Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022), pages 4842–4850
Marseille, 20-25 June 2022
© European Language Resources Association (ELRA), licensed under CC-BY-NC-4.0

Unifying Morphology Resources with OntoLex-Morph. A Case Study in German

Christian Chiarcos, Christian Fäth, Maxim Ionov
Applied Computational Linguistics (ACoLi)
Goethe University Frankfurt, Germany
{chiarcos|faeth|ionov}@cs.uni-frankfurt.de

Abstract

The OntoLex vocabulary has become a widely used community standard for machine-readable lexical resources on the web. The primary motivation to use OntoLex in favor of tool- or application-specific formalisms is to facilitate interoperability and information integration across different resources. One of its extension that is currently being developed is a module for representing morphology, OntoLex-Morph. In this paper, we show how OntoLex-Morph can be used for the encoding and integration of different types of morphological resources on a unified basis. With German as the example, we demonstrate it for (a) a full-form dictionary with inflection information (Unimorph), (b) a dictionary of base forms and their derivations (UDer), (c) a dictionary of compounds (from GermaNet), and (d) lexicon and inflection rules of a finite-state parser/generator (SMOR/Morphisto). These data are converted to OntoLex-Morph, their linguistic information is consolidated and corresponding lexical entries are linked with each other.

The main contribution of this paper is the discussion of the current state of OntoLex-Morph and its validation on different types of real-world resources for a single language. In the longer term, the successful application of OntoLex-Morph to such diverse data, along with the adjustments to the vocabulary observed in the process, will be a means to establish interoperability among morphological resources as well as between them and classical lexical data such as dictionaries, WordNets, or thesauri.

Keywords: morphology, OntoLex, Linguistic Linked Open Data

1. Background

OntoLex[1] is a widely used community standard for publishing lexical resources as RDF data, as well as to link them with knowledge graphs, ontologies or other lexical data sets. With the publication of language resources on the web and using resolvable URIs to identify concepts, entities and relations, information from different sources can be more easily integrated and interlinked with each other, hence the term 'Linked Data' (Berners-Lee, 2009), and in the language resource community, the application of OntoLex to an increasing amount of lexical data has been a key element in the recent development of Linguistic Linked Open Data technology (Cimiano et al., 2020), e.g., in the European projects ELEXIS and Pret-a-LLOD (Krek et al., 2019).

The OntoLex core module provides fundamental data structures for lexical resources, most notably lexical entries, lexical forms and lexical senses, and beyond that, the data structures of the OntoLex core module already provides vocabulary to represent morphological information, but limited to complementing lexical entries with a morphosyntactic categories such as part of speech, and individual forms with different agreement features such as gender, case, number, etc.

In addition to that, the `decomp:` module of OntoLex is relevant to certain aspects of morphology, however, it is primarily concerned with multi-word expressions. Although this approach can be used to represent morphological composition, it is not fully clear how linking morphemes or inflected forms in compounding should be represented in this context: Decomposition is defined here as an operation between lexical entries, and a linking morpheme can be a lexical entry in its own right (this is the function of `ontolex:Affix`, a subclass of lexical entries in the core module). So, `ontolex:Affix` and `decomp:Component` can represent sub-word units and can be put into relation to the lexical entries in which they are contained via properties like `decomp:correspondsTo` or `decomp:subterm`, but problems arise when confronted with a compound like German *Gästehaus* 'guest house'. This expression involves the plural form *Gäste* of German *Gast* 'guest' rather than its canonical form, and using the OntoLex-Lemon vocabulary with the inventory of modules published until 2019, it is unclear how to express the morphological constraints involved in this compound.

So, while the core and `decomp` modules of OntoLex-lemon already contain various classes and properties that can be used to describe morphological data, they are also severely limited. Neither derivational morphology nor morphological information beyond the specification of grammatical features was expressible with this model, and lexicalizations of the same concept with different parts of speech required independent lexical entries, without being able to represent the systematic relations on the level of form and meaning that hold between them.

However, it was foreseen in OntoLex that more

[1] `https://www.w3.org/2016/05/ontolex/`

detailed morphological information would be provided at a later point in time. In particular, the OntoLex core model includes the object property `ontolex:morphologicalPattern`, which, however, remained underspecified until a future module for morphology would have been created.

OntoLex-Morph is the current prototype for this module, and the primary publication on the topic is Klimek et al. (2019). However, it is still under development, so certain aspects of the modelling may change. Indeed, the primary goal of our paper is to validate – and, if necessary, revise – the current OntoLex-Morph draft by converting open source resources representative for all major types of NLP-relevant morphological resources for a particular language to OntoLex-Morph. As a sample language, we focus on German, as a typical inflectional language which previously has also been addressed by Declerck and Racioppa (2019).

We deem such a validation necessary, as Klimek et al. (2019) motivated OntoLex-Morph from requirements and applications in digital lexicography and chose their use cases accordingly. Although a designated focus of the module has always been to address the needs of language technology as well as lexicography, technological requirements have not been broadly discussed in the literature so far. In particular, as described by Klimek et al. (2019), primarily the representation of the decomposition of ontolex:LexicalEntry and ontolex:Form resources and their relation with the morph:Morph element had been modelled (these are the primary requirements for modelling lexical resources), whereas the originally foreseen vocabulary elements for generative morphology (i.e., applications in language technology) were still under development and have not been addressed in detail.

However, these topics have been internally discussed under the auspices of Bettina Klimek and Maxim Ionov and their application has also been documented in a small number of papers. As such, Declerck and Racioppa (2019) described the conversion of a proprietary multilingual resource to OntoLex-Morph, but limited to inflectional morphology and their description. Other lexical resources in language technology, however, provide specific information on derivation, compounding and morphological generation, and here, we explore to what extent the current OntoLex-Morph vocabulary can be applied to represent these in an adequate fashion, as well.

For studying the linguistic coverage of the current draft of the OntoLex-Morph vocabulary, we focus on the conversion of *heterogeneous* open source resources that have been developed independently, and, in parts, in international community efforts that aim to provide compatible resources for a large number of languages. So, while we focus on a single language in our presentation, the converters can produce resources for a multitude of languages.

We describe the conversion of morphological resources for German, also to demonstrate the key benefit of OntoLex-Lemon and RDF technology in general. That is, resources can be easily integrated by means of declarative links between them, So, as a final step, we also produce an automated linking between the individual converted resources where they refer to the same lexical element.

2. OntoLex-Morph

In this paper, we operate with vocabulary version 4.5.2 from November 2021, however, some of its conclusions will lead to minor adjustments of this data model.[2] Figure 1 illustrates this version of the OntoLex-Morph vocabulary, but note that it continuously evolved since then, so that the most recent version contains minor modifications. The current status can be confirmed against the OntoLex wiki.[3]

The `morph:Morph` class represents the fundamental data structure of the module and provides a number of subclasses for different types of morphemes. However, `morph:Morph` does not represent a linguistic morpheme, but it can also stand for a morpheme variant (allomorph). Whether a particular `morph:Morph` represents a morpheme (including its allomorphic variants) or merely an allomorphic variant of a small set of related forms associated with the same underlying (but not explicitly modelled) morpheme is a decision left to the provider of the data, and indeed, not all morphological resources provide robust criteria to distinguish morphemes and morphs.

A more controversially discussed modelling decision is that `morph:Morph` is understood as a type of lexical entry. This definiton follows the idea that computational morphologies usually adopt morphemes as individual entities with a number of grammatical and semantic properties, comparable to headwords in a print dictionary. Accordingly, a `morph:Morph` can have one or multiple forms, a (potentially empty) set of associated senses and it can be a structural unit in a `lime:Lexicon`. This is consistent with and appeals to the needs of language technology, it does, however, contradict the implicit expectation that lexical entries are, indeed typical head words of *print* dictionaries, as usually, inflectional morphemes receive a different treatment in lexicography.

Another important data structure for *describing* morphological relations between lexical entries are word formation relations. These are relations that link (the lexical entry that represents) a derived word with (the lexical entry representing) its morphological base,

[2] Note to reviewers: For the final paper, we plan to incorporate these changes into the then-current model and describe it as is. However, agreement on these changes is achieved by a community-wide consensus and has not been achieved at the time of writing.

[3] See `https://www.w3.org/community/ontolex/wiki/Morphology` and links on the site.

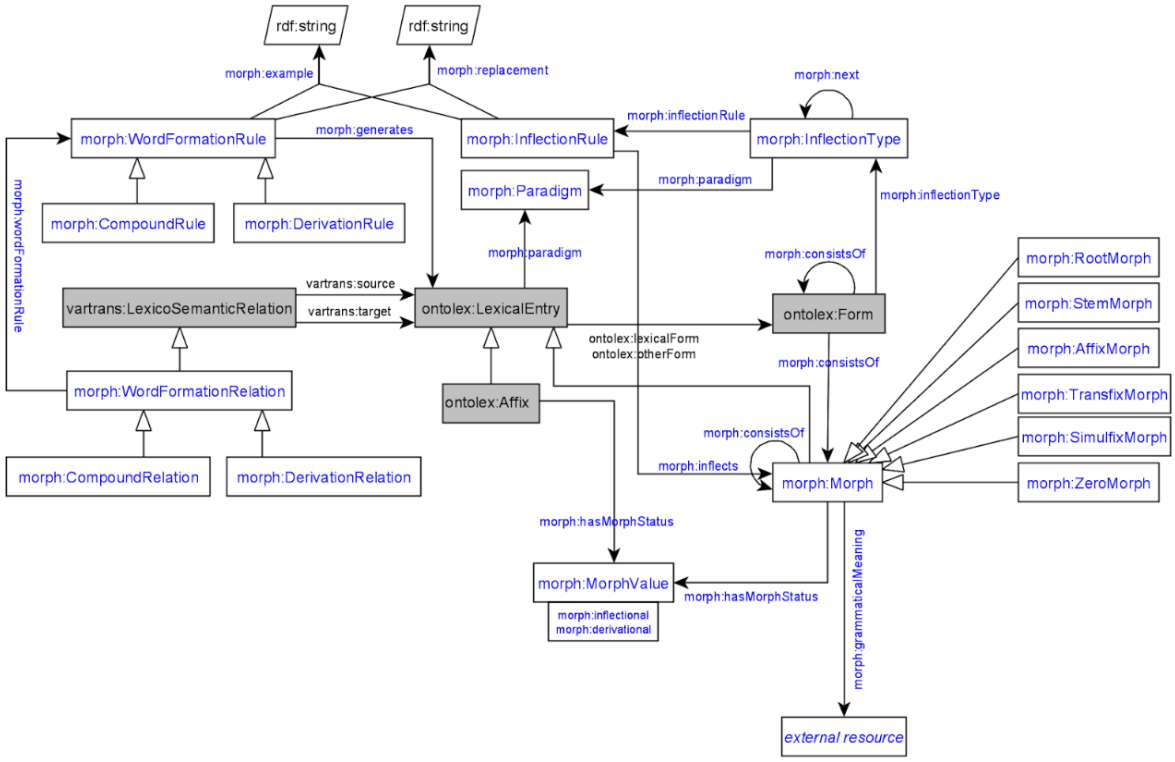

Figure 1: Draft of the OntoLex-Morph module, version 4.5.2, Nov. 2021. Courtesy of Bettina Klimek.

resp., (the lexical entry that represents) a compound with (the lexical entries representing) its head.

Word formation relations are to be distinguished from word formation rules that can be used to *generate* a lexical entry from one or multiple underlying morphemes. These rules can be represented by their exemplary application (`morph:example`) or a direct replacement operation (`morph:replacement`), formalized as string operations with regular expressions comparable to capturing groups in Perl (or, for example, SPARQL). Declerck and Racioppa (2019) describe `morph:replacement` as an object property with `source` and `target` objects, but we follow the November 2021 diagram in assuming that this is a datatype property with a string literal that represents a Perl-style replacement operation, i.e., `s/`(*source expression*)`/`(*target expression*)`/`. Since the November 2021 diagram, a novel property has been introduced that also allows to link a word formation rule (and thus, indirectly, a word formation relation) with any (morphs representing) derivational or linking morphemes involved in the process.

Another type of rules with similar characteristics are inflection rules. The use of `morph:replacement` is analoguous to word formation rules, and like (more recent versions of) word formation rule, inflection rules can be associated with `morph:Morphs`. A specific feature of inflection is, however, that inflection rules are organized into paradigms (associated with lexical entries) and inflection types (associated with lexical forms). Inflection types have been introduced for agglutinating languages and they can be used to model a series of 'slots' there individual inflection morphemes for different functions, e.g., case, number, gender, can be inserted. For an inflecting language like German, normally, there will be one inflection type for one inflection rule.

The original scope of the module involves two main parts: 1) enabling the representation of elements that are involved in the decomposition of lexical entries and word-forms (description), and 2) enabling the representation of building patterns that are involved in the formation of lexical entries and word-forms (generation). So far, the application to generative morphology has not been demonstrated, and below, we show that these data structures can indeed be used for the purpose, but also, that some revisions of the original intuitions and definitions are necessary.

Although we argue that OntoLex-Morph is an adequate vocabulary for generative morphology, it is to be noted that its scope clearly limits it to *morphological* phenomena and processes. In particular, morpho*phonological* processes (e.g., assimilation, etc.) are beyond the scope of the model. For this reason, OntoLex-Morph allows to describe and generate the 'deep' morphological structure — the first level in terms of two-level morphologies, but not necessarily its surface forms, the second level (Koskenniemi, 1983). Heuristically, it is possible to overcome this limitation, but it is important to note that this is not a technologi-

cal limitation, but a conscious decision about the scope of the model. It is possible that future OntoLex modules will also address phonological and orthographical transformations or formalisms for transliteration, and then, morphophonological rules could be one of their domains of application. At the moment, however, no such vocabulary does exist and it is not planned to be a part of OntoLex-Morph.

This also means that the primary use of OntoLex-Morph with respect to generative morphology will probably be to exchange rules and morphological grammars in an implementation-independent form, but *neither* morphological generation on the fly *nor* their implementation with native RDF technology (e.g., the application of replacement rules in SPARQL Updates).

In the following sections, we now describe selected resources for inflectional morphology, derivational morphology and compounding in German, as well as the partial conversion of an FST grammar to OntoLex-Morph, and their subsequent application for creating morphological generators in any host language with support for Perl-style regular expressions.

3. Inflection: Unimorph

Following the success of the Universal Dependencies as a growing community project, a similar effort for the development of cross-linguistic features for inflectional morphology has been initiated: Universal Morphology.[4]

UniMorph provides data for more than 100 languages as tables with three rows: lemma, (inflected) form and (morphosyntactic) features as illustrated for the genitive singular of German *Zufall* 'chance' below:

```
Zufall Zufalls N;GEN;SG
```

Previously, Chiarcos et al. (2018) suggested a Linked Data representation of the original Unimorph data in OntoLex. Using the OntoLex-Morph inventory, we extend this representation slightly: We extract morphological patterns from individual entries by comparing canonical form and generated from, if one form contains the other as a full string, we consider the string difference as an independent morph and provide an inflection type for it.

The replacement itself is represented as a regular expression as a property of the `morph:InflectionRule` object that is associated with a `morph:InflectionType`. Furthermore, inflection types are grouped into paradigms. As Unimorph does not provide us with explicit paradigm information, we assume that every part of speech consitutes a separate paradigm. However, other datasets may provide more fine-grained differentiations between paradigms.

[4] `https://unimorph.github.io/`

In this way, the conversion from Unimorph to OntoLex-Morph already involves an interpretative element that goes beyond the original resource. Of course, the patterns extracted in this way will be heuristic and limited in coverage, but they already provide a baseline functionality for morphological generation and analysis, as they can subsequently be used to generate hypotheses for inflected forms or to count the frequency of morphs and allomorphs in a UniMorph dataset or in a corpus annotated against UniMorph.

We use TARQL (Cyganiak, 2015) to perform the conversion, so that the entire transformation can be described as a single SPARQL update, and directly applied to the original TSV data. In this SPARQL update, we normalize canonical form and inflected form (lower case) and then perform a string comparison to (heuristically) determine the morph element. If the inflected form equals the canonical form, we predict a `morph:ZeroMorph`, if it starts with the canonical form, we predict a `morph:Suffix`, if it ends with the canonical form, we predict a `morph:Prefix`, and if it otherwise contains the canonical form, we predict a `morph:Circumfix`.

These regularities are also stored as Perl-style replacement operations in the property `morph:replacement` of the corresponding `morph:InflectionRule`:

```
entry:Zufall_N a ontolex:LexicalEntry ;
 ontolex:canonicalForm form:Zufall_N ;
 ontolex:otherForm     form:Zufalls_N ;
 morph:paradigm        paradigm:N .

form:Zufalls_N a ontolex:Form ;
 ontolex:writtenRep "Zufalls" ;
 unimorph:feats "N;GEN;SG" ;
 morph:consistsOf <morph#s_N.GEN.SG> ;
 morph:inflectionType type:N.GEN.SG .

type:N.GEN.SG a morph:InflectionType ;
 morph:inflectionRule rule:s_N.GEN.SG ;
 morph:paradigm       paradigm:N .

rule:s_N.GEN.SG a morph:InflectionRule ;
 morph:inflects <morph#s_N.GEN.SG> ;
 morph:replacement "s/$/s/".

<morph#s_N.GEN.SG> a morph:Morph ;
 ontolex:lexicalForm "-s" ;
 morph:grammaticalMeaning feats:N.GEN.SG .
```

With this kind of data aggregated into a graph, it now becomes possible to explore, for example, the frequency of a particular replacement for a particular feature combination, or to search over the graph and apply contextual filters to identify additional regularities, e.g., for cases where the heuristic prefix/suffix identification failed. In this way, the conversion by itself already provides a potential added value for the qualitative and explorative analysis of morphology.

The main contribution for the UniMorph modelling in OntoLex-Morph, is that we illustrate the successful application of OntoLex-Morph to the encoding of morphological rules as they can be heuristically extrapolated from a fullform dictionary such as provided by Unimorph.

4. Composition: GermaNet Compounds

In German, composition is a highly productive process. It can involve additional morphemes that have diachronic roots in nominal inflection but that have lost the original grammatical meaning and instead serve a morphological linker element. This presents an additional challenge to represent composition datasets in a machine-readable form. One of the available resources for morphological composition in German is a complement to GermaNet (Hamp and Feldweg, 1997), a German WordNet, in a format comparable to the UniMorph format:

```
Zufallszahl Zufall Zahl
```

The first column contains the compound, the second column contains one or multiple space-separated modifiers and the last column contains the head of the compound. Note that the example *Zufallszahl* 'random number' (from *Zufall* 'chance' and *Zahl* 'number') features the linking morpheme *-s-* but that this is not made explicit, and instead, only the lemmas of the compounds is provided.

As for Unimorph, the conversion is done by means of TARQL, i.e., with a single SPARQL query that transforms line by line into RDF. We represent compounding primarily by means of the established *decomp* module of OntoLex, in a way that heads and modifiers are defined as `decomp:subterm`s of the compound (we do not include the linking morpheme as subterm). However, note that decomp cannot express the order of compounds, and in particular, it cannot replicate the information about the morphological head (that defines the morphological characteristics of the compound) as annotated in the GermaNet split compounds. The representation of morpheme order is also an unresolved matter within OntoLex-Morph, but, here, it is primarily used as a means to identify the morphological head of a compound (for German nouns, this is the last word in the compound) which is annotated explicitly in this resource.

To capture this aspect, we use `morph:CompoundRelation`, a subclass of `ontolex:LexicalSemanticRelation`, and we define the (lexical entry of the) morphological head as the source and the resulting compound as the target of the compounding process. The linking morph(eme) (extrapolated using a string diff) is attached to the compound relation by `morph:contains`. Note that OntoLex-Morph does not provide any vocabulary to express head information, we thus recommend the restriction of compound relations to exactly this function:

```
entry:Zufallszahl a ontolex:LexicalEntry ;
 ontolex:canonicalForm form:Zufallszahl .

form:Zufallszahl a ontolex:Form ;
 ontolex:writtenRep  "Zufallszahl" .

entry:Zahl a ontolex:LexicalEntry ;
 ontolex:canonicalForm form:Zahl .

form:Zahl  a ontolex:Form ;
 ontolex:writtenRep  "Zahl" .

[a morph:CompoundRelation ]
 vartrans:source entry:Zahl ;
 vartrans:target entry:Zufallszahl ;
 morph:contains  <morph#s> .

entry:Zufallszahl
 decomp:subterm entry:Zahl ;
 decomp:subterm  entry:Zufall .

rule:s a morph:CompoundRule ;
 morph:generates  entry:Zufallszahl .

entry:Zufall  a ontolex:LexicalEntry ;
 ontolex:canonicalForm form:Zufall .

form:Zufall a ontolex:Form ;
 ontolex:writtenRep  "Zufall" .

<morph#s>a morph:Morph , ontolex:Affix ;
 ontolex:lexicalForm form:_s_ .

form:_s_  ontolex:writtenRep "s" .
```

5. Derivation: UDer

Similar to scope and approach of UniMorph, the Universal Derivations (Kyjánek et al., 2020, UDer) expose a considerable number of multilingual resources that feature derivational information. Again, these are harmonized into a simple, but highly restricted format:

```
Zufall_Nm zufällig_A dNA05
```

The conversion to OntoLex-Morph is done by means of TARQL in a way, similar to the UniMorph transformation described above. For source and target word (base and derived form), we split off their morphological information:

```
entry:Zufall_Nm a ontolex:LexicalEntry ;
 uder:POS "Nm" ;
 ontolex:canonicalForm  form:Zufall_Nm .

form:Zufall_Nm a ontolex:Form ;
 ontolex:writtenRep  "Zufall" .

entry:zufällig_A a ontolex:LexicalEntry ;
 uder:POS               "A" ;
 ontolex:canonicalForm  form:zufällig_A .

form:zufällig_A a ontolex:Form ;
 ontolex:writtenRep "zufällig" ;
```

```
 morph:consistsOf  <morph#dVA03%3E> .

rule:dNA05%3E  a morph:DerivationRule ;
 morph:generates  entry:zufällig_A ;
 morph:replacement  [] .

[ a morph:DerivationRelation ]
 vartrans:source entry:Zufall_Nm ;
 vartrans:target entry:zufällig_A ;
 morph:contains  <morph#dVA03%3E> .
```

Note that the (normalized) string of the base form *zufall* is not contained in the (normalized) string of the derived form *zufällig*, as, here, umlaut is involved. So, instead of a concrete replacement operation, we produce a blank node (`[]`) to indicate that *there is* a replacement, but we cannot ascertain its identity or values. Accordingly, we also do not link a concrete morph (for this example). The same strategy is applied for morphs in derivation and inflection.

On this data, we observed a gap in the OntoLex vocabulary, i.e., that we express restrictions that an affix imposes on the bases it can be applied to. As an example, the German suffix *-heit* can be used to produce abstract (feminine) nouns from adjectives, but it cannot be applied to nouns (unless these are homophone with an adjective). In UDer, this information is encoded in the second character of the rule identifier.

Another gap in OntoLex-Morph is the order of morphemes: As derivation relations in OntoLex-Morph are binary, ordering information can be implicitly captured by the type of affix involved. For this reason we deviate from the original OntoLex-Morph vocabulary and make use of the classes `morph:Suffix`, `morph:Prefix` and `morph:Circumfix` in place of the `morph:Affix` class that is currently defined in OntoLex-Morph.

6. Rules: SMOR/Morphisto

Morphisto (Piskorski and others, 2009) is a morphological analyzer and generator for the German SMOR morphology (Schmid et al., 2004) shipped with SFST system (Schmid, 2005) which contains a large-scale morphological lexicon for German. This lexicon differs from the resources described before in that it does not provide a full-form inventory for a certain class of morphological phenomena, but, instead, is a part of a fully generative system (resp., analyzer) that defines string replacement operations (rules) along with underlying forms, from which different surface realizations can be generated (resp., that can be used to parse different realizations). So, Morphisto provides rules and morphs in an explicit form, whereas they had been extrapolated above from string replacements operations.

Finite state transducers are a well-established and efficient technique for morphological analyzers and it makes little sense to replicate their *functionality* in RDF or with RDF technology. However, there is potential in linking their *information* with other morphological resources. On the one hand, this can facilitate the application of the original rule set to new vocabulary. On the other hand, different FST systems (SFST, XFST, or FOMA) are not compatible with each other, and an OntoLex-Morph representation that captures core aspects of the formalism is capable to provide a certain level of interoperability between them.

Finite state transducers are an extension of finite automata that define states, transitions between states and string transformations that are associated with state transitions. For inflectional morphology, this behavior can be replicated by using the `morph:InflectionType` class to represent states, by modelling state transitions by `morph:next` properties that connect different inflection types, and by means of the `morph:InflectionRule` class, resp., its `morph:replacement` property to capture the relation between input and output strings.

Although the terminology does not directly carry over to finite state transducers, we can re-use the `morph:next` transition between inflection types as a means to describe the transition between states, and accordingly, model states in an FST as 'inflection types'. From a linguistic perspective, this is a little bit of a stretch, because an FST state can, indeed, represent information about slots and their order, but is by no means restricted to that. Instead, FST states and their transitions can also be used just to group different rules together, or to implement morphophonological processes. Although we adopt `morph:InflectionType` for FST states and transitions that implement inflectional grammar, it would be less confusing to linguists or lexicographers using the vocabulary to use the `morph:InflectionRule` for this purpose, instead. This is, in fact, one of the suggestions for a revision of OntoLex-Morph that we formulate in the conclusion of our paper.

The main lexical component of the Morphisto grammar is a lexicon of morphemes and base forms (in the file 'lexicon'). Along with it, we also consult the inflection rules (in the file 'flexion.fst'). However, aspects of Morphisto beyond lexicon and inflection have not been addressed, so far, as we focus on a proof-of-principle implementation for generative rules for the example of inflection. Other rules we do not cover include derivation and composition, and, more importantly, morphophonological rules that provide contextualized linearizations of a particular morpheme in different contexts.

For illustration, consider the inflection of the German adjective *zufällig* 'random' that we found to derived from the noun *Zufall* 'chance' but for which UniMorph does not provide inflection information.

The Morphisto/SMOR 'lexicon' file provides a tabular format, but with a somewhat irregular structure, as different types of entries (prefixes, base stems, etc.) have different kinds columns:

A base stem entry consists of the following columns:

1. ENTRY_TYPE: Base$_S$tems
2. FORM, i.e., the base form
3. POS (ADJ (adjective), ADV (adverb), etc.)
4. RULE_TYPE: base
5. METADATA (e.g., 'fremd 'foreign')
6. PARADIGM: e.g., NFem$_{0s}$.

The PARADIGM column represents a tag that corresponds to the start state in the SFST from which possible inflections can be generated (i.e., successor states of $FLEXION$). So, the tag <Name-Neut_s> corresponds to the FST state $Name-Neut_s$ that can be reached from the state $FLEXION$ by replacing the empty string <> with the tag <Name-Neut_s>. It is not specified in the replacement, but we assume that such replacements always occur at the end of a string that represents either the base form or the immediate representation that was generated from the base form in a number of earlier replacement operations.
For *zufällig*, the corresponding lexical entry is

```
<Base_Stems>zufällig<ADJ><base><nativ><Adj+>
```

The paradigm Adj+ corresponds to the state $Adj+$ in the FST grammar, and for the example of a feminine nominative singular form, this provides the following state transitions and replacements:

```
$Adj+$ =         {<>}:{<FB>}           $AdjPos$
$AdjPos$ =       {<+ADJ><Pos>}:{<>}    $AdjFlexSuff$
$AdjFlexSuff$ = {<Fem><Nom><Sg>}:{e} $Adj#$
$Adj#$ =         <>:<Low#>
```

For rendering this information in terms of OntoLex-Morph, we introduce the inflection types (states) :type%23AdjPos (for $AdjPos$), :type%23AdjFlexSuff (for $AdjFlexSuff$) and :type%23Adj%2B (for $Adj+$) and we provide an interpretation of the replacement operations in replacement patterns akin to Perl, i.e., taking the form s/$input_symbol$/$output_symbol$/;. A special character in the replacement patterns is <> which we map to the empty string. If an empty string serves as input to a replacement operation, we assume that this refers to an attachment of information to the end. For an input expression, <> is thus not mapped to the empty string, but to the regular expression $ (in Perl, this matches the end of the input).
As a result, we arrive at the following replacements (slightly simplified for better readability):

```
type:Adj+ a morph:InflectionType;
  morph:inflectionRule [
    morph:replacement "s/$/<FB>/;" ];
  morph:next type:AdjPos.

type:AdjPos a morph:InflectionType;
 morph:inflectionRule [
  morph:replacement "s/<+ADJ><Pos>//;" ];
 morph:next :AdjFlexSuff.

:AdjFlexSuff a morph:InflectionType;
 morph:inflectionRule [
  morph:replacement "s/<Fem><Nom><Sg>/e/;" ];
 morph:next :Adj.

:Adj a morph:InflectionType;
 morph:inflectionRule [
  morph:replacement "s/$/<Low#>/;" ].
```

In order to facilitate such generation, we need to provide pairs of morphosyntactic properties and tags. We use the OLiA system ontology for this purpose (Chiarcos and Sukhareva, 2015), which provides the properties olias:hasTagStartingWith, olias:hasTagEndingWith. Originally, these properties (along with olias:hasTag, olias:hasTagContaining and olias:hasTagMatching) have been used for the precise, partial or regular-expression-based matching from tags in annotated corpus or dictionary to concepts in an ontology of linguistic annotation categories. Here, we use them in a different function: with olias:hasTagEndingWith and olias:hasTagStartingWith, we now define one or multiple tags as used in the replacement rules and express that they are to be inserted before or after the base.
We create an ontological model of the Morphisto/SMOR annotations that defines UnitOfAnnotation objects that carry olias: tag properties along with morphosyntactic features in accordance with the LexInfo vocabulary.[5] Given a lexical entry (which may also carry LexInfo information such as lexinfo:partOfSpeech), and a paradigm (i.e., the inflection type/state assigned) as PARADIGM in the SMOR lexicon file), we then obtain all feasible combinations of units of annotations and the lexical entry (i.e., those that overlap in lexinfo properties). If a lexical entry is defined as lexinfo:partOfSpeech lexinfo:adjective, the corresponding units of annotation are limited to those that agree in the lexinfo:partOfSpeech. For a feminine nominative singular adjective in positive degree, one possible unit of annotation is

```
:ADJ_Pos_Fem_Nom_Sg olias:hasTagEndingWith
   "<+ADJ><Pos><Fem><Nom><Sg>";
lexinfo:case lexinfo:nominative;
lexinfo:partOfSpeech lexinfo:adjective;
lexinfo:gender lexinfo:feminine;
lexinfo:degree lexinfo:positive;
lexinfo:number lexinfo:singular.
```

As this is compatible with any lexical entry that specifies lexinfo:partOfSpeech

[5] https://lexinfo.net/

lexinfo:adjective (and which does not disagree in any of the other features), this is a valid unit of annotation for the lexical entry.
As for the modelling of lexical entries, we create a lexical entry for every row in the lexicon:

```
entry:16594 a ontolex:LexicalEntry ;
 lexinfo:partOfSpeech lexinfo:adjective ;
 morph:baseForm base:16594_zufällig ;
 morph:paradigm paradigm:Adj+ .

base:16594_zufällig a ontolex:Form ;
 ontolex:writtenRep "zufällig" .

paradigm:Adj+ a morph:Paradigm ;
 morph:isParadigmOf type:Adj+ .
```

7. Generation with Chains of Regular Expressions

Our OntoLex-Morph model allows us to trivially extract an (morph:next-ordered) sequence of morph:replacement patterns that can be retrieved using SPARQL and used in any programming language. As an illustration, we use SPARQL to generate Sed scripts for morphological generation.
For generation, we thus retrieve all base forms, concatenate them with the olias: tag and apply the transformations defined by the paradigm (inflection type) associated with the lexical entry. For the baseform *zufällig* in a lexical entry marked as adjective, the string zufällig<+ADJ><Pos><Fem><Nom><Sg> and the lexinfo properties of :ADJ_Pos_Fem_Nom_Sg is thus the basis for generating a particular lexical form.
A limitation of the approach is that we only cover the first level of the two-level SMOR grammar, so that a number of placeholder symbols for allomorphic variants remain. As a heuristic, we just remove all these symbols from the resulting string and use the result as basis for validation.
As our conversion of Morphisto lexicon and SMOR grammar, as well as the generation performed on this basis is incomplete, we evaluate the generated hypothetical form by parsing them with the Ubuntu 20.4 package 'fst', and the Morphisto/SMOR grammar. Out of 25 859 different written representations of the generated hypothetical forms (excluding those identical to base forms), our generator achieved a precision of 78.5% against SMOR/Morphisto,[6] i.e., 20 308 of 5 551 written representations of hypothetical forms (excluding those that identical to the base form) could be successfully parsed. As for the remaining 21.5%, these can be attributed to the insufficient support for morphophonological rules in OntoLex-Morph as well as invalid combinations of alternative base forms and inflection rules that are filtered out in SMOR in subsequent processing steps.
It is to be noted, however, that this vanilla morphological generation from OntoLex-Morph still remains a baseline functionality, and even though it has advantages in portability and sustainability, it lacks optimizations of FST, e.g., in disambiguation strategies and filtering conditions.

8. Conclusion

We have described the conversion of four morphological lexicons from NLP applications to OntoLex-Morph, we illustrated how we can use infer string replacement rules from UniMorph, UniDer and GermaNet compound data using SPARQL, how these can be modelled, how explicit rules (finite state transitions) from FST grammars can be modelled analoguously, and how these rules modelled with OntoLex-Morph can be applied to bootstrap a vanilla system for morphological generation for any programming language with support for regular expressions.
Although this generation is incomplete, it shows the potential of this aspect of the module, as now, essential parts of the Morphisto morphology, for example, can be re-used independently from its original technical enviroment, and even though an RDF representation may be less compact than tabular data formats used otherwise, this establishes a completely new dimension of interopoerability for this data.
At the same time, we found two gaps in the vocabulary: the lack of means to specify the order in derivation (or, alternatively, subclasses that encode order, e.g., suffix and prefix instead of affix) and the lack of links between word formation relations (or rules) and morphs (which is already added to a more recent version of the draft). Other than that, no major obstacles were observed.
Another modelling decision we applied may have an even more profound impact on the model: We demonstrated how sequences of inflection types can be used to emulate transitions in finite state automata. This is quite different from the original motivation of inflection type, and we would suggest a number of smaller revisions here: Instead of creating next transitions between inflection types, it would be beneficial to have transitions between rules, so that *these* could represent finite states. In that case, the terminology would more easily apply to both lexicographic-linguistic and computational uses of this concept. Then, rules and paradigm, as well as rules and lexical forms could be directly connected with each other. With inflection type detached from paradigm and form, it could be reintroduced as a means for classifying rules rather than as an entity that serves to connect rules with forms and paradigm.
All our code and data is available under open source licenses from our GitHub repository.[7] In addition to the

[6] We do not calculate recall, as our conversion only encompasses the inflection component of of SMOR, and neither the derivation nor compounding rules that it also provides.

[7] https://github.com/acoli-repo/

transformations described above, we also performed a linking of the resulting data set for German, but conversion and linking are applicable to a large number of other languages also covered by Unimorph, Universal Derivations or SFST grammars.

9. Bibliographical References

Berners-Lee, T. (2009). Tim Berners-Lee on the next Web. `http://www.ted.com/talks/tim_berners_lee_on_the_next_web.html` (July 31, 2012), February.

Chiarcos, C. and Sukhareva, M. (2015). OLiA – ontologies of linguistic annotation. *Semantic Web*, 6(4):379–386.

Chiarcos, C., Donandt, K., Ionov, M., Rind-Pawlowski, M., Sargsian, H., Schreur, J. W., Abromeit, F., and Fäth, C. (2018). Universal morphologies for the caucasus region. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Cimiano, P., Chiarcos, C., McCrae, J. P., and Gracia, J. (2020). Linguistic linked open data cloud. In *Linguistic Linked Data*, pages 29–41. Springer.

Cyganiak, R. (2015). Tarql (SPARQL for tables): Turn CSV into RDF using SPARQL syntax. Technical report, Technical Report, 2015. Available at: http://tarql.github.io.

Declerck, T. and Racioppa, S. (2019). Porting multilingual morphological resources to OntoLex-Lemon. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 233–238.

Hamp, B. and Feldweg, H. (1997). GermaNet – a lexical-semantic net for German. In *Automatic information extraction and building of lexical semantic resources for NLP applications*.

Klimek, B., McCrae, J. P., Bosque-Gil, J., Ionov, M., Tauber, J. K., and Chiarcos, C. (2019). Challenges for the representation of morphology in ontology lexicons. *Proceedings of eLex*.

Koskenniemi, K. (1983). *Two-Level Morphology: A General Computational Model for Word-Form Recognition and Production*. Number 11 in Publications. University of Helsinki. Department of General Linguistics, Finland.

Krek, S., Declerck, T., McCrae, J., and Wissik, T. (2019). Towards a global lexicographic infrastructure. In *Proceedings of the Language Technology 4 All Conference*.

Kyjánek, L., Zabokrtskỳ, Z., Sevcíková, M., and Vidra, J. (2020). Universal derivations 1.0, a growing collection of harmonised word-formation resources. *Prague Bull. Math. Linguistics*, 115:5–30.

Piskorski, J. et al. (2009). Morphisto – an open source morphological analyzer for German. In *Finite-state Methods and Natural Language Processing: Postproceedings of the 7th International Workshop FSMNLP*, page 224.

Schmid, H., Fitschen, A., and Heid, U. (2004). SMOR: A German computational morphology covering derivation, composition and inflection. In *LREC*, pages 1–263. Citeseer.

Schmid, H. (2005). A programming language for Finite State Transducers. In *FSMNLP*, volume 4002, pages 308–309. Citeseer.

`acoli-morph`