WG7: UniCoDeX
(*Universal Construction Dependency Xrammar*)

Peter Ljunglöf and Lori Levin (editors)
with contributions from Archna Bhatia, Nina Böbel,
Nurit Melnik, Nathan Schneider, and Amir Zeldes

(based on discussions with the whole working group)

June 14, 2023

1 Introduction

This is a summary of the discussions that took place in Working Group 7 during the Dagstuhl seminar 23191 *Universals of Linguistic Idiosyncrasy in Multilingual Computational Linguistics*.

WG7 was formed from the initial working groups 2 (*Annotation of particular kinds of constructions*) and 4 (*Finding idiosyncrasy in corpora*), who independently of each other realised that they wanted a group that was focused on the relation between Construction Grammar and Universal Dependencies. So for the second half of the seminar we formed WG7 with the initial name "CxG meets UD", but we changed the name of the group to the more catchy UniCoDeX (*Universal Construction Dependency Xrammar*).

The group met for four sessions and had intense discussions which resulted in the formation of three interconnected tasks. This document tries to reflect the results of our discussions and the way forward.

1.1 Participants

The following people were active in the discussions, at least for some sessions (in alphabetical order):

- Timothy Baldwin (MBZUAI – Abu Dhabi, AE)
- Archna Bhatia (Florida IHMC – Ocala, US)
- Nina Böbel (Universität Düsseldorf, DE)
- Francis Bond (Palacký University Olomouc, CZ)
- Jörg Bücker (Universität Düsseldorf, DE)

- Mathieu Constant (ATILF – Nancy, FR)
- Daniel Flickinger (North Newton, US)
- Sylvain Kahane (University Paris Nanterre, FR)
- Lori Levin (Carnegie Mellon University – Pittsburgh, US)
- Peter Ljunglöf (University of Gothenburg, SE)
- Teresa Lynn (MBZUAI – Abu Dhabi, AE)
- Nurit Melnik (The Open University of Israel – Raanana, IL)
- Joakim Nivre (Uppsala University, SE)
- Alexandre Rademaker (IBM Research – Sao Paulo, BR)
- Manfred Sailer (Goethe-Universität Frankfurt am Main, DE)
- Agata Savary (University Paris-Saclay, CNRS – Orsay, FR)
- Nathan Schneider (Georgetown University – Washington, DC, US)
- Leonie Weissweiler (LMU München, DE)
- Amir Zeldes (Georgetown University – Washington, DC, US)

1.2 Overview of discussion topics

The overall question of WG7 was how to connect Construction Grammar (CxG) with Universal Dependencies (UD) in a way that both Construction Grammar projects and Computational Linguistics projects can benefit from. Our discussions were organized around three interrelated topics:

A. Documenting morphosyntactic diversity in UD[1]

How can UD annotations and guidelines be improved to better reflect typological differences between languages?

B. Standard for a construction annotation layer in UD

How could construction annotations be added to augment UD treebanks? What annotation standard would be needed?

C. Searching for constructions in UD treebanks

How could UD treebanks be queried for interesting examples of a given construction?

[1] Previous name: "Typologically valid UD annotation guidelines"

2 Topic A. Documenting morphosyntactic diversity in UD

The general goal of this topic is to verify the typological coverage of UD, increase its consistency, and advise users on how to analyze constructions in their languages. There are two important concepts: *meta-constructions* framed in comparative terms based on Croft (2022); and *morphosyntactic strategies* that specific languages may employ. The group working on this topic will create a collection of annotation examples and guidelines for different languages and meta-constructions:[2]

- This collection corresponds to the "Swadesh list" for morphosyntax from WG2 – we dub it the *Nivre list* since it was proposed by Joakim Nivre. Using this checklist, treebankers can determine which mophosyntactic strategies are used in their languages for each meta-construction and how to annotate them.
- One component of this collection is a spreadsheet with languages in the rows and meta-constructions in the columns, where the cells contain links to queries yielding annotation examples and notes indicating which morphosyntactic strategy is illustrated in each example.
- Another component is a table of meta-constructions and morpho-syntactic strategies, with examples from different languages and language families.

This collection will be used to promote consistent and typologically informed coverage of morphosyntactic strategies in UD and update the general UD annotation guidelines from a typological perspective, including morphosyntactic strategies as defined by Croft (2022).

- This will help UD to find areas where we can improve current explanations and analyses to be more typologically oriented.
- It will also be of help when extending the guidelines to improve the coverage of explanations.
- It will ensure that it is possible to represent all (or at least most) known typological diversity in UD.
- It will facilitate using UD for research in language typology.

2.1 Example(s)

Object predication is a cross-linguistic meta-construction which uses different strategies in different languages. In this meta-construction a semantic object is information-packaged as a predicate. (This is conventionally called a predicate nominal.)

[2] Croft (2022) calls them "constructions", but we refer to these as meta-constructions to differentiate them from language-specific constructions in the Construction Grammar sense.

- English: verb copula strategy ("Dani is a student")
- Russian: zero strategy ("Dani student")
- Hebrew: pronoun copula strategy ("Dani hu student")
- Classical Nahuatl: inflect the noun as a verb: ("ni-ticitl", 1sg-doctor)

We want annotation guidelines for each language/strategy so that an annotator will not necessarily go to an English treebank by default. The table of morphosyntactic constructions and strategies should be able to guide a treebanker to the right strategy. The spreadsheet should guide the treebanker to examples from languages that use that strategy, which illustrate how to make dependency trees for that strategy.

2.2 Future work

People who are interested in working on this topic after Dagstuhl:

- Lori and Joakim (group leaders)
- Alexandre, Amir, Archna, Jörg, Leonie, Nathan, Nurit, Sylvain

As a concrete first step the group agreed to do the following in the near future:

- add at least 10 languages and 10 meta-constructions to the Nivre list.
  - We will start with basic meta-constructions like clausal possession, comparison, and argument alignment (accusative or ergative).
  - We will write guidelines for each meta-construction and each strategy.
- when this has been finished, there will be an internal review within the group to decide about future steps.

3 Topic B. Standard for a construction annotation layer in UD

The general goal of this topic is to develop recommendations for how to annotate UD treebanks with constructions. They should be useful for several different use cases, workflows and granularities, such as:

- Use cases: we could be building a new constructicon from the ground up and want to come up with good definitions of constructions, or we might already have an existing constructicon which we want to use for annotation or extend with new constructions
- Different annotation granularities: from the coarsest level (to just annotate the head of a construction with its name), to the most fine-grained (to also annotate all the construction elements with their names and spans within the sentence)

- Different workflows: people might want to annotate one construction at a time, or several at once – or they might want to use an iterative approach where they start with coarse-level annotation and then refine them

There is a very rough initial proposal for a CoNLL notation in the appendix.

3.1 Future work

People who are interested in working with this topic after Dagstuhl:

- Leonie (group leader)
- Alexandre, Amir, Archna, Francis, Lori, Manfred, Nathan, Nina, Nurit, Peter, Sylvain

As concrete first steps the group agreed to annotate a limited family of constructions in different languages, with the hope of writing a joint paper during autumn.[3] Some initial ideas of constructions that could be interesting to annotate were:

- age constructions, rates (mph etc), comparatives, resultatives, ...
- idiosyncratic, lexicalised constructions, such as X-and-X (Swedish), N-über-N (German), N-after-N (English)
- cross-linguistically common constructions such as types of conditionals, possession, comparison etc. (i.e. exponents of meta-constructions, see above)

The group will continue discussing topics such as:

- naming convention for constructions
- integration with existing annotation tools
- annotating/marking candidates that have been checked and are not a certain construction
- which token in the UD tree should be annotated with the construction?
  - the natural choice is to annotate the token that is highest in the UD tree – but it is unclear what to do if the construction covers disconnected parts of the UD tree

The group agreed to postpone more complicated questions, such as:

- how to handle cross-sentential constructions
- how to handle nesting and composition of constructions
- how to handle constructions on different levels of granularity (more specific vs. more general constructions)

[3] Possibly targeting LREC-COLING 2024, with submission deadline October, or ICCG 2024 with deadline in spring.

4 Topic C. Searching for constructions in UD treebanks

The general goal of this topic is how to formulate search queries that can locate interesting examples of a given construction. This is very closely related to the previous two topics, as they all depend on being able to search for constructions in treebanks.

The group discussed some issues that arise when it comes to formulating search queries, such as:

- we want guidelines that help people with writing and refining queries
- we want (semi-)automatic techniques for extracting relevant search queries from an existing constructicon entry
- precision/recall tradeoff: it is probably more important to have a good recall than good precision, but it is usually easier to improve the precision by modifying a query
- possible strategies to increase the recall can be to use approximate tree matching or to loosen some constraints in the query

4.1 Future work

People who are interested in working with this topic after Dagstuhl: the same as for topic B, with Leonie as group leader.

In the beginning this topic will be closely related with topic B. To be able to annotate the treebanks we will have to formulate search queries that can find potential candidates. While doing this iterative process for a diverse set of constructions in different languages we hope to come up with more general guidelines on how to write construction queries.

5 Related work/links

The following are the existing constructicons that we are aware of:

- English: Berkeley FrameNet Constructicon: http://sato.fm.senshu-u.ac.jp/frameSQL/cxn/CxNeng/cxn00/21colorTag/
- English: Birmingham English Constructicon: https://englishconstructicon.bham.ac.uk/
- English: CASA (FAU Erlangen-Nürnberg): https://constructicon.de/
- German: FrameNet-Konstruktikon (HHU Düsseldorf): http://framenet-constructicon.hhu.de/

- Swedish: Svenskt konstruktikon (Univ. of Gothenburg): https://spraakbanken.gu.se/karp/#?mode=konstruktikon
- Brazilian Portuguese: FrameNet Brasil (FU Juiz de Fora): https://www2.ufjf.br/framenetbr-en/
- Japanese: Japanese FrameNet (Keio University): https://jfn.st.hc.keio.ac.jp/
- Russian: Russian Constructicon (UiT Arctic University of Norway): https://constructicon.github.io/russian/
- Most of the different constructions were presented at the Constructicon Alignment Workshop (CAW, December 2022), and video recordings are available here: https://www.globalframenet.org/caw2022

Croft (2022) contains a glossary of different *comparative concepts* (meta-constructions, strategies, information packaging, etc.), and this glossary is available online here:

- Interactive interface: https://spraakbanken.github.io/ComparativeConcepts/
- GitHub repo: https://github.com/spraakbanken/ComparativeConcepts

Finally, here is a list of different search engines for corpora, tools and treebanks, that can be used to find constructions:

- Grew-match: https://match.grew.fr/
- SPIKE (query-by-example): https://spike.apps.allenai.org/
- DepEdit: https://gucorpling.org/depedit/
- UDAPI: https://udapi.github.io
- Korap (IDS-Mannheim: https://korap.ids-mannheim.de/ and https://github.com/KorAP/
- Corpus workbench (CWB, useful for larger corpora) – several sites use CWB, such as:
  - Språkbanken Korp (Univ. of Gothenburg): https://spraakbanken.gu.se/korp/
  - CQPWeb (Lancaster Univ.): https://cqpweb.lancs.ac.uk/
  - CWB source code can be found here: https://cwb.sourceforge.io/

References

Croft, William. 2022. *Morphosyntax: constructions of the world's languages*. Cambridge: Cambridge University Press.

A Proposed standard for construction annotation in UD

We propose a new layer for selectively annotating constructions on top of UD trees. This is intended for constructions (in the sense of Construction Grammar) whose form and meaning/function is not already captured well by the UD tree. Construction instances receive a type name (possibly from a constructicon resource) and may contain relations to construction elements. The elements of the construction are not constrained by the UD tree: e.g., a construction element may cut across multiple UD subtrees. For now, we envision that they would be marked in the MISC column of .conllu files, though in principle they could be moved to a separate extension column.

The annotation layer does not have the goal of directly indicating the elements of form or meaning that are characteristic of or required by the construction, beyond indicating the construction evoker and spans of construction elements. Aspects of the UD analysis (tags, deprels, morphological features) that are characteristic of a construction's form should be described as such in a type-level constructicon entry. The precise contents of such an entry are not part of this proposal, but constructicons incorporating UD information in some way already exist (e.g., the Russian Constructicon).

A.1 Full

Showing three overlapping constructions for completeness:

```
1   Sam    CxnEltOf=5:predicative-age.Individual,5:property-predication.Subj
2   is     CxnEltOf=property-predication.Cop
3   three  CxnEltOf=4:num-mod.Quantity,5:predicative-age.Value
4   years  Cxn=num-mod|CxnEltOf=4:num-mod.Counted,5:predicative-age.Units
5   old    Cxn=predicative-age,property-predication|CxnEltOf=5:property-predication.Pred
```

This effectively encodes construction-element relationships as dependencies (*offset:relation* notation echoes DEPS column), which would allow for straightforward graph querying. A common query might be to list the UD deprels associated with a construction element.

Note that i) a word may evoke multiple constructions, ii) a word may be both the evoker and an element of an evoked construction, iii) a word may participate in multiple elements of the same evoked construction.

Comma-separated lists should be sorted primarily by head node (where present), secondarily by construction name, thirdly by construction element name.

A.2 Full-consolidated

```
1   Sam    _
2   is     _
3   three  _
```

```
4   years  Cxn=num-mod(3:Quantity,4:Counted)
5   old    Cxn=predicative-age(1:Individual,3:Value,4:Units),\
           property-predication(1:Subj,2:Cop,3-5:Pred)
```

This is equivalent to the Full representation but consolidates all parts of an evoked construction on one line. It might be suitable for human annotation, to be automatically expanded to the Full representation with a script.

Comma-separated construction elements should be listed in node sort order. Constructions should be sorted alphabetically by name.

A.3 Simple

A partial representation may be useful in certain stages of an annotation workflow, e.g. before the full description of the construction is known, or before applying semiautomatic methods to identify construction elements.

The Simple notation includes the name of a construction, omitting any construction elements. A span may optionally be included for rendering purposes, but this span does not necessarily have any theoretical status.

```
1   Sam    _
2   is     _
3   three  _
4   years  Cxn=3-4:num-mod
5   old    Cxn=1-5:predicative-age,property-predication
```

A.4 Exclusions

When manually reviewing forms that are candidate matches of a construction, it may be helpful to indicate that one of them is a non-match (a false positive). This can be done with the `ExcludeCxn` feature:

```
1   Sam    _
2   is     _
3   three  _
4   years  Cxn=3-4:num-mod
5   old    Cxn=1-5:predicative-age,property-predication|ExcludeCxn=object-predication
```

Though we suggest the name `ExcludeCxn` in this standard, it should be regarded as a tool for development. Ideally, a corpus will be systematically reviewed for candidates of a construction, and excluded candidates discarded in the final version of the data.

A.5 Linking to a constructicon

If a constructicon resource exists, it should be declared in a metadata line in the file, and names of constructions from the resource should be prefixed with a namespace.

A.6 TBD issues

- Where are spans vs. heads used? Is a construction-evoking element allowed to be a span? Allow discontinuous spans (and change existing commas to semicolons)?
- A status field to indicate auto rather than gold matches?
- Allow question marks to indicate uncertainty during development?

A.7 Example annotations

A.7.1 "The more you post the more money you make"

Let's assume that the first comparative word ("more") is the head. Then that word will be annotated like this in the simple format (the span 1–8 is optional):

```
1   the    _
2   more   Cxn=1-8:comparative-correlative
3   money  _
...
```

And like this in the Full notation:

```
1   the    _
2   more   Cxn=1-8:comparative-correlative(1-4:Condition,6-10:Result,\
           2:ConditionDegree,7:ResultDegree)
3   money  _
...
```

A.7.2 "Sam is so glad that you are here that he baked a cake"

Advanced example (consolidated notation), showing two candidate matches of the same construction type on the same construction evoker, one of which is correct and one of which is incorrect (indicated by an excluded span):

```
...
3   so    _
4   glad  Cxn=causal-excess(1:Predicand,3:Degree,3-8:Cause,9-13:Result)\
          |ExcludeCxn=causal-excess(5-8:Result)
5   that  _
...
```

Or in the simple form:

```
...
3   so    _
4   glad  Cxn=1-13:causal-excess|ExcludeCxn=1-8:causal-excess
5   that  _
...
```