Proceedings of the ESSLLI Student Session 2021

32nd European Summer School in Logic, Language and Information
July 26 – August 13, online

Mina Young Pedersen & Alexandra Pavlova
(Editors)

A conceptualization of language-specific metagrammatical classes in a multilingual project

Valeria Generalova[0000−0002−9963−0197]

Heinrich Heine University of Dusseldorf, Germany

1 Introduction

A metagrammar ([3]) is a formalized way of describing grammars[1]. Developed initially for Lexicalized tree-adjoining grammars (LTAG), it can be used for making representations originating in other theories. The main idea is that a metagrammar helps to describe more complex units building them from pre-defined smaller blocks. This saves large grammatical representations from redundancy since atomic parts of grammatical rules are encoded separately and get imported when needed.

Before the machine learning era, metagrammars have also been considered tools that accelerate the development of parsing and generating systems. Today it is clear that they outperform machine-learning approaches in quality, but not in speed due to still high involvement of people into the whole process.

With metagrammars, it becomes easier to add new phenomena to grammatical descriptions: it often suffices to add only a part of the new grammatical rule, while the rest is produced automatically through combinations of building blocks. [4, 189] illustrates this advantage of metagrammars with the following example: "if rules are generated for "transitive-passivewhExtractedByPhrase" (e.g., *By whom was the mouse eaten*), and if the hierarchy includes ditransitive verbs, then the automatic crossing of phenomena ensures that sentences will be generated for "ditransitive-passive-whExtractedByPhrase" (i.e., *By whom was Peter given a present*).

However, the addition of new phenomena to cover is only one axis of expanding the description. The other axis would be adding more languages and showing details of a particular grammatical phenomenon cross-linguistically.

This paper aims to present an architecture of a metagrammar that accounts clearly and concisely for typologically different languages. The goal of this architecture is to organize language features in a way facilitating access to language-specific information for other components of the metagrammar.

This research lies in the theoretical domain of grammar formalization (which in turn is part of the vast area of grammar engineering). This paper does not present any quantitative data, yet it is inspired by large typological databases.

[1] This work is part of the TreeGraSP project funded by the ERC Consolidator Grant awarded to Prof. Laura Kallmeyer.

Nevertheless, the stage of architecture configuration is necessary for every project. Thus, a study suggesting a design of common principles and data structures contributes to the development of future projects.

The paper is organized as follows. Section 2 shows the framework and some prior studies. Section 3 presents our solution. Section 4 offers a short comparison with a somewhat different project in the same direction, and Section 5 summarizes major claims.

2 Background

2.1 The XMG framework

Metagrammars can be created with different tools. In this study, the eXtensible MetaGrammar (XMG) framework ([5, 16]) is used.

This framework has some significant advantages ([5, 592-593]) that are summarized below. First, it offers a description language based on static declarative principles. Instead of formulating rules that apply one after another, XMG descriptions offer constituents and constraints that regulate their combinations. With the use of simple logical operations, conjunction and disjunction, XMG gets rid of the "ordering ant termination issues" often occurring in procedural approaches and offers "monotonic (no information removal combinations" as tree descriptions ([5, 597]).

Second, XMG follows some principles of logical programming (not limited to usage of conjunction and disjunction) to offer adequate representations of various linguistic dimensions (mainly syntax and semantics; see also [11] for morphology, [15] for semantic frames, [22] for multiword expressions).

Moreover, XMG offers an advanced system of "sharing information between distinct elementary fragments of grammar specification" ([5, 599]). The basic unit of an XMG metagrammar is a class, which can correspond to an entity of any level, from morpheme to sentence. A class can comprise one or several dimensions. Within each dimension, there are declared variables that have features assigned to them. The values of these features can be specified or defined as unification requirements (e.g. value of feature f_1 on variable v_1 is required to be equal to value of feature f_1 on variable v_2). Without going deep into how the mechanism of unification work, we emphasize its importance to constrain static metagrammatical descriptions.

Classes as basic units are organized in hierarchies. This is possible due to the inheritance mechanism that allows to borrow the content of one class and add to the description of another one. Conjunction and disjunction help fine-tuning the borrowing process to ensure the inheritance of necessary fragments only.

The inheritance mechanism contributes to the greater modularity of descriptions. For two classes being alike to even a small extent, it is possible to declare the common traits apart and only once in order to inherit them afterwards.

2.2 Multilingual XMG

The idea of creating a single metagrammar to describe several languages has been around for some time. However, how language information has been inserted in the description of linguistic phenomena has not been made explicit until recently.

An attempt to describe two languages, English and French, and explore two grammar formalisms, LFG (Lexical Functional Grammars) and TAG (Tree-Adjoining Grammars), is presented in [4]. For constraining the generation of syntactic trees, the authors introduce the notion of HyperTag ([4, 185, 190]). It helps to select constructions that are required for a given tree. A special feature in the HyperTag tells which of the two languages the given structure is applicable.

A more advanced multilingual project that uses XMG is presented in [14]. It covers five Germanic languages and Kashmiri and demonstrates a thorough implementation of early XMG principles. The paper keeps the HyperTag notion, but it is still not totally clear how exactly the sharing of features or components between languages is realized. The authors tell that the acceptable structures for each language are listed in a "head inventory, i.e., a list of possible heads for that language" ([14, 21]). But a list is not a metagrammatical structure; it is an additional type of data storage. Therefore, lists cannot inherit information from each other, share elements and features, assign values to features of XMG classes.

A recent study [12] suggests a type of XMG classes to encode language-specific information. The main suggestion of that paper is to separate abstractions about language phenomena (the so-called "Construction Classes") and values of features attested in each language. These latter types of classes are called Language Plugin, and we will reuse both the notion and the term.

According to [12, 44], each Language Plugin is a single variable to which a wide range of features is assigned. This single variable is referenced in other classes and enables their access to the values of the necessary features.

We find this design very helpful since it makes use of XMG advantages. In the present paper, we would like to discuss what features should be listed in Language Plugins and whether it is possible to build these classes more efficiently.

2.3 Initial structure of Language Plugins

In [12], there are two types of features assigned to the single variable in Language Plugins. The first type is Inventory Booleans. As the name indicates, these are boolean features that tell whether a construction is available in a language or not.

Introducing features of this kind in the metagrammar is advantageous because they swiftly filter out all the unacceptable rules for each given language. Their drawback, however, is the complexity of design. In order to formulate a range of boolean features to appear in a grammatical rule, one must perform a thorough linguistic (typological) study of each phenomenon which is intended to be described by the metagrammar.

Another downside of Inventory Booleans is not quite visible in [12]. The paper presents a description of constructions with morphological causatives derived from transitive verbs – a very narrowly defined range of constructions! For doing that, the authors introduced no less than five different boolean features. The pace of expansion of Language Plugins when wider rages of linguistic phenomena are covered is difficult to predict. Nevertheless, the volume of a single plugin can thus approach (or even exceed) the volume of a grammar, which can cause some issues during implementation. It would be nice to overcome this difficulty without losing the precision of definitions.

The second type of features suggested in [12] is called Morphological Features. These are categorical features whose values have to be specified manually. As the authors state, "the formalization work in this area consists of converting traditional grammar descriptions into typed features" ([12, 46]). As an example of this procedure, they provide conversion of a case system of a language into a set of features. Namely, they postulate features like "direct object case", "recipient case", etc.

The description of Morphological Features in [12] is very brief and does not really tell how exactly this data is collected, handled and formalized. The very goal of the present paper is to re-examine the architecture of the feature system within Language Plugins in order to overcome the drawbacks and the limitations of the solution presented in [12].

3 Further development of Language Plugins

As shown above, there are some issues in prior approaches to language-specific data in metagrammars. In our view, it is necessary to provide this information uniformly to ensure correct linking of Language Plugins to other parts of the metagrammar, and second, facilitate the orientation of human users in the data structure. We also find it necessary to keep the new classes in line with the overall trend towards high modularity proper to metagrammars in general and XMG projects in particular.

In this section, we present some suggestions for a better internal organization of Language Plugins.

3.1 Sources of features

Automatic extraction of features from large typological databases is much debated these last years ([25], [21], [20] *inter alia*). Although metagrammars have so far been developed to cover only some narrow fields, we would like to discuss the possibility of exporting features for language plugins massively from well-known typological resources for future studies. In this section, we would like to look at only two resources of different kinds, supposing that the overall idea is robust enough to hold for other databases.

The first resource is WALS [10]. To date, it offers 192 features from various areas of linguistics, and the data gets updated from time to time. The systems

of values for various features described in WALS are different depending on the authorship of the respective map. Nevertheless, it is possible to extract several patterns that are particularly useful (or not) for metagrammatical projects.

Many features describing morphological inventories (e.g., "Feature 22A: Inflectional Synthesis of the Verb" [2]) have values formulated as intervals ("2-3 categories per word"). What one would expect as values of this feature in the metagrammar would be not only the precise number of categories but also their names and, perhaps, the order of multiple categories where applicable.

This imprecision is unacceptable for a metagrammar, making these features impossible to include in the Language Plugin. The same concern arises with feature values formulated as a scale (e.g., "Feature 26A: Prefixing vs. Suffixing in Inflectional Morphology"; examples of values: "Strongly suffixing", "Weakly prefixing", "Equal prefixing and suffixing"; [9]).

There are also some inventory features in WALS that have boolean values (e.g., "Feature 67A: The Future Tense" with only two values: "Inflectional future exists" and "No inflectional future"; [6]). These features can be reused in the metagrammar and help to reject ungrammatical trees for languages having no phenomenon in question. But if a language has the phenomenon (i. e., when the feature value is positive), further information (e. g., how exactly the morpheme looks like) is required. Thus, purely binary WALS features are helpful when they are negative and appear to be incomplete when they are positive.

Feature value systems indicating the absence of a value and specifying it in case of presence also exist in WALS. An example would be "Feature 81A: Order of Subject, Object and Verb" ([8]). It specifies six basic orders along with a value formulated as "No dominant order". In other words, this single feature filters out languages to which the constraints on basic word order are not applicable, along with specifying the relevant constraints for those languages that have a dominant order.

This latter type of feature value system, on the one hand, seems to offer precise information about a linguistic phenomenon, and on the other hand, does it in a fairly concise form. The absence of a general feature that toggles off further options feels missing in [12]. If a phenomenon is not in the language, their solution requires specifying all the inventory booleans and setting their values to negative, which is too verbose, especially when a single plugin covers different constructions.

Not all typological databases present features and their values for each language. In this paper, we would like to explore The Universals Archive [18]. This database lists some features shared by a group of languages (not necessarily all) that do not "vary independently across the domain of all languages" ([17, 110]).

Information about language universals is a benefit for a metagrammar since it relates features to each other, which, in turn, helps to decrease the volume of individual plugins. Moreover, universals can help to predict some feature values in situations where actual data is lacking.

The database presents several logical types of universals (e.g., unconditional, target-source), most of them being various implications. In XMG, implications

are presented as disjunctions of conjunctions. Thus, each universal will be a conjunction of a boolean feature (name of the universal) and at least two other features (content of the universal). All of them must have positive values unless the opposite is stated.

Below we offer an illustration of this approach. We are going to explore Universal 71: "IF there are postpositions, THEN IF the adjective precedes the noun, THEN the genitive precedes the noun", see also the logical representation: "Postp ⇒ (A N ⇒ G N)" ([17], source: [13, 67]). This universal has two conditions. The first one ("if there are postpositions") restrains the range of languages to which the universal is applicable. In XMG terms, the class describing Universal 71 needs to unify only with those Language Plugins, where the value of the feature "has postpositions" is positive. The second condition ("IF the adjective precedes the noun") can be introduced differently. Either a Language Plugin has to contain a boolean feature "adjective precedes noun", or a categorical feature describing the word order must have the value "AdjN". Again, the XMG class describing the Universal 71 unifies only with those plugins, where the feature in question has the same value. Finally, what follows from the two conditions ("the genitive precedes the noun") is in metagrammatical terms the assertion of a value for a feature. Similar to the feature encoding the order of adjective and noun, this feature can be either boolean or categorical.

Language universals and statistical tendencies are a powerful instrument to predict feature values and avoid asserting many of them manually, relying instead on the inferences provided by the universals. The main issue in using The Language Universals archive (and similar databases) is the necessity to formalize the notation and re-describe most data formally compatible with XMG classes. This time-consuming task can nevertheless be profitable for various projects dealing with typological features and thus awaits its researcher.

3.2 Umbrella features

A closer look at possible sources of features for Language Plugins has shown that features must be organized within Language Plugins in a more complex structure than the plain list. Implications make some features relevant only under certain conditions. The boolean features telling whether the implication applies toggle several features. We suggest the term *umbrella* for features that are shortcuts for a set of others.

One can specify several features at once without theoretical implication, but specifying features for more complex phenomena. For example, in [12, 46], there are two independent features: "objectCase" and "subjectCase" [2] We find it possible to introduce the umbrella feature "alignment type" that would specify the value of the two main case features.

The introduction of umbrella features contributes to the elaboration of the Language Plugins' internal structure. It does not make descriptions less specific

[2] In [12], the authors use some terms from Role and Reference Grammar ([24], [23]): PSA (= priviledged syntactic argument) for subject and DO (= direct object) for object.

or less modular but creates a hierarchy of features, putting some regularities into separate small XMG classes. The resulting language descriptions inside the Plugin classes become shorter and better understandable for human readers. We find this last concern important as metagrammars typically are not used in machine-learning projects but appear in services, presupposing human involvement in development and usage.

3.3 The system of Language Plugins

Not only features inside Language Plugins can be organized as a hierarchy, but the whole Plugins can be grouped. The question of grouping languages is essential for any multilingual research: sometimes, it is helpful to follow the genetic classification; in other cases, putting together languages sharing some typological properties makes more sense. Each method requires decisions that may seem arbitrary and lead to some counter-intuitive results.

As stated earlier, the XMG framework offers mechanisms to inherit content from one class to another. Moreover, one can import the content only partly and add some new information to the target class. We suggest employing this mechanism to create a system of language plugins.

Once one has the description of a language given as a feature structure, one can mathematically calculate the similarity between language descriptions. Moreover, it is possible to automatically extract common parts, store them separately, and import them to other Plugins.

The grouping principle will then rely on quantitative similarity: the more umbrella features the languages share, the larger common part they have, the more likely this common part is to be inherited by one of these languages from the other one (or by both these languages from a common source). Importantly, languages do not necessarily need to share the values of the common features since feature complexes are specified in separate modules. Presumably, features like “alignment type”, “basic word order”, “locus of makring”, and alike will constitute the part common to all language plugins. In contrast, features like “has postpositions” will appear only in a subset of plugins.

Not only does this approach introduce more structure to the whole system, but it also keeps the inherited parts intact each time they are used and saves the metagrammatical descriptions from (some) errors. Creating language groups based on the number of common features can be set as a goal of a separate project by itself. Not only does it verify how well language plugins have been created, but also its result could bring new insights regarding linguistic typology.

4 Comparison with LinGO Grammar Matrix

In previous sections, we reviewed some projects exploiting the XMG framework and compared our solution to them. However, it is impossible to talk about a grammar engineering solution without referring to the LinGO Grammar Matrix project, running since 2002 at the University of Washington [1]. It uses HPSG

(Head-driven phrase structure grammar, [19]), where features have more importance not only in the implementation but also in the theoretical part than in LTAG and XMG.

The module of LinGO Grammar Matrix that resembles Language Plugins the most is the customization system. It is nourished by libraries that are being designed separately within smaller projects affiliated with the LinGO Grammar Matrix. Libraries suggest web-based questionnaires that are afterwards given to users interested in covering their language. Questionnaires vary from topic to topic but always elicit lexical and grammatical information. The features used in this questionnaire are in the vast majority categorical; the value 'none' is sometimes just one of more than two options. An example of such choices for typologically varied languages can be found in [7, 208-245]. The list of choices for each language concerning the topic case and marking (direct or invert) together with the description of lexical items necessary for processing test sentences contains around 100 features. It is not complete as not all features are called within each language, but all necessary information can be found easily.

Even though it is difficult to concur with such a developed system, we nevertheless wish to point out some differences that might make our architecture look appealing. First of all, our idea of separate blocks for different classes of construction and inheritance between languages could save end users time in determining the values of the features and also reduce the size of the questionnaire (= plugin). Also, the elicitation of language-level features will not be required in each specific module since they would become available for all Construction Classes when introduced at least once. The major advantage of the LinGO Grammar Matrix that we have nothing to oppose is its very advanced stage of development, while our solution is only a proof of concept.

5 Conclusion

In this paper, we have presented a development XMG classes, first introduced in [12] and called Language Plugins. We have offered an idea of how to ensure a sustainable extension of their coverage by splitting the multitude of functions into blocks, the presence of which in the target plugin is regulated by the single boolean feature. We also have indicated some directions in which WALS and The Universals Archive can be explored to provide a tighter link between features and facilitate the creation of new plugins. Although our presentation of related work is very brief, we have shown that the novel solution overcomes prior XMG-based solutions and is to an extent comparable to the LinGO Grammar Matrix.

Further directions of our research concern transforming this proof of concept into a working product. This would require creating a typologically salient sample of languages and language phenomena to develop the plugins according to our architecture. Automatic extraction of features from typological resources is a separate task that is a great help in creating the final product.

References

1. Bender, E.M., Flickinger, D., Oepen, S.: The grammar matrix: An open-source starter-kit for the rapid development of cross-linguistically consistent broad-coverage precision grammars. In: Carroll, J., Oostdijk, N., Sutcliffe, R. (eds.) Proceedings of the Workshop on Grammar Engineering and Evaluation at the 19th International Conference on Computational Linguistics. pp. 8–14. Taipei, Taiwan (2002)
2. Bickel, B., Nichols, J.: Inflectional synthesis of the verb. In: Dryer, M.S., Haspelmath, M. (eds.) The World Atlas of Language Structures Online. Max Planck Institute for Evolutionary Anthropology, Leipzig (2013), https://wals.info/chapter/22
3. Candito, M.: A principle-based hierarchical representation of ltags. In: COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics (1996)
4. Clément, L., Kinyon, A.: Generating parallel multilingual lfg-tag grammars from a metagrammar. In: Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics. pp. 184–191 (2003)
5. Crabbé, B., Duchier, D., Gardent, C., Roux, J.L., Parmentier, Y.: Xmg: extensible metagrammar. Computational Linguistics **39**(3), 591–629 (2013)
6. Östen Dahl, Velupillai, V.: The future tense. In: Dryer, M.S., Haspelmath, M. (eds.) The World Atlas of Language Structures Online. Max Planck Institute for Evolutionary Anthropology, Leipzig (2013), https://wals.info/chapter/67
7. Drellishak, S.: Widespread but not universal: Improving the typological coverage of the Grammar Matrix. Ph.D. thesis, University of Washington (2009)
8. Dryer, M.S.: Order of subject, object and verb. In: Dryer, M.S., Haspelmath, M. (eds.) The World Atlas of Language Structures Online. Max Planck Institute for Evolutionary Anthropology, Leipzig (2013), https://wals.info/chapter/81
9. Dryer, M.S.: Prefixing vs. suffixing in inflectional morphology. In: Dryer, M.S., Haspelmath, M. (eds.) The World Atlas of Language Structures Online. Max Planck Institute for Evolutionary Anthropology, Leipzig (2013), https://wals.info/chapter/26
10. Dryer, M.S., Haspelmath, M. (eds.): WALS Online. Max Planck Institute for Evolutionary Anthropology, Leipzig (2013), https://wals.info/
11. Duchier, D., Ekoukou, B.M., Parmentier, Y., Petitjean, S., Schang, E., et al.: Describing morphologically-rich languages using metagrammars: a look at verbs in ikota. Language Technology for Normalisation of Less-Resourced Languages pp. 55–60 (2012)
12. Generalova, V., Petitjean, S.: A prototype of a metagrammar describing three-argument constructions with a morphological causative. Typology of Morphosyntactic Parameters **3**(2), 29–51 (2020)
13. Hawkins, J.A.: Word Order Universals. Academic Press, New York (1983)
14. Kinyon, A., Rambow, O., Scheffler, T., Yoon, S., Joshi, A.: The metagrammar goes multilingual: A cross-linguistic look at the v2-phenomenon. In: Proceedings of the Eighth International Workshop on Tree Adjoining Grammar and Related Formalisms. pp. 17–24 (2006)
15. Lichte, T., Petitjean, S.: Implementing semantic frames as typed feature structures with xmg. Journal of Language Modelling **3**(1), 185–228 (2015)
16. Petitjean, S., Duchier, D., Parmentier, Y.: Xmg 2: describing description languages. In: International Conference on Logical Aspects of Computational Linguistics. pp. 255–272. Springer (2016)

17. Plank, F., Filimonova, E.: The universals archive: brief introduction for prospective users. STUF - Language Typology and Universals **53**(1), 109–123 (2000), https://typo.uni-konstanz.de/rara/archive-intro/
18. Plank, F., Mayer, T., Mayorava, T., Filimonova, E.: The universals archive (May 2020), https://typo.uni-konstanz.de/rara/category/universals-archive/
19. Pollard, C., Sag, I.A.: Head-driven phrase structure grammar. University of Chicago Press (1994)
20. Ponti, E.M., O'horan, H., Berzak, Y., Vulić, I., Reichart, R., Poibeau, T., Shutova, E., Korhonen, A.: Modeling language variation and universals: A survey on typological linguistics for natural language processing. Computational Linguistics **45**(3), 559–601 (2019)
21. Rama, T., Wichmann, S.: Towards unsupervised extraction of linguistic typological features from language descriptions. In: Proceedings of the First Workshop on Typology for Polyglot NLP (2019)
22. Savary, A., Petitjean, S., Lichte, T., Kallmeyer, L., Waszczuk, J.: Lexical encoding of multiword expressions in xmg. In: Poibeau, T., Parmentier, Y., Schang, E. (eds.) Actes des 2èmes journées scientifiquesdu Groupement de Recherche Linguistique Informatique Formelle et de Terrain (LIFT). pp. 59–62 (2020)
23. Van Valin, Jr., R.D.: Exploring the syntax-semantics interface. Cambridge University Press (2005)
24. Van Valin, Jr., R.D., LaPolla, R.J.: Syntax: Structure, meaning, and function. Cambridge University Press (1997)
25. Virk, S.M., Borin, L., Saxena, A., Hammarström, H.: Automatic extraction of typological linguistic features from descriptive grammars. In: International Conference on Text, Speech, and Dialogue. pp. 111–119. Springer (2017)