Headword overriding

A report for OASIS LEXIDMA
Author: Michal Měchura `michmech@lexiconista.com` January 2021

This is the second part in my series of reports to LEXIDMA on the relational remodelling of dictionaries. The first part dealt with **subsensing**: the practise of putting senses inside other senses, and how we can remodel such sense hierarchies into a "flat" relational structure. This second part will deal with a phenomenon I have decided to call **headword overriding**. Headword overriding is the technique which allows dictionary authors to put entire entries (or things that look like entire entries) inside other entries, as subentries.

Introduction

Normally, a dictionary entry is headed by a headword, and then the rest of the entry describes that headword. This seems logical and regular. But this logical and regular pattern is sometimes broken by things that **override** the headword.

A typical cause of overriding is the presence of a multiword subentry inside the entry. Its presence somewhere in the body of the entry changes the object of description: from now on, we are no longer describing the headword we started with, we are now describing this multiword expression instead. In the following example (adapted from DWDS), when we enter sense number 2 the object of description changes from the headword *sicher* to the multiword expression *sicher ist sicher!* and then, as we leave sense number 2, it changes back to the headword *sicher*.

```
- entry
    headword: sicher
    pos: adj
    - sense
        definition: nicht von Gefahr bedroht, ungefährdet
        example: ein sicherer Weg
    - sense
        expression: sicher ist sicher!
        definition: lieber vorsichtig sein, lieber nichts riskieren!
        example: ich nehme den Regenschirm mit, sicher ist sicher!
    - sense
        definition: zuverlässig, verlässlich
        example: ein sicherer Tresor
        ...
```

Another frequent cause of overriding is when the object of description changes from the headword in its canonical form to some other form of the headword, such as an inflected form, a variant form or a capitalized form. In the following example (adapted from LDOCE) shows how the object of description changes from *bible* to *the Bible* as we enter sense number 1 and then it changes back to *bible* as we leave the sense:

```
- entry
    headword: bible
    pos: n
    - sense
        expression: the Bible
        definition: the holy book of the Christian religion
    - sense
        label: informal
        definition: the most useful and important book on a particular subject
        example: It's the anatomy student's bible!
```

How dictionary schemas enable overriding

There is usually an element somewhere in the schema which allows itself to be *headed* by something, to have something that resembles a *headword*. Having a headword is normally the privilege of entries, but the idea of headword overriding is that some elements inside entries have this privilege too.

In some dictionary schemas, the elements that are allowed to override the headword are ordinary senses. That is how it is in our two invented examples above: the senses which act as ordinary senses and the senses which act as subentries are instances of the same type (`sense`). What turns a sense into a subentry is the presence of that one heading element (here named `expression`).

Other dictionary schemas have a dedicated type for the elements that are allowed to override the headword, with a name such as `subentry`.

What is and what isn't overriding

Not all entry-internal elements which can be headed by something are necessarily overriding the headword. Take a look at this (adapted) example from DWDS again:

```
- entry
    headword: sicher
    pos: adj
    - sense
        definition: nicht von Gefahr bedroht, ungefährdet
        example: ein sicherer Weg
    - sense
        pattern: vor etw|jmdm sicher sein
        example: hier seid ihr vor der Entdeckung sicher
    - sense
        expression: sicher ist sicher!
        definition: lieber vorsichtig sein, lieber nichts riskieren!
        example: ich nehme den Regenschirm mit, sicher ist sicher!
```

Here, sense number 1 is an ordinary sense and sense number 3 is clearly an example of overriding. But what about sense number 2? It is headed by the grammatical pattern `vor etw|jmdm sicher sein` ("to be safe from sth/sb"). Here it is probably reasonable to argue that this `sense` element is describing (one sense of) the headword *sicher*: the object of description has not changed. The grammatical pattern is merely one of the properties of this sense of *sicher* that are being communicated to the user.

Another way to decide whether the head of an entry-internal element is or isn't a subentry is to ask yourself whether it is likely that a user would search for it. Users might well type expressions such as "the Bible" or "sicher ist sicher!" into the search box of an online dictionary, but probably not a grammatical pattern like "to be safe from sth/sb".

Another clue that can tell us whether we are dealing with an overridden headword or not is to ask whether it would be weird *not* to display the subentry inside the entry, but to provide a clickable hyperlink instead. The user would have to click that hyperlink and this would take him or her to another screen where the subentry would be displayed. If this would not be weird, then this is a clue that the subentry can function as an independent entry in it own right, its "head" can function like a headword, and we are indeed dealing with an instance of headword overriding.

Either way, what ultimately decides the question (whether something is a subentry or not) is the intention of the lexicographer. In a well-designed dictionary schema the answer will be in the names and types of the elements used for encoding. In our invented examples, a `sense` is a subentry if it has an `expression`, otherwise it is an ordinary sense.

Is overriding bad?

As a human dictionary user, as you are skimming down a dictionary entry, you recognize when the object of description has changed and when it has changed back: you figure this out from the way the dictionary entry is formatted on your screen and from your knowledge of the language. This requires almost no extra effort.

As a software engineer, however, when building a program which will process the entries, having to deal with overriding (= with changes in what is being described) is a complication you could do without. Recognizing when overriding has occurred and when not requires some additional programming: the program must be written to know that when it has entered a `sense` which has an `expression` then then object of description has changed to whatever this `expression` contains. It must also remember the previous object of description and know that when it has left that particular `sense` then the object of descripton changes back.

As in the previous report, I would argue again that the needs of software programs (and the people who write them) need to be taken seriously. If our goal is to produce an IT-friendly dictionary encoding standard then we should, ideally, avoid the need for changing the object of description altogether. IT people will find it easier (and themselves more willing) to work with entries in this format if they can count on the fact that the object of description never changes inside an entry, in other words, that all sense inside one entry describe one and the same headword.

Flattening subentries: option 1

Now it is finally time to see how we could remodel subentries relationally. My proposal is to take those entry-internal elements that override the headword out of the entries, and promote them to the status of entries. The fact that they should be shown inside other entries as subentries will be encoded relationally, through unique IDs.

Example 1

Original (adapted from DWDS)

```
- entry
    headword: sicher
    pos: adj
    - sense
        definition: nicht von Gefahr bedroht, ungefährdet
        example: ein sicherer Weg
        - sense
            pattern: vor etw|jmdm sicher sein
            example: hier seid ihr vor der Entdeckung sicher
            - sense
                expression: sicher ist sicher!
                definition: lieber vorsichtig sein, lieber nichts riskieren!
                example: ich nehme den Regenschirm mit, sicher ist sicher!
        - sense
            expression: Nummer Sicher
            definition: Gefängnis
            example: in Nummer Sicher sitzen
    - sense
        definition: zuverlässig, verlässlich
        ...
```

Flattened

```
- entry (ID: #sicher)
    headword: sicher
    pos: adj
    - sense (ID: #sicher_1)
        definition: nicht von Gefahr bedroht, ungefährdet
        example: ein sicherer Weg
    - sense (ID: #sicher_2) (is-subsense-of: #sicher_1)
        pattern: vor etw|jmdm sicher sein
        example: hier seid ihr vor der Entdeckung sicher
    - sense (ID: #sicher_5)
        definition: zuverlässig, verlässlich
        ...

- entry (ID: #sicher-ist-sicher) (is-subentry-of: #sicher_2)
    headword: sicher ist sicher!
    - sense (ID: #sicher-ist-sicher_1)
        definition: lieber vorsichtig sein, lieber nichts riskieren!
        example: ich nehme den Regenschirm mit, sicher ist sicher!

- entry (ID: #nummer-sicher) (is-subentry-of: #sicher_1)
    headword: Nummer Sicher
    - sense (ID: #nummer-sicher-1)
        definition: Gefängnis
        example: in Nummer Sicher sitzen
```

We have done two things in this example. First, we have flattened the senses as suggested in my previous report. Second, we have taken those senses which are subentries and refactored them as separate entries. The fact that a given entry is to be "read" as a subentry of another entry is encoded in the `is-subentry-of` property. Its value is the unique ID of the sense inside which the subentry should be placed at presentation time.

For purposes other than presenting the entry to end users, we have a flat list of entries and inside each entry we have a flat list senses. There are no senses inside senses and there are no entries (or entry-like things, with their own headword) inside entries.

Disadvantages of option 1

When "un-flattening" entries and subentries for display to the end user, we can look at the property `is-subentry-of` of a given entry, and we can figure out from this which (sense of which) other entry this entry should be insert into as a subentry. So far so good. What this doesn't tell us, though, is **where** inside the target the entry should be inserted.

For example, we know that the subentry `#nummer-sicher` should be inserted inside the sense `#sicher_1`, but where exactly? The sense already has some content of its own, such as a definition and an example. During composition for display to end users it will receive some subsenses; these will be appended at the end in the order in which they come in the entry, this is probably uncontroversial. But the subentries? Can we assume that they should be inserted at the end of their target sense? And in what order? And going back to subsenses, is it really safe to assume that subsenses should be inserted first and subentries second?

It turns out that data model has problems with preserving order. The suggestions that subentries and subsenses should always be inserted at the end, in whatever arbitrary order they come, is probably going to be a show-stopper for lexicographers. The order of elements as they are shown to human users is important in human-oriented lexicography. For example, in the *bible* entry from LDOCE, the lexicographers had obviously decided that the *the Bible* subentry should be listed **before** the

other sense:

```
- entry
    headword: bible
    pos: n
    - sense
        expression: the Bible
        definition: the holy book of the Christian religion
    - sense
        label: informal
        definition: the most useful and important book on a particular subject
        example: It's the anatomy student's bible!
```

If we now flatten this entry, we get two entries:

```
- entry (ID: #bible)
    headword: bible
    pos: n
    - sense (ID: #bible_1)
        label: informal
        definition: the most useful and important book on a particular subject
        example: It's the anatomy student's bible!

- entry (ID: #the-bible) (is-subentry-of: #bible)
    headword: the Bible
    - sense (ID: #the-bible_1)
        definition: the holy book of the Christian religion
```

If we now attempt to un-flatten them again into a single entry for display to users, and if we always assume that subentries are to be inserted at the end, we get the opposite order of information from what the lexicographer intended:

```
- entry
    headword: bible
    pos: n
    - sense
        label: informal
        definition: the most useful and important book on a particular subject
        example: It's the anatomy student's bible!
    - sense
        expression: the Bible
        definition: the holy book of the Christian religion
```

Flattening subentries: option 2

All the relations we are dealing with here, namely the entry-subentry relation and the sense-subsense relation, are **directed** relations from a *container* to a *containee* (or from a *parent* to a *child*, if you prefer). We have two options as to where we want to encode the existence of these relations. So far, we have always encoded them in the *containee* (in the *child*): the subsense "knows" which sense it is a subsense of, not the other way around; the subentry "knows" which entry it is a subentry of, not the other way around.

If we do turn it the other way around and encode the relations in the *container* (in the *parent*), we get the additional benefit of being able indicate where exactly in the container the containee should be inserted.

Example 2

Original (adapted from LDOCE)

```
- entry
    headword: bible
    pos: n
    - sense
        expression: the Bible
        definition: the holy book of the Christian religion
    - sense
        label: informal
        definition: the most useful and important book on a particular subject
        example: It's the anatomy student's bible!
```

Flattened

```
- entry (ID: #bible)
    headword: bible
    pos: n
    - SUBENTRY: #the-bible
    - sense (ID: #bible_1)
        label: informal
        definition: the most useful and important book on a particular subject
        example: It's the anatomy student's bible!

- entry (ID: #the-bible)
    headword: the Bible
    - sense (ID: #the-bible_1)
        definition: the holy book of the Christian religion
```

The one and only difference here is that the entry `#bible` now has inside itself the element `SUBENTRY: #the-bible` which, quite simply, says "the entry `#the-bible` should appear here as subentry".

Example 3

For a more substantial example, see this re-worked entry adapted from DWDS.

Flattened (from the same source as in option 1)

```
- entry (ID: #sicher)
    headword: sicher
    pos: adj
    - sense (ID: #sicher_1)
        definition: nicht von Gefahr bedroht, ungefährdet
        example: ein sicherer Weg
        - SUBSENSE: #sicher_2
        - SUBENTRY: #nummer-sicher
    - sense (ID: #sicher_2)
        pattern: vor etw|jmdm sicher sein
        example: hier seid ihr vor der Entdeckung sicher
        - subentry: #sicher-ist-sicher
    - sense (ID: #sicher_5)
        definition: zuverlässig, verlässlich
        ...

- entry (ID: #sicher-ist-sicher)
    headword: sicher ist sicher!
    - sense (ID: #sicher-ist-sicher_1)
        definition: lieber vorsichtig sein, lieber nichts riskieren!
        example: ich nehme den Regenschirm mit, sicher ist sicher!

- entry (ID: #nummer-sicher)
    headword: Nummer Sicher
    - sense (ID: #nummer-sicher-1)
        definition: Gefängnis
        example: in Nummer Sicher sitzen
```

Here, the `SUBSENSE` AND `SUBENTRY` elements indicate where subsenses and subentries should be inserted during un-flattening. The lexicographer's intended order is preserved.

Conclusion

In this report I have attempted to explain how the phenomenon of "entries inside other" is enabled by the technique of **headword overriding** in dictionaries, and how this poses an inconvenience for the computational processing of dictionary entries.

I have gone through two options for remodelling subentries as a flat list of entries where the parent-child relations ate recorded as relations.

In option 1, the relations are recorded inside the child. This follows the suggestion made in my previous report on subsensing. This has one undesirable consequence: it does not preserve the lexicographer's intended order of subentries. Moreover, if both subentries and subsenses are present in the dictionary, it does not preserve their intended relative order either.

In option 2, the relations are recorded in the parent. This preserves the lexicographer's intended order on all elements. Option 2 is the option I am suggesting to LEXIDMA, for both subentries and subsenses.