Converting phraseological informationin Walenty to XLE/LFG constraints [WG1/WG2]

Agnieszka Patejuk, Institute of Computer Science, Polish Academy of Sciences

1 Introduction

The aim of this paper is to discuss how the current version of Walenty valence dictionary of Polish ((Przepiórkowski et al., 2014); `http://zil.ipipan.waw.pl/Walenty`) can be converted into XLE/LFG constraints so as to be used for parsing with XLE. The current version of Walenty features a greatly extended formalism for encoding phraseological arguments, making it possible to precisely specify constraints on such arguments and, if need be, on their dependents.

2 Lexicalised arguments

Lexicalised arguments are now formalised in Walenty in a uniform way: a metacategory, `lex`, was introduced, which takes any base category used in Walenty as the first parameter, followed by parameters imposing constraints appropriate for the relevant base category (the number of such parameters depends on the base category) and finally the displayed modification pattern.

(1) Oni witali ją z
they.NOM welcomed she.ACC with
(szeroko) otwartymi ramionami.
widely open.INST.PL arm.INST.PL
'They welcomed her with (widely) open arms.' (== very warmly)

(2)
```
subj{np(str)} + obj{np(str)}
+ {lex(prepnp(with,inst),pl,
XOR('arm','hand'),ratr1({
lex(adjp(agr),agr,agr,'open',
atr1({lex(advp(mod),'widely',
natr)}))}))}
```

There are three arguments in (2): they are enclosed in curly brackets and separated from each other by +. The first argument is the subject, the second is the object – both are not lexicalised. The last argument is a lexicalised (`lex`) prepositional nominal phrase (`prepnp`) with the preposition `with` which requires instrumental case (`inst`) from the nominal which must be specified for plural number (`pl`) and must be a form of either `arm` or `hand`, which must (`ratr1`) be modified (an embedded specification follows) by a lexicalised agreeing adjectival phrase (`adjp`) headed by `open`, which in turn may be optionally (`atr1`) be modified by a lexicalised adverbial phrase (`advp`) headed by `widely`, which must not be modified (`natr`).

3 Types of modification

There are three main modification types defined in Walenty, each of which is exemplified in (2): `natr` (no modification), `atr(1)` (optional), `ratr(1)` (obligatory). The last two have two variants, allowing for or requiring, respectively: any number of modifiers, `(r)atr`, or exactly one, `(r)atr1`.

In (2) the lexicalised prepositional argument (`prepnp`) requires exactly one modifier (`ratr1`) – an adjectival phrase (`adjp`) headed by OPEN, which in turn may be modified by exactly one (`atr1`) adverbial phrase (`advp`) headed by WIDELY, which cannot be modified (`natr`).

4 Choosing the grammatical function

There are two arguments which are labelled with a grammatical function in Walenty: subject (`subj` in (2)) and object (`obj`; defined as the argument which becomes the subject under passive voice). Other arguments are not labelled – see the last, third argument in (2).

In order to use valence information from Walenty in an XLE/LFG grammar, all arguments must be assigned a grammatical function so as to impose relevant constraints. The same applies to lexicalised arguments whose modification pattern is formalised as in (2).

Lexicalised arguments such as `prepnp`, `adjp` and `advp` in (2) are dependents of their respective heads. While all dependents of a verb are assumed to be its arguments (the lexicalised `prepnp`), this does not hold for embedded lexicalised arguments – `adjp` and `advp` in (2) – which are dependents of relev-

ant predicates, but no information is provided concerning their argument/adjunct status.

The grammatical function linking the head with its dependent can be determined on the basis of their respective categories, in a similar way to the strategy which is used for arguments of verbs. In (2) the adjectival phrase (`adjp`) is the adjunct of the nominal head of the prepositional phrase (`prepnp`), while the adverbial phrase (`advp`) is an adjunct of the adjectival phrase (`adjp`).

It may happen, however, that more than one grammatical function is possible – for instance, prepositional dependents of nominals may be an oblique argument or an adjunct. In such situations functional uncertainty may be used in the constraints – a disjunction of potentially relevant grammatical functions (see (4) below). When parsing, such underspecification should be narrowed down to one possibility by the lexical entry of the nominal head, which either takes an oblique or not.

5 Imposing constraints

The `natr` modification pattern (disallowing modification) is handled using negation coupled with functional uncertainty:

(3) `~(PATH GF)`

(4)
```
GF = {SUBJ|OBJ|OBL(-?*)
   |(X)COMP|...|ADJUNCT}.
```

(4) defines `GF` as a disjunction of all possible grammatical functions. The constraint in (3) makes sure that the f-structure in `PATH` contains no grammatical function defined in (4), making it impossible to attach dependents. In (2) the `PATH` for `advp` is `^ OBL ADJUNCT $ ADJUNCT $` – adjunct of the adjunct of the oblique argument.

The constraint provided in (5), where `DEP` is the grammatical function assigned to the modifier, corresponds to `ratr` (which requires modification), `ATTR` corresponds to the relevant attribute and `val` to its value.

(5) `(PATH DEP ATTR)=c val`

(6), which corresponds to `ratr1`, contains an additional constraint ensuring that there are no other dependents: the fragment `GF-DEP` removes `DEP` from the list of grammatical functions in (4), disallowing all except `DEP`.

(6)
```
(PATH DEP ATTR)=c val
~(PATH GF-DEP)
```

However, when the grammatical function corresponding to the modifier is `ADJUNCT`, the constraint in (7) must be used instead of (6):

(7)
```
(PATH ADJUNCT $)=%DEP
(%DEP ATTR)=c val
~[(PATH ADJUNCT $) <h %DEP]
~[%DEP <h (PATH ADJUNCT $)]
```

(7) assigns a local variable (`%DEP`) to an element of the adjunct set (first line), requiring that it satisfies relevant constraints (second line). The remaining lines ensure that there are no other elements of the adjunct set apart from `%DEP` using the head precedence operator (`<h`) – the third line ensures there is no set element to the left of `%DEP`, the fourth line checks there is none to its right.

The constraints corresponding to optional modification patterns, `atr(1)`, are analogous to the ones provided in (5)–(7) for `ratr(1)` – optionality is formalised using a disjunction of appropriate constraints and the `natr` constraint provided in (3), as shown for `atr` in (8) (compare with `ratr` in (5)):

(8)
```
{(PATH DEP ATTR)=c val
| ~(PATH GF)}
```

6 Conclusion

Though Walenty abounds in interesting examples with rich lexicalised information, for reasons of space, only the very basics of the procedure of converting lexicalised dependents described in Walenty into LFG/XLE constraints could be presented here.

References

[Przepiórkowski et al.2014] Adam Przepiórkowski, Elżbieta Hajnicz, Agnieszka Patejuk, Marcin Woliński, Filip Skwarski, and Marek Świdziński. 2014. Walenty: Towards a comprehensive valence dictionary of Polish. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC 2014*, pages 2785–2792, Reykjavik, Iceland. ELRA.