Using Dependency Grammars in guiding templatic Natural Language Generation

Ariel Gutman[1]
relgu@google.com

Anton Ivanov
aii@google.com

Jessica Saba Ramírez
sabaramirez@google.com

Google

Working paper
Version of May 2022

Abstract

We propose a templatic Natural Language Generation system, which uses a dependency grammar together with feature structure unification to guide the generation process. Feature structures are unified across dependency arcs, licensing the selection of correct lexical forms. From a practical perspective, the system has numerous advantages, such as the possibility to easily mix static and dynamic content. From a theoretical point of view, the templates can be seen as *linguistic constructions*, of which the relevant grammar is specified in terms of dependency grammar. In this paper we present the architecture of the system, and two case studies: verbal agreement in French, including the object-agreement pattern of past participles, and definiteness spreading in Scandinavian languages. The latter case study also exemplifies how this framework can be used for cross-lingual comparison and generation.

Keywords: Templatic NLG, Universal Dependencies, Unification Grammar, cross-lingual NLG

[1] Corresponding author. Postal address: Ariel Gutman, Google, Brandschenkestrasse 110, 8002 Zürich, Switzerland

1. Introduction

In recent decades Dependency Grammars have become an important part of the NLP toolkit, especially in the domains of parsing and treebank constructions (Nivre 2006, Hajičová et al. 2010), as well as in the domain of text generation.

In this paper we describe a templatic NLG system which uses dependency grammar together with feature structure unification to guide the generation process. Generating texts with dependency grammars has been proposed, *inter alia,* within Mel'čuk's Meaning-Text Theory (cf. Kittredge & Mel'Cuk 1983, Lareau & Wanner 2007), while the combination of feature unification with dependency grammar has been proposed by Hellwig (1986; 2003).

The interest in our system lies in the fact that it did not start its life-cycle as a dependency-grammar based system. Rather, we already had at our disposal a templatic system which was serving multilingual responses to users in the production system of a vocal assistant. In this templatic system, each possible response corresponds to a particular template, which may contain both static and dynamic elements, the latter being also invocations of "sub-templates", which represent recurring phrases or sentences across multiple messages. This allows maximum flexibility in the authoring of the system's messages, as the template is not required to follow any grammatical formalism and can also contain static text. At the same time, the template author has full control over the generated output, thus ensuring high precision of the desired output, in contrast to statistical approaches. Yet being a "pure" templatic system, grammatical knowledge was not encoded in any systematic way in the system, requiring ad-hoc solutions for phenomena such as agreement or case assignment (cf. Reiter 1995).

We sought to enrich the system with a grammatical formalism which would enable scaling up within and across languages. At first, we explored using an HPSG system akin to Sag et al. (2003). However, this formalism did not match well with the existing templatic system. Conceptually, a constituency grammar like HPSG takes care of two different tasks: a) the linear arrangement of the words (lexemes) in the generated sentence; and b) the selection of the right surface forms of these words, which may vary due to agreement phenomena, case government or morpho-phonological sandhi. Yet in a templatic system the linear arrangement of words is already taken care of by the structure of the templates themselves. Thus, we found that requirement (b) can more easily and directly be taken care of by a dependency grammar formalism. At the same time, we found that feature structures amenable to unification remain a convenient representation of the linguistic constraints necessary for lexical selection, and can easily be combined with a dependency grammar framework.

In our renewed system, the existing templates can be augmented with dependency grammar annotations. This allows the author of the templates to leverage linguistic regularities to ensure grammaticality of the output. Thus, the spread of grammatical information, required for the correct inflection of the dynamic elements, is done by the dependency grammar component. The dependency annotations enable propagating grammatical features exactly where needed, and can also be used for long-distance dependencies, which are complicated to model in a pure constituency-based system, which transforms concepts into natural language using abstract rules alone. Our system thus combines the simplicity of templates together with the power of a grammatical framework (cf. van Deemter et al., 2005; McRoy et al., 2003). By using the Universal Dependencies framework (McDonald et al. 2013; de Marneffe et al. 2014) and a modular design, we can share most details of the grammatical specifications between languages (e.g. what features are propagated by different relations, how the type hierarchy is modeled, etc.), making only small adjustments per language to account for language-specific phenomena. A system similar to the one we present here has been deployed successfully for production usage.

The paper is organized as follows: In the following section we give an overview of the system architecture. Then, in the subsequent sections we describe two particularly interesting case studies, namely French verbal agreement and Scandinavian definiteness spreading. We conclude with some general remarks.

2. Materials and Methods

The NLG component described here is a part of a larger NLP stack. This stack analyzes user queries, extracts the query's intent, fetches the relevant data needed for a response, and then sends the NLG component the relevant intent with the response data. The intent is then mapped to a specific templatic response, which is realized according to the method described below.

2.1 Templatic Generation

At its core, the NLG system we describe is a templatic system: to express a given intent, a planner component selects a pre-authored template which consists of a sequence of *elements*. Some elements are simply static text, while others are dynamic: they may consist of a parametric slot, like a number, or a to-be-inflected lexeme (cf. Gutman et al. 2018), as well invocations of "subtemplates".[2] For instance, a message like “You have 3 new notifications” may be produced from a template like the following:

```
You have <number> new /NOTIFICATION/.
```

Here, `<number>` denotes an open numeric slot, and upper-case words between slashes like `/NOTIFICATION/` indicate lexemes. Other words are considered to be static content. To generate a correct sentence using this template, the system must populate the number slot with the correct data and inflect the `/NOTIFICATION/` lexeme accordingly.

2.2 Feature structure

Each element in the above template may be associated with a grammatical feature structure (Shieber 1985; Sag et al., 2003). To model English nominal agreement we only need a

[2] For simplicity, we abstract away from the usage of subtemplates in the following examples.

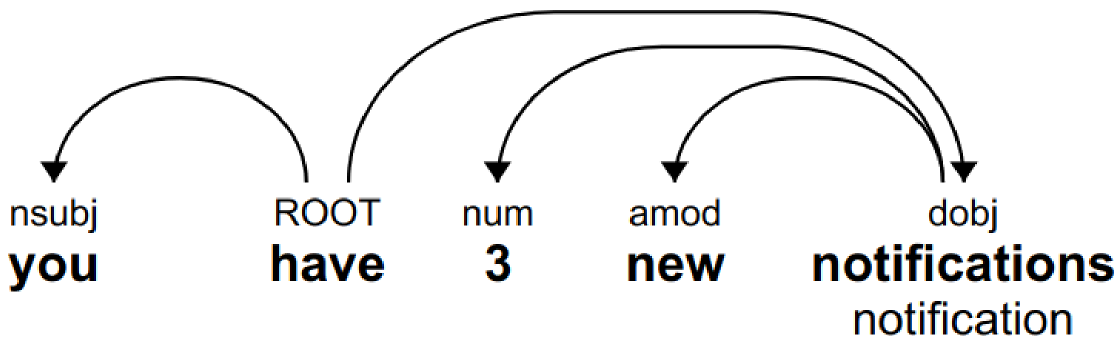
single feature, namely the NUMBER feature, but other features, such as GENDER, CASE, PERSON etc. are required in other circumstances. (By convention we mark features in upper-case.) Importantly, these features may be partly or completely unspecified. Elements which consist of parametric slots may derive these features from the input parameters.

In the above template, the slot `<number>` will have the feature NUMBER set according to the actual number used: 1 will be mapped to the value **singular** and other numbers to the value **plural** (we shall mark feature values in **bold**). The `/NOTIFICATION/` lexeme, on the other hand, will have its NUMBER feature **unspecified** in its uninflected form. Once the feature becomes specified, according to the method explained below, a specific inflected form of the lexeme can be chosen.

2.3 Dependency analysis

Each template is annotated with a dependency analysis by the template author, using a framework similar to the Universal Dependencies framework (McDonald et al. 2013; de Marneffe et al. 2014). The main advantage of using this framework is that it enables minimal changes in the dependency annotation across languages (especially typologically similar languages). More practically, we chose this specific framework as it was also used in the parsing component of the overall NLP stack used, allowing a single set of guidelines both for parsing and generation annotations.

In order to analyze the templates with this framework, one has to consider an example realization of the template (i.e. a sentence which may be produced by the template, in which all slots are filled and all lexemes are inflected). For instance, for the above template, the dependency structure is exemplified through the sentence “You have 3 new notifications”:



It is worth noting that the Universal Dependencies framework is based on the idea of semantic, and not syntactic, headedness (contrast this with the Surface-Syntactic Universal Dependencies framework, a.k.a. SUD, suggested by Gerdes et al., 2018). This means that content words, rather than function words, are generally chosen to be root nodes of sub-trees in the dependency analysis. Above, the root of the noun phrase “3 new notifications” is the noun “notifications” rather than the numeral *three*.[3] This makes sense in a framework used mostly for NLU purposes, but poses some difficulties in the NLG context: for instance, in a language such as Russian, where the case of a quantified NP is exposed

[3] It should be noted that even in the SUD the numeral would be considered a modifier, but one could take a more consistent approach in which the numeral would be the root, since it can be analyzed as the syntactic head in many languages.

on the numeral rather than on the noun (which is typically genitive). Another case of indirect assignment of verbal agreement features to the French auxiliary verb is discussed below. Notwithstanding these difficulties, we decided to use this framework for consistency with other components of our NLP stack, as mentioned above.
In more complex examples, where subtemplates are used, it is the root node of each subtemplate which represents the subtemplate grammatically in the context of the invoking template. For example, the templatic phrase <number> new /NOTIFICATION/ could be encapsulated as a subtemplate in which the lexeme /NOTIFICATION/ would serve as the root, thus marking the subtemplate as representing a noun phrase. In the super-template You have <notification-template> the *dobj* relation will have as its target the <notification-template>. Used thusly, the subtemplates may represent a subtree of the dependency tree of the complete message, and, if crafted appropriately, also a constituent phrase (in this example the object noun phrase). While this is often convenient, the system does not enforce the constituency structure, allowing the template author the flexibility of modeling subtemplates differently than any established constituency structure, the only constraint being that they represent contiguous text.

2.4 Unification across dependency edges

To resolve unspecified feature values, feature structures are unified among elements of the template.

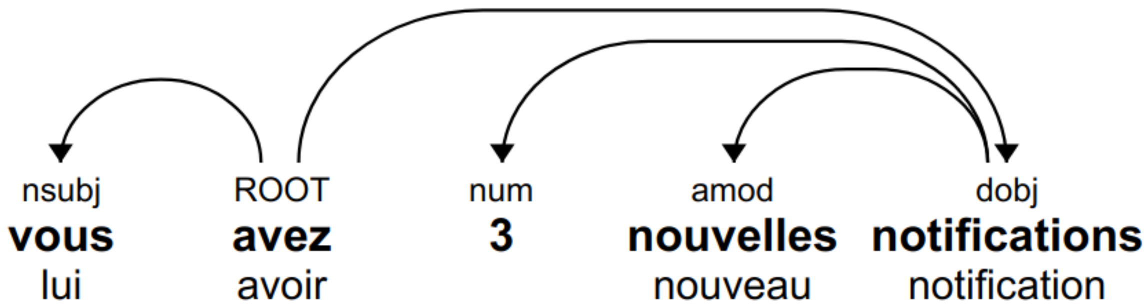
In the above example, <number> is the numeral modifier of /NOTIFICATION/ and thus is linked via a *num* relation, as can be seen in the analysis given above.

Each relation specifies which features should be unified across it. In the case of *num* (in English) only the NUMBER feature needs to be unified. As a consequence, the **unspecified** NUMBER feature is set to the appropriate value (in this case **plural**) and the correct form ("notifications") is selected.

This may seem like a quite complex setup to get simple number agreement, but let's consider the same example in another language, namely French. Here, we would like to produce a message like "*Vous avez 3 nouvelles notifications*". The corresponding template would be:

```
Vous avez <number> /NOUVEAU/ /NOTIFICATION/.
```

While the language is different, the syntactic structure, as reflected by the dependency analysis, is identical to the English one:



In French, nouns, numerals and adjectives must have the NUMBER and GENDER features specified in order to select an inflected form.[4] These are bundled together in one complex feature named AGR (for "agreement").

As before, the NUMBER feature of <number> is set according to the given number.[5] Assuming the number is "3" we get the following feature structures before unification:

3	AGR	[NUMBER **plural,**	GENDER **unspecified**]
/NOUVEAU/	AGR	[NUMBER **unspecified,**	GENDER **unspecified**]
/NOTIFICATION/	AGR	[NUMBER **unspecified,**	GENDER **feminine**]

Note that the gender feature is only specified initially for the /NOTIFICATION/ lexeme, given that this noun has a fixed gender. The various features are now unified across the *num* and *amod* relations (through the /NOTIFICATION/ node). The unification operation does not merely copy the information, but makes the features identical so that they refer to the same piece of information. Consequently, the order of these operations is irrelevant.[6] After unification, the new features are as follows:

3	AGR①	[NUMBER **plural,**	GENDER **feminine**]
/NOUVEAU/	AGR①	[NUMBER **plural,**	GENDER **feminine**]
/NOTIFICATION/	AGR①	[NUMBER **plural,**	GENDER **feminine**]

The circled ① indicates that the AGR features are unified with each other.

Once the feature structures are unified, the inflected forms of the lexemes can be selected and the sentence rendered correctly.

Different dependency arcs may specify unification of different sets of features, and also set features to predetermined values (e.g. set the case of a verbal subject to **nominative**). Formally, we associate each dependency relation type with a set of unification and

[4] For numerals, the gender is only apparent on numbers ending with the digit 1.
[5] The logic for French is different though. Numbers from 2 and above are assigned the feature **plural**, while the numbers 0 and 1 are assigned **singular**.
[6] For example, if the two **unspecified** NUMBER features are first unified through the *amod* relation, they remain unchanged, but now share the same information. When the numeral's NUMBER feature is unified through the *num* relation, both of these features become **plural.**

feature-setting operations, restricted according to the part-of-speech classes of the participating nodes. In the above example, we can define these rules as follows:

num (numeral, noun) -> Unify numeral.AGR with noun.AGR
amod (adjective, noun) -> Unify adjective.AGR with noun.AGR

In the current setup, these rules are written by hand. Given, however, that typologically-related languages have similar agreement and case-assignment patterns, these rules can be leveraged to be used in multiple languages, where only some details (such as the content of the AGR node) need to be adjusted per language (see Section 4).

2.5 Lexical constraints and markedness

Our system is also lexical, in the sense that specific constraints on the feature bundles can be specified for certain part-of-speech classes, or individual lexemes. For instance, in general, we assume that nouns have a third person feature, while only pronouns may vary this feature. This is expressed in a rule like the following (more examples will be given below):

noun -> Set the PERSON feature to **third**.

Each lexeme may have several inflected forms, all of which satisfy the appropriate constraints, but vary in the other features. One of these inflected forms may be seen as a default “unmarked” form, rather than being a positively marked form (Trubetzkoy 1931; 1939). To account for this, we use a type hierarchy of feature values (cf. Sag et al. 2003: 52). Given a possibly partially-filled feature structure for a certain lexeme, we can choose the surface realization which satisfies the most marked constraints. The choice of lexical forms, while taking into account lexical constraints and unmarked forms, will be illustrated in the case studies below.

3. Results

In order to demonstrate how our system can scale to different languages, we present two case studies, demonstrating its usefulness for different aspects of Romance (exemplified by French) and Scandinavian (Swedish and Danish) languages. We have not run a quantitative evaluation of the system; thus the case studies presented are qualitative in nature.

3.1. Case study 1: French verbal agreement

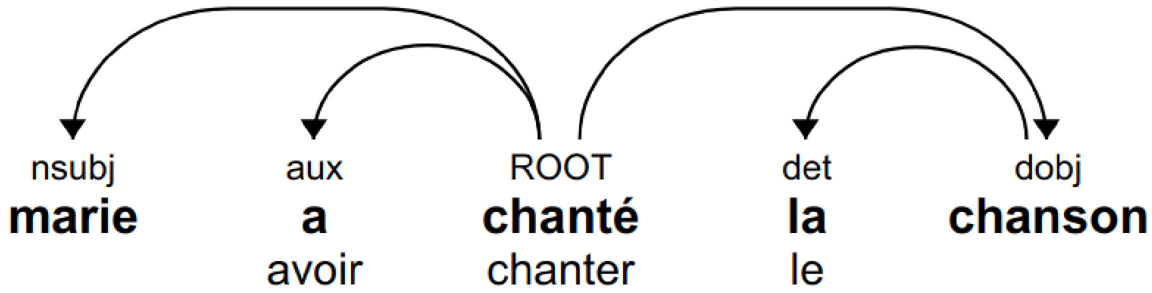
In French, as in other languages, verbal agreement depends on the PERSON and NUMBER features of the subject noun. When past participles are involved, the GENDER feature plays a role too, as we shall see. Verbs also govern the case of pronouns, for which we use a CASE feature, which can be set to **nominative** (e.g. *il*), **accusative** (*le*) or **dative** (*lui*).[7]

[7] Possessive pronouns can be analyzed as genitive case pronouns, but this is not relevant for our example.

A special challenge is posed by past-tense constructions which involve past participles, since these agree with the subject only when the auxiliary verb is the verb *être* (with some exceptions); normally, when the auxiliary is *avoir* they only agree with topicalized or pronominalized direct objects.

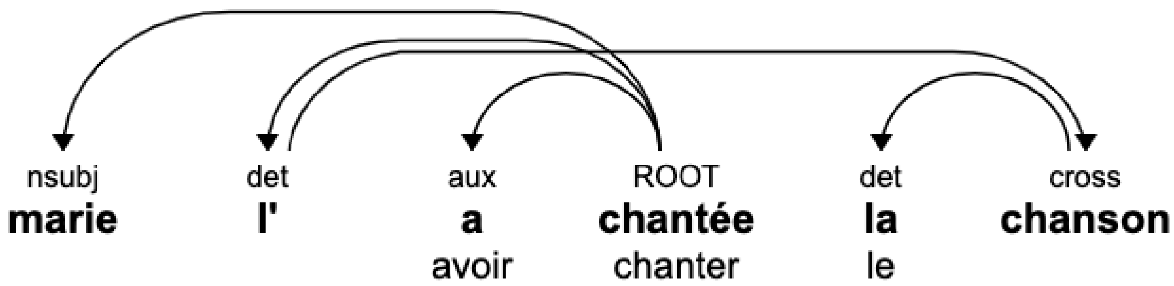
Let's consider the following two sentences involving the *avoir* auxiliary*,* together with their dependency analysis, and corresponding template. For convenience, we use English glosses together with labels of the Leipzig Glossing Rules (Comrie et al., 2008).

(1)	*Marie*	*a*	*chanté*	*la*	*chanson.*
	M.	have.3SG	sung.M.SG	the.F.SG	song
	"Marie sang a song."				



<name> /AVOIR/ <verb> /ARTICLE/ <noun>

(2)	*Marie*	*l'*	*a*	*chantée,*	*la*	*chanson.*
	M.	OBJ.3SG	have.3SG	sung.F.SG	the.F.SG	song
	"Marie sang it, the song."					



<name> /PRONOUN/ /AVOIR/ <verb> /ARTICLE/ <noun>

To generate these sentences, the planner needs to select a name, a transitive verb and an object noun. The inflection of the verb, pronoun, article and auxiliary would then follow. For simplicity, we assume here that the selection of the auxiliary's present tense is given by the template, as well as the usage of a participial form of the verb.

The Universal Dependencies framework presents a challenge here. Due to the semantic-headedness of this specific framework, agreement features of the subject need to be passed through the participle to the auxiliary.[8] Our solution is to use two “covert” agreement feature bundles, which we name SUBJ_AGR and OBJ_AGR. The “overt” agreement bundle (which guides the selection of the actual form) is simply called AGR. As finite verbs always agree with the subject and past participles sometimes agree with their object we posit the following lexical rules:

finite verb -> Unify AGR with SUBJ_AGR
past participle -> Unify AGR with OBJ_AGR

Since only pronouns can trigger object agreement with participles we posit also the following lexical rule:

pronoun -> Unify AGR with OBJ_AGR

Given these constraints, we need the following unification rules across dependency arcs to account for the verbal inflection patterns seen above (note that rules which apply to nouns, apply equally well to pronouns):

nsubj (noun, verb) ->
Unify noun.AGR with verb.SUBJ_AGR and set noun.CASE to **nominative**
dobj (noun, verb) ->
Unify noun.OBJ_AGR with verb.OBJ_AGR and set pronoun.CASE to **accusative**
aux (auxiliary, verb) -> Unify auxiliary.SUBJ_AGR with verb.SUBJ_AGR, and auxiliary.OBJ_AGR with verb.OBJ_AGR

To get the right form of the determiner and the object pronoun, we need two additional rules:[9]
det (determiner, noun) -> Unify noun.AGR with determiner.AGR
cross (pronoun, noun) -> Unify pronoun.AGR with noun.AGR

The *cross* relation, which models a cross-reference of a pronoun to another noun, is not a relation of the UD framework, but it is convenient to model it as such. For clarity, the referent noun is part of the example, but this is not a strict requirement: the system allows targets of relations to be "covert", i.e. not realized textually.

[8] If we were to use the SUD framework (Gerdes et al., 2018; see section 2.3 above) then the subject would stand in direct relation to the auxiliary verb. As explained there, for consistency reasons within our larger system we decided to use the semantic-centered UD framework, even though it adds some complexity in the feature transfer. In practice, however, where no direct object pronoun is present, we can treat the verbal phrase of auxiliary + participle as the root node, thus allowing direct feature transfer to the auxiliary verb.
[9] To account for the elided form of the feminine object pronoun *la*, realized as *l'*, the system keeps track of phonological features as well, and applies an implicit linear adjacency relation between adjacent tokens. For the sake of brevity, we don't give the details of this here.

To better understand how this system works in practice, let's examine the "flow" of the grammatical properties within the sentence:[10]

1. The AGR features of the slot `<name>` are set semantically according to the choice of the noun (in the example the name *"Marie"* is **third** person, **feminine**, **singular**).
2. These flow to the past participle's SUBJ_AGR feature bundle via the *nsubj* arc. The past participle's form is not affected, as it does not affect the overt AGR bundle.
3. From the participle, they flow to the auxiliary's SUBJ_AGR bundle. Being a finite verb, this bundle is unified with the overt AGR bundle, enabling the selection of the overt form of the verb.

In the first example, where no object pronoun is present, this accounts for the transfer of all agreement features (note that the *dobj* arc has no effect, since the object noun has no specified OBJ_AGR features). Clearly, the agreement features allow the selection of the verbal form of *avoir*, but how is the correct (masculine-singular) form of the past participle *chanté* selected, given that its AGR features are never specified? The lexemes in our system are stored as a set of grammatical features associated with inflected forms, very much like a traditional morphological paradigm. In contrast to a traditional paradigm, however, we can associate forms also with unspecified features. These *unmarked* forms will be selected if a more specific form cannot satisfy a given feature valuation. In practice, this means that for past participles in French we list 5 forms: the masculine-singular form is listed once together with its agreement features, and once as an unmarked form for which no features are specified.[11] Thus, this form is chosen when no object agreement features are specified.

Let's consider now the further flow of features in the second example, where an object pronoun is present.

4. The choice of the object `<noun>` sets its AGR features. In the case of *"chanson"* these are **third** person, **feminine**, **singular**.
5. These flow to the article's AGR features, through the *det* relation, so that the form *"la"* can be selected. Similarly, these features flow to the object pronoun's AGR features through the *cross* relation. Yet until the case of the pronoun is set, a surface form cannot be selected.
6. The lexical rule applied to the pronoun ensures that the pronoun's OBJ_AGR features are identical to its overt AGR features.
7. The *dobj* relation sets the case of the pronoun to **accusative**. Now the object form *"la"* can be selected. Since it appears before the vocalic onset of the verb *avoir* (which is known irrespective of the actual inflected form of the verb), the surface form is realized as *l'*.
8. At the same time, the *dobj* relation propagates the OBJ_AGR features to the past participle's OBJ_AGR features.

---

[10] The notion of "flow" here is really a metaphor, as the unification operation is order-agnostic. Yet it is useful to understand how the data is propagated in the system.

[11] In fact, one could dispense with the form associated with the **masculine** and **singular** features, as the unmarked form could anyhow be chosen for these features, but for clarity we keep the distinction between the two.

9. The lexical rule of past participles unifies the latter with the participle's overt AGR features, allowing the selection of the surface form "*chantée".*

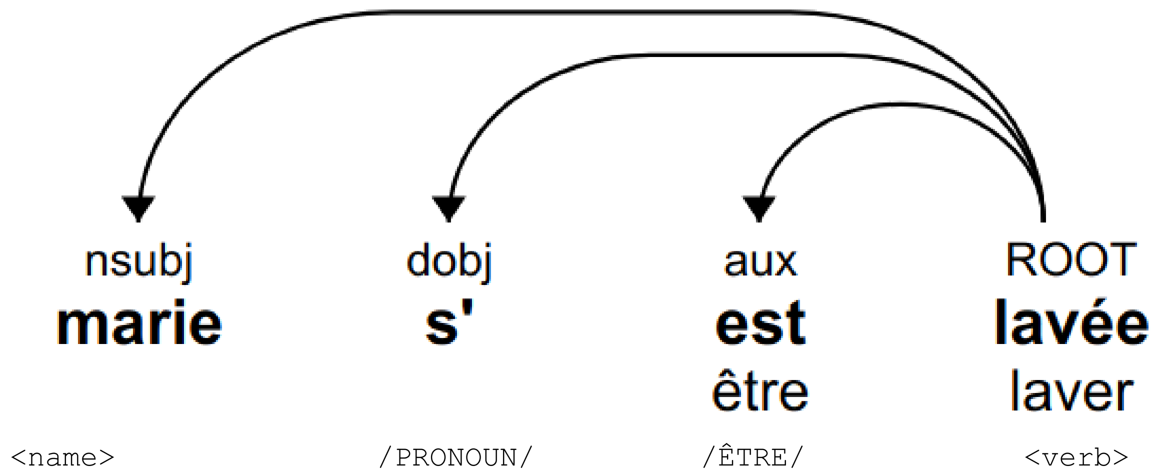
Finally, let's consider the case of French reflexive verbs, where the past auxiliary verb is the verb *être.* When this auxiliary verb is used (whether the verb is reflexive or not), the past participle does agree with the subject (and, if the verb is reflexive, also with the identical object).[12]

To account for such cases, we introduce a new lexical rule, applied only on the auxiliary verb *être:*

être -> Unify OBJ_AGR with SUBJ_AGR.

Now, let's consider an example using the *être* auxiliary:

(3) Marie s' est lavée.
M. REFL is.3SG washed.F.SG
"Marie washed herself."



In this case, the agreement features of the subject agreement features are similar to those outlined above (steps 1-3), allowing the selection of the inflected form *est*. From here, however, the flow is different:

4. The lexical rule above unifies the auxiliary's SUBJ_AGR features (being also its overt AGR features) with its OBJ_AGR features.
5. These are unified with the participle's OBJ_AGR features through the *aux* relation.
6. Due to the participle's lexical rule, these are unified with the participle's overt AGR features, allowing the selection of the inflected form *lavée.*
7. The *dobj* relation unifies the participle's OBJ_AGR features with the pronoun's OBJ_AGR features.
8. The lexical rule of the pronoun unifies the latter with the overt AGR features, allowing the selection of the third person pronoun *se*.

[12] There is in fact an exception to this rule: whenever the reflexive object represents an indirect (dative) object, the past participle does not inflect, such as in the sentence *Marie s'est fait des pâtes* ("Marie made herself pasta").

Note that in a simple architecture, supporting only a limited number of verbal predicates, one may assume that the choice of the auxiliary itself, as well as the reflexive nature of the pronoun, are specified in the template itself, or possibly through the use of specialized relations. Alternatively, it is possible to add more lexical features which would allow the verbal lexeme to select the right form of the auxiliary (*avoir* or *être*), as well as the reflexive form of the object pronoun. The latter is necessary if the system needs to scale up to support many different verbs. In such case, the selection of the right phrase structure of the verb phrase, depending *inter alia* on the lexical valency of the required verb, can be relegated to a specialized sub-template which is chosen according to the aforementioned valency.

3.2. Case study 2: Scandinavian definiteness spreading

In order to demonstrate the ease of use of this framework to model cross-lingual phenomena, we shall contrast the analysis of Swedish and Danish noun phrases. Both of these will be analyzed using the same dependency structure and unification rules, yet a subtle change in the lexical features of the two languages allows the system to account for the difference between them. This is somewhat similar to the multi-lingual approach presented by Lareau & Wanner (2007), in which grammatical rules were shared among similar languages.

In Swedish, as in Danish, nouns have two distinct forms, a short, unmarked form (e.g. *hus* "house"), and a suffixed form, marked as being definite (e.g. *huset* "the house"). Adjectives, too, show definite inflection, being historically a remnant of the weak Germanic declension, as in the contrast *stort* "big.INDEF" ~ *stora/store* "big.DEF". While indefinite articles are used whenever appropriate semantically, the definite article is typically only used when a definite noun is combined with an adjective. Otherwise, the suffixed definite form is used alone. Both articles and adjectives show as well gender and number agreement with the noun.
Consider the following examples:

	Swedish	Danish
Indefinite noun "a house"	(4) *Ett hus*	(9) *Et hus*
Indefinite noun "a house"	(5) *Ett stort hus*	(10) *Et stort hus*
Possessed & qualified noun "my big house"	(6) *Mitt stora hus*	(11) *Mit store hus*
Definite noun "the house"	(7) *Huset*	(12) *Huset*
Qualified definite noun "the big house"	(8) *Det stora* ***huset***	(13) *Det stora* ***hus***

As one can see, in Swedish the unmarked form *hus* is used both with the indefinite article (4)-(5) and the possessive article (6). The latter implies a definite reading, as is also clear from the definite form of the adjective. The marked definite form is used when a definite noun stands on its own (7), or together with a definite article and an adjective, marked definite as well (8). The latter phenomenon is termed in the literature as "definiteness spreading" or “double determination” (Dahl 2004). In Danish, the situation is similar except for the last case. Whenever a definite article is used, the unmarked form of the noun is used with it (13).

To model this, we add to the AGR bundle another bundle of features called DET, containing the features DEFINITENESS and DECLENSION (which can be **weak, strong** or **unspecified**). The marked definite form has the features DEFINITENESS **definite** and DECLENSION **strong** while the unmarked form has both features **unspecified**.

The dependency relations trigger the following three identical rules:
det/amod/poss (x, noun) -> Unify x.AGR+DET with noun.AGR+DET

The difference between Swedish and Danish can be accounted as a lexical difference. The Swedish definite determiner has the feature DECLENSION **strong** while the Danish one has the feature DECLENSION **weak**. This accounts for the difference in the selection of the marked vs. unmarked form of the head noun. Yet the feature DEFINITENESS **definite** is passed in both cases from the determiner to the adjective (through the noun), accounting for the adjective's definite form. The modular design of the grammar specification allows us to share the common grammar specification between the two languages, while storing the specifics (in this case the features and forms of the various determiners) in language-specific files.

4. Conclusions

In this paper we described how a dependency grammar framework can be used to guide a templatic NLG generation system. From an applied point of view, this type of grammar is convenient, as it allows partial specification of grammatical relations, as well as easy specification of long-distance relations. Thus, one can easily use it when combining static text together with dynamic elements. It is even possible to have some disjoint (partial) dependency trees in a given template. We have deployed this system in a real-world production system, generating millions of NLG utterances, using it in a combination with a rich lexicon of referential expressions (cf. Gutman et al. 2018).

From a theoretical linguistic perspective, we believe that the idea of feature unification across dependency arcs can provide a practical method of analysis of agreement and government phenomena. From a theoretical perspective, our templates may be seen as *constructions*, i.e. a linear ordering of segmental material together with open paradigmatic slots, tied to a specific meaning (Goldberg 1995; see also Gutman 2011:22). The use of dependency relations can provide a clear formalism of how agreement and case assignment is handled in such a framework.

From a typological perspective, since the specific dependency framework can be cross-lingual (as the Universal Dependencies framework is), our approach provides for interesting cross-lingual comparisons, as we have seen above. It is our hope that further research in this direction will bring both practical and theoretical advances in the field of NLG and beyond.

5. Author contributions

Conceptualization - A.G. and A.I.; Methodology - A.G. and J.K.; Software - A.G. and A.I.; Writing – Original Draft Preparation - A.G.; Writing – Review & Editing - A.G., A.I, and J.K.

6. Conflict of interest

The authors declare no conflict of interest.

7. References

Bernard Comrie, Martin Haspelmath, and Balthasar Bickel. 2008. *The Leipzig Glossing Rules: Conventions for interlinear morpheme-by-morpheme glosses (revised version of February 2008).* https://www.eva.mpg.de/lingua/resources/glossing-rules.php.

Östen Dahl. 2004. Definite articles in Scandinavian: Competing grammaticalization processes in standard and non-standard varieties. In Bernd Kortmann (ed.), *Dialectology meets typology: Dialect grammar from a cross-linguistic perspective*, Mouton de Gruyter, Berlin/New York.147-180.

Kees van Deemter, Emiel Krahmer, and Mariët Theune. 2005. Real versus Template-Based Natural Language Generation: A False Opposition? *Computational Linguistics* 31.1. 15–24.

Kim Gerdes, Bruno Guillaume , Sylvain Kahane , and Guy Perrier. 2018. SUD or Surface-Syntactic Universal Dependencies: An annotation scheme near-isomorphic to UD. *Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)*.

Adele Goldberg. 1995. *Constructions: A construction grammar approach to argument structure.* The University of Chicago Press, Chicago.

Ariel Gutman. 2018. *Attributive constructions in North-Eastern Neo-Aramaic*. Language Science Press, Berlin.

Ariel Gutman, Alexandros A. Chaaraoui, and Pascal Fleury. 2018. Crafting a lexicon of referential expressions for NLG applications. In: Ilan Kernerman and Simon Krek (eds.), *Proceedings of the LREC 2018 Workshop "Globalex 2018 – Lexicography & WordNets".*

Eva Hajicová, Anne Abeillé, Jan Hajič, Jiří Mírovský, and Zdeňka Urešová. 2010. Treebank Annotation. In Nitin Indurkhya and Fred J. Damerau (eds.), *Handbook of Natural Language Processing.* Second Edition. Chapman and Hall/CRC, Boca Raton. 67–188.

François Lareau and Leo Wanner. 2007. Towards a Generic Multilingual Dependency Grammar for Text Generation. In Tracy Holloway King and Emily M. Bander (eds.), *Proceedings of the GEAF 2007 Workshop.* CSLI Publications, Stanford. 203–223.

Peter Hellwig. 1986. Dependency unification grammar. In *Proceedings of the 11th International Conference on Computational Linguistics (COLING).*195–198.

Peter Hellwig. 2003. Dependency unification grammar. In Vilmos Agel, Ludwig M. Eichinger, Hans-Werner Eroms, Peter Hellwig, Hans Jürgen Heringer, and H. Lobin (eds.), *Dependency and Valency*, Walter de Gruyter, Berlin/New York, pp. 593–635.

Richard Kittredge and Igor A. Mel'Cuk. 1983. Towards a Computable Model of Meaning-Text Relations Within a Natural Sublanguage. In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence (IJCAI-83).* 657–659.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. 2013. Universal Dependency Annotation for Multilingual Parsing. In *Proceedings of ACL 2013*.

Susan W. McRoy, Songsak Channarukul, and Syed S. Ali. 2003. An augmented template-based approach to text realization. *Natural Language Engineering* 9(4). 381–420.

Marie-Catherine de Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D. Manning. 2014. Universal Stanford Dependencies: A cross-linguistic typology. In *Proceedings of LREC 2014*.

Joakim Nivre. 2006. *Inductive Dependency Parsing*. Springer, Dordrecht.

Ehud Reiter. 1995. NLG vs. templates. In P*roceedings of the Fifth European Workshop on Natural Language Generation*. Leiden, The Netherlands. 95–105.

Ivan A. Sag, Thomas Wasow, and Emily M. Bender. 2003. *Syntactic Theory: A Formal Introduction*. Second edition. CSLI publications, Stanford.

Stuart M. Shieber. 1985. Criteria for designing computer facilities for linguistic analysis. *Linguistics* 23(2). 189-212.

Nikolai Trubetzkoy. 1931. *Die phonologischen Systeme*. Travaux du Cercle Linguistique de Prague 4. 96-116.

Nikolai Trubetzkoy. 1939. *Grundzüge der Phonologie*. Vandenhoeck & Ruprecht, Göttingen.