that of nouns or verbs would require going over each of the idiom entries.

6. Conclusion

Since idioms shows signs of form flexibility, it is crucial that their handling makes the isolation of specific nodes possible in order to enable the addition of inflections, complements or modifiers. We propose a creative solution to handling MWEs in MNLG, inspired from a generalization of Pausé's (2017) idiom classification. Our data were thus collected from the LN_{fr} (Polguère, 2009; Polguère, 2014; Ollinger and Polguère, 2020), an open resource for French.

We implemented our solution in the multilingual generic deep realizer GenDR (Lareau et al., 2018), which is built on top of the graph transducer MATE (Bohnet and Wanner, 2010). We automatically generated graph transduction rules for GenDR's template lexicalization. These rules were based on generic patterns that use integer partition to list all possible *n*-node trees. Our generic patterns are dependency syntactic trees with empty slots that will be completed with lexical data. We thus automatically generated a lexical dictionary encoding 452 linguistic pattern that describe the mapping of 2846 French idioms. We then manually mapped generic and linguistic patterns onto each other.

As a result, we covered 97.5% of the idioms in LN_{fr} excluding only idioms that contain seven lexemes or more. Our implementation also features a precision of 97.8% and a near perfect Cohen's κ of 0.91. The few problems identified stemmed from the pattern mapping caused by vague linguistic patterns and node inflection.

Many of the problems we encountered while implementing our solution originated from our very first decisions regarding our data collection from LN_{fr} data. If we were to start over, we would map each idiom's lexical units to their POS. This would allow us to design a script that fetches the POS describing embedded idioms. This solution would enable us to promptly encode the inflection of the idioms as grammemes and to subtract it from the lexemes' citation form in SSyntR.

Since our handling of idiom is based on data from the LN_{fr}, it would also gain in precision if idioms' lexicographic files systematically described their government pattern (to allow the addition of the proper preposition). Furthermore, these files could include information on the possible coreferences between the idiom's constituents and its actancts. Among other things, this would be relevant for idioms that include a determinative pronoun. For example, the file of the idiom AVALER SON CHAPEAU 'eat one's hat' (`V Det N`) could describe the coreference relationship between the idiom's `Det` and its first actant (X) : *X avala son chapeau* ('X ate his hat').

Our solution is language-agnostic but relies on complex lexical resources that are currently only available for French. The team behind LN_{fr} is also developing resources for English and Russian. Thus, we expect to be able to extend our solution to these languages fairly easily in the near future. Concretely, porting our grammar to a new language could easily be done without modification if the dictionary used to describe this language is in the same format as ours. All that would be required to do would be mapping language-specific idiom patterns to our generic patterns. This mapping

can be done by a trained linguist in a matter of days. Obviously, the hard part is to write the dictionary itself (years of work by a whole team of highly trained lexicographers), but this is independent of our implementation.

Appart from handling idioms in MNLG, one of the purposes of our system was to check the accuracy of lexical resources. Accessing the internal structure of idioms in order to regenerate them proved a good way of highlighting errors and inconsistencies in LN_{fr}. In particular, besides the occasional errors one would expect to find in a large lexical database, we found that the linguistic patterns used in LN_{fr} were not explicit enough with regards to the inflection of the words within an idiom. For example, compare the two synonyms À FOND LA CAISSE ('at full throttle', lit. 'all the way (with) the car') and À FOND LES MANETTES ('at full throttle', lit. 'all the way (with) the controls'). In the first case, *caisse* is singular, but in the second *manettes* is plural. This information was not captured by the linguistic patterns used in LN_{fr}.

Finally, our grammar design could be ported to other graph transducers, such as GREW (Bonfante et al., 2018), but this would require significant effort. However, a GREW implementation could be used to apply the rules in reverse, allowing the automatic construction of deep-syntactic corpora from existing surface-syntactic corpora in the SUD format (Gerdes et al., 2018).

7. Bibliographical References

Andrews, G. E. (1998). *The theory of partitions*. Cambridge university press, Cambridge, 2nd edition.

Apresjan, J. D., Boguslavsky, I. M., Iomdin, L. L., and Tsinman, L. L. (2002). Lexical functions in actual NLP applications. In *Computational Linguistics for the New Millennium: Divergence or Synergy? Festschrift in Honour of Peter Hellwig on the occasion of his 60th Birthday*, pages 55–72. Peter Lang, Frankfurt.

Apresjan, J. (2000). *Systematic Lexicography*. Oxford University Press, Oxford.

Basile, V. (2015). *From Logic to Language: Natural Language Generation from Logical Forms*. Ph.D. thesis, University of Groningen.

Bateman, J. A., Kruijff-Korbayová, I., and Kruijff, G.-J. (2005). Multilingual resource sharing across both related and unrelated languages: An implemented, open-source framework for practical natural language generation. *Research on Language and Computation*, 15:1–29.

Bateman, J. A. (1996). *KPML Development Environment*. GMD/Institut für Integrierte Publikations- und Informationssysteme, Darmstadt.

Bohnet, B. and Wanner, L. (2010). Open source graph transducer interpreter and grammar development environment. In *Proceedings of LREC'10*, pages 211–218, Malta.

Bonfante, G., Guillaume, B., and Perrier, G. (2018). *Application of Graph Rewriting to Natural Language Processing*. iSTE/Wiley, London/Hoboken.

CoGenTex, (1998). *RealPro: General English Grammar*. User manual.

Cohen, J. (1960). A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37–46.

Constant, M., Eryiğit, G., Monti, J., van der Plas, L., Ramisch, C., Rosner, M., and Todirascu, A. (2017). Survey: Multiword expression processing: A Survey. *Computational Linguistics*, 43(4):837–892.

Danlos, L. (2000). A lexicalized formalism for text generation inspired by tree adjoining grammar. In Anne Abeillé et al., editors, *Tree Adjoining Grammars: Formalisms, Linguistic Analysis, and Processing*, chapter 15. CSLI Publications, Stanford.

Daoust, N. and Lapalme, G. (2015). JSREAL: A text realizer for web programming. In Núria Gala, et al., editors, *Language Production, Cognition, and the Lexicon*, pages 361–376. Springer, Zürich.

de Marneffe, M.-C., Manning, C. D., Nivre, J., and Zeman, D. (2021). Universal dependencies. *Computational Linguistics*, 47(2):255–308.

Dubinskaitė, I. (2017). Développement de ressources lituaniennes pour un générateur automatique de texte multilingue. Master's thesis, Université Grenoble Alpes, Grenoble.

Elhadad, M. and Robin, J. (1996). An overview of SURGE: A reusable comprehensive syntactic realization component. In *Proceedings of INLG'96*, pages 1–4, Brighton.

Elhadad, M. (1993). FUF: the universal unifier. User manual version 5.2. Technical report, Computer Science, Ben Gurion University of the Negev, Beer Sheva, Israel.

Gatt, A. and Reiter, E. (2009). SimpleNLG: A realisation engine for practical applications. In *Proceedings of ENLG'09*, pages 90–93, Athens.

Gerdes, K., Guillaume, B., Kahane, S., and Perrier, G. (2018). SUD or Surface-Syntactic Universal Dependencies: An annotation scheme near-isomorphic to UD. In *Proceedings of the Universal Dependencies Workshop 2018*, Brussels, Belgium.

Kahane, S. (2003). The Meaning-Text Theory. In Vilmos Ágel, et al., editors, *Dependenz und Valenz: Ein internationales Handbuch der zeitgenössischen Forshung/ Dependency and Valency: An International Handbook of Contemporary Research*, volume 1, pages 546–570. Walter de Gruyter.

Lambrey, F. and Lareau, F. (2015). Le traitement des collocations en génération de texte multilingue. In *Actes de la 22e conférence sur le Traitement Automatique des Langues Naturelles (TALN)*, pages 579–585, Caen.

Lapalme, G. (2020). The jsRealB text realizer: Organization and use cases. arXiv:2012.15425v2 [cs.CL].

Lareau, F. and Lambrey, F. (2016). GÉCO. Technical report, OLST, Université de Montréal.

Lareau, F. and Wanner, L. (2007). Towards a generic multilingual dependency grammar for text generation. In *Proceedings of the GEAF07 Workshop*, pages 203–223, Stanford. CSLI Publications.

Lareau, F., Lambrey, F., Dubinskaitė, I., Galarreta-Piquette, D., and Nejat, M. (2018). GenDR: A generic deep realizer with complex lexicalization. In Nicoletta Calzolari, et al., editors, *Proceedings of LREC'18*, pages 3018–3025, Miyazaki.

Lavoie, B. and Rambow, O. (1997). A fast and portable realizer for text generation systems. In *Proceedings of ANLP'97*, pages 265–268, Washington.

Le, H., Vial, L., Frej, J., Segonne, V., Coavoux, M., Lecouteux, B., Allauzen, A., Crabbé, B., Besacier, L., and Schwab, D. (2020). FlauBERT: Unsupervised language model pre-training for French. In *Proceedings of LREC'20*, pages 2479–2490, Marseille, France.

Mel'čuk, I. A., Clas, A., and Polguère, A. (1995). *Introduction à la lexicologie explicative et combinatoire*. Duculot, Louvain-la-Neuve.

Mel'čuk, I. A. (1988). *Dependency syntax: theory and practice*. State University of New York Press, Albany.

Mel'čuk, I. A. (1995). The future of the lexicon in linguistic description and the explanatory combinatorial dictionary. In Ik-Hwan Lee, editor, *Linguistics in the morning calm*, volume 3. Hanshin, Seoul.

Mel'čuk, I. A. (2006). Explanatory combinatorial dictionary. In Giandomenico Sica, editor, *Open Problems in linguistics and lexicography*, pages 225–355. Polimetrica, Monza.

Mel'čuk, I. A. (2012). *Semantics: From Meaning to Text*, volume 1. John Benjamins, Amsterdam/ Philadelphia.

Mel'čuk, I. A. (2016). *Language: From Meaning to Text*. Ars Rossica, Moscow/Boston.

Mel'čuk, I. A. (2012). Phraseology in the language, in the dictionary, and in the computer. *The Yearbook of Phraseology*, 3:31–56.

Meunier, F. and Danlos, L. (1998). FLAUBERT: A user friendly system for multilingual text generation. In *Proceedings of INLG'98*, Niagara-on-the-Lake, Canada.

Mille, S. and Wanner, L. (2017). A demo of FORGe: the pompeu fabra open rule-based generator. In *Proceedings of INLG'17*, pages 245–246, Santiago de Compostela, Spain. Association for Computational Linguistics.

Mille, S. (2014). *Deep stochastic sentence generation: Resources and strategies*. Ph.D. thesis, Universitat Pompeu Fabra, Barcelona.

Molins, P. and Lapalme, G. (2015). JSrealB: A bilingual text realizer for web programming. In *Proceedings of ENGL'15*, pages 109–111, Brighton.

Ollinger, S. and Polguère, A. (2020). Distribution des systèmes lexicaux, ver. 2.0. Technical report, ATILF-CNRS, Nancy, France.

Pausé, M.-S. (2017). *Structure lexico-syntaxique des locutions du français et incidence sur leur combinatoire*. Ph.D. thesis, Université de Lorraine, Nancy.

Polguère, A. (2009). Lexical systems: graph models of natural language lexicons. *Language Resources and Evaluation*, 43(1):41–55.

Polguère, A. (2014). From writing dictionaries to weaving lexical networks. *International Journal of Lexicography*, 27(4):396–418.

Ramos-Soto, A., Janeiro-Gallardo, J., and Bugarín, A. (2017). Adapting SimpleNLG to Spanish. In *Proceedings of INLG'17*, pages 144–148, Santiago de Compostela.

Sag, I. A., Baldwin, T., Bond, F., Copestake, A., and Flickinger, D. (2002). Multiword expressions: A pain in the neck for NLP. In *Proceedings of the International Conference on Intelligent Text Processing and Computational Linguistics*, pages 1–15, Mexico.

Vaudry, P.-L. and Lapalme, G. (2013). Adapting SimpleNLG for bilingual English-French realisation. In *Proceedings of ENLG'13*, pages 183–187, Sofia.

Wanner, L. and Lareau, F. (2009). Applying the Meaning-Text Theory model to text synthesis with low- and middle-density languages in mind. In Sergei Nirenburg, editor, *Language Engineering for Lesser-Studied Languages*, volume 21 of *NATO Science for Peace and Security*. IOS Press, Amsterdam.

Wanner, L., Bohnet, B., Bouayad-Agha, N., Lareau, F., and Nicklaß, D. (2010). MARQUIS: Generation of user-tailored multilingual air quality bulletins. *Applied Artificial Intelligence*, 24(10):914–952.

Leo Wanner, editor. (1996). *Lexical functions in lexicography and natural language processing*, volume 31 of *Studies in language*. John Benjamins, Amsterdam/Philadelphia.

Weißgraeber, R. and Madsack, A. (2017). A working, non-trivial, topically indifferent NLG system for 17 languages. In *Proceedings of INLG'17*, pages 156–157, Santiago de Compostela.

White, M., (2008). *OpenCCG Realizer Manual*.

Žolkovskij, A. K. and Mel'čuk, I. A. (1965). O vozmožnom metode i instrumentax semantičeskogo sinteza. *Naučno-texničeskaja informacija*, 5:23–28.