Plugins for structurally varied languages in XMG

1 Introduction

Accounting for structurally varied languages in a multilingual grammar engineering project is a challenging task. Most projects agree on separating language information from cross-linguistic generalizations. Nevertheless, the exact way how parameters of individual languages are stored depends on the framework and the general architecture.

The present paper grounds on the architecture suggested in Generalova and Petitjean, 2020 and uses the XMG (eXtensible MetaGrammar, Crabbé et al., 2013; Petitjean et al., 2016) framework. It presents the design of classes responsible for storing linguistic information (called Language Plugins in Generalova and Petitjean, 2020).

The present paper is not a ready-to-use piece of code for performing parsing or any other task since Language Plugins alone are insufficient for describing any linguistic phenomena. This paper aims to suggest an XMG-based designed of such classes and estimate how many languages and features can be supported after reusing the information from WALS. Further applications of this design will strongly depend on the goals and limitations of projects willing to use this design.

2 Background

The project by Generalova and Petitjean, 2020 presents the grammar architecture featuring two main kinds of XMG classes: Construction Classes and Language Plugins. The former describes primarily syntactic and semantic dimensions, capturing properties of constructions that can be generalized within or across languages. The essential is that the values of all features are specified not in Construction Classes but in Language Plugins. Construction Classes only reference them.

The correct accessibility by Construction Classes is the critical requirement for Language Plugins. This determines their design: each Language Plugin is a single variable to which features are assigned.

The first type of features suggested in Generalova and Petitjean, 2020 is Inventory Booleans. These are boolean features that tell whether a construction is available in a language. The advantage of Inventory Booleans is that they filter out the unacceptable rules for each given language quickly. However, they are complicated to design since one must perform a thorough typological study before describing each specific language phenomenon. One of the present paper's goals is to suggest reusing knowledge from WALS for designing these features.

The second type is called Morphological Features. Since the main sources for extracting morphological features are descriptions of individual languages but not typological resources, they are not examined in much detail in this paper.

The study by Generalova and Petitjean, 2020 seems to overlook one more important class of features, i. e. language-level features like word order, locus of marking, etc. Although these parameters vary across languages, they usually bear the same values in all types of sentences within each particular language and determine the choice of basic fragments for building construction classes. The main objection is that genuine universal features are found very rarely. This paper suggests a way for combining knowledge about general trends of a language and the appearance of particular constructions.

3 Suggestions

Our principal suggestion is the hierarchy of features. We postulate language-level boolean Umbrella Features that would tell whether a phenomenon exists in a given language. For example, whether a language has future tense or basic word order. We use the term "umbrella" because these features import other boolean features (Inventory

Booleans) or categorical features (e. g., the basic word order pattern) into the Plugin once they bear positive values. Keeping generalized features separately from feature specifications would allow for optimal combination of Language Plugins without increasing their length unnecessarily. The XMG language allows for such architecture since it has a developed class inheritance system and supports logical conjunction and disjunction. More technical details are omitted in this abstract but will constitute a significant part of the talk.

In the talk, we will show which combinations of boolean and categorical features appear optimal for describing various linguistic phenomena. We also discuss the applicability of other feature types. Finally, we sketch a "must-have" list of features for all Language Plugins and model some of its modifications. The suggested approach also proves helpful if language data is insufficient, namely, when the value of only one sub-feature is known. In the talk, this topic will receive due attention.

4 WALS analysis

To date, WALS (Dryer and Haspelmath, 2013) offers 192 features from various areas of linguistics. For many of them, the range of values contains one value telling that the title phenomenon does not exist in a given language and some other values presenting the encountered varieties. An example is "Feature 81A: Order of Subject, Object and Verb", where, in addition to the six fundamental word orders, the value "No dominant order" stands. This WALS feature could be reused in a Language Plugin in the form of two different features: the Umbrella Boolean would be included in all plugins and tell whether any dominant order is available for a language, and once it is valued positive, the more specific categorical feature telling the word order pattern will be imported.

There are also features that have only two values, i. e., can be exported either as binary categorical features or booleans. An example would be "Feature 67A: The Future Tense" with only two values: "Inflectional future exists" and "No inflectional future." In this situation, the boolean toggling the future tense will be put into a Language Plugin, but no other construction varieties will be imported even if it is positive. Once a typology of inflectional future is developed (or acquired from another resource), the respective module will be created and associated with the already existent boolean.

Discussion of more feature types in WALS will be given in our talk. It will also include some quantitative data about reusing WALS features and its limitations (omitted here for brevity).

5 Conclusion

Our architecture, together with the method of reusing WALS features, has the potential to cover a large number of structurally varied languages in a short delay. It largely outnumbers prior XMG-bases studies (Kinyon et al., 2006; Generalova and Petitjean, 2020) and is comparable with the LinGO Grammar Matrix (Bender et al., 2002). However, Language Plugins are insufficient for a working grammar without the respective Construction Classes, which is a huge limitation for advancing our research. Nevertheless, we aim to polish our architecture, aspiring to incorporate it in some grammar engineering projects in the future.

References

Emily M Bender, Dan Flickinger, and Stephan Oepen. 2002. The grammar matrix: An open-source starter-kit for the rapid development of cross-linguistically consistent broad-coverage precision grammars. In *Proceedings of the Workshop on Grammar Engineering and Evaluation at the 19th International Conference on Computational Linguistics*, pages 8–14, Taipei, Taiwan.

Benoit Crabbé, Denys Duchier, Claire Gardent, Joseph Le Roux, and Yannick Parmentier. 2013. Xmg: extensible metagrammar. *Computational Linguistics*, 39(3):591–629.

Matthew S. Dryer and Martin Haspelmath, editors. 2013. *WALS Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

Valeria Generalova and Simon Petitjean. 2020. A prototype of a metagrammar describing three-argument constructions with a morphological causative. *Typology of Morphosyntactic Parameters*, 3(2):29–51.

Alexandra Kinyon, Owen Rambow, Tatjana Scheffler, SinWon Yoon, and Aravind Joshi. 2006. 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*, pages 17–24.

Simon Petitjean, Denys Duchier, and Yannick Parmentier. 2016. Xmg 2: describing description languages. In *International Conference on Logical Aspects of Computational Linguistics*, pages 255–272. Springer.