Notice

This paper is the author's draft and has now been published officially as:

Beuls, Katrien (2011). Construction Sets and Unmarked Forms: A Case Study for Hungarian Verbal Agreement. In Luc Steels (Ed.), *Design Patterns in Fluid Construction Grammar*, 237–264. Amsterdam: John Benjamins.

BibTeX:

```
@incollection{beuls2011construction,
   Author = {Beuls, Katrien},
   Title = {Construction Sets and Unmarked Forms:
            A Case Study for {Hungarian} Verbal Agreement},
   Editor = {Steels, Luc},
   Pages = {237--264},
   Booktitle = {Design Patterns in {Fluid Construction Grammar}},
   Publisher = {John Benjamins},
   Address = {Amsterdam},
   Year = {2011}}
```

Construction Sets and Unmarked Forms: A Case Study for Hungarian Verbal Agreement

Katrien Beuls

Abstract

Construction application can be made more efficient by organizing constructions into sets and by imposing an ordering on when a construction set should be considered. This technique gives us moreover a way to handle unmarked cases, which are abundant in all the world's languages. This paper introduces a non-trivial case study to introduce and illustrate the utility of construction sets, namely Hungarian verbal agreement, which is part of the Hungarian system for expressing argument structure. Hungarian verbal agreement is interesting because it has a dual conjugation system with mono-personal and poly-personal agreement, i.e. agreement with subject only or with subject and object. The choice which system is chosen depends on complex syntactic and semantic considerations. Moreover the morphemes chosen to express agreement and case marking depend on many factors, including the phonological properties of the stem. This chapter therefore illustrates not only how construction sets are useful but also how construction grammar can take multiple linguistic levels into account.

1. Introduction

Fluid Construction Grammar supports two ways to organize constructions in a construction inventory. They are either grouped into sets, so that one set can be considered before the next one, or they are organized in a network, so that one construction can prime or take precedence over others. The present chapter focuses on the first topic, whereas the organization of constructions in networks is considered in the next chapter (Wellens, 2011).

Constructions are typically grouped in a set according to the nature of the work that their members have to carry out (Croft & Cruse, 2004). Some are dealing with the lexical meaning of individual words, others combine multiple words based

on their functions, others add grammatical meaning by attaching morphological endings to a word such as the use of the regular *-s* for plural entities in English, etc. (Gerasymova et al., 2009). This kind of grouping is useful both from a *processing* point of view, and from an *implementational* point of view.

In terms of processing, the benefits are the following:

- In parsing and production, the FCG-interpreter has to go through all constructions in the inventory in order to find the one that could usefully apply, and if there is more than one, a branching of the search path must be organized. The optimization of language processing can therefore take two forms: either the process of finding constructions that might apply could be optimized or the number of branchings that need to be explored could be minimized. Construction sets help with the first issue. Only constructions belonging to a specific set are considered at a given point in processing, before the next set is considered.

- As discussed in Steels & van Trijp (2011), the FCG-interpreter uses a meta-level to diagnose any possible problems with the routine application of constructions and possibly runs repair strategies to deal with these problems. The use of construction sets makes it possible to apply such diagnostics and repairs after each construction set. Processing errors can thus be discovered early on in the processing pipeline.

But there are also advantages from an implementational point of view.

- By organizing constructions into sets, the overall structure of the grammar becomes clearer and grammar design can focus on different construction sets. This helps to manage the inevitable complexity of working out real grammars (Steels & Wellens, 2006).

- The main focus of this paper is on the organization and ordering of linguistic processing as a way to deal with unmarked forms in language. The constructions that deal with marked forms can be grouped together in a set and operate before those dealing with unmarked forms are even considered.

Verbal agreement in Hungarian is known to be an extraordinary complex linguistic phenomenon (MacWhinney & Pléh, 1997) and is therefore an excellent challenge to test the representational and computational adequacy of FCG in general and the utility of construction sets in particular. Before tackling the grammar design, this chapter gives a brief overview of the Hungarian verbal agreement system.

Our goal is of course not to deal with all of Hungarian grammar, but to consider an interesting grammar fragment in which the phenomena of interest here arise.

2. Hungarian Verbal Agreement

In most Indo-European languages verbal endings agree with the subject of the verb that is expressed by a clause (Siewierska & Bakker, 1996). There exist however languages that seem not only to mark their verbs by subject-verb agreement but to display object-verb agreement, as is the case with Hungarian (Kiss, 2002). Belonging to the Uralic family tree, this language systematically marks both the person feature of the subject and that of the object of an event. So-called *poly-personal* agreement is found in Hungarian when the following *syntactic* (i), *semantic* (ii) conditions are satisfied:

(i) The semantic direct object is syntactically marked with the *accusative* case.[1]

(ii) The direct object is a *definite* referent in the context that is removed *further* away from the deictic center than the subject.

(iii) In addition to syntax and semantics playing a role in the decision-making process that precedes the choice of a verbal conjugational suffix, the *phonological* properties of the verb itself also need to be taken into account, such as:

  (a) the *phonological structure* of the verb stem actually limits the range of suffixes that can follow it, and

  (b) the *main vowel* in the verb stem should always belong to the same phonological class (front/back) as the vowel in the suffix.

The distinction between mono-personal (i.e. only *one* person feature matters) and poly-personal (i.e. *two* features matter) verbal agreement requires several linguistic issues to be taken into consideration before the computational implementation can be explained. This section sheds light on the nature of the three linguistic levels that play a role when conjugating verbs in Hungarian.

1. In Hungarian the object can take other cases, such as the partitive, for instance.

2.1. Semantics

The traditional answer to the dual architecture of the Hungarian conjugational paradigm is situated in the absence or presence of a definite object (Törkenczy, 2005). Linguists therefore often speak of definite vs. indefinite conjugation . The latter occurs with verbs that either have an indefinite object or no object at all (e.g. *he is reading (a book).*), whereas the former is found when the object is definite (e.g. *He is reading the book.*). This difference is illustrated in Table 1.

Table 1. *Definite vs. indefinite conjugation (I am reading a/the book).*

		conjugated verb	object
intransitive verb		olvas**-ok** (*indef.*)	∅
transitive verb	with indefinite object	olvas**-ok** (*indef.*)	egy könyvet
	with definite object	olvas**-om** (*def.*)	a könyvet

However, the previous explanation does not take into account the fact that transitive verbs that take other definite objects, such as pronouns in first and second person, do not always follow the definite conjugation. The following events that take place between two *definite* first and second person participants instantiate endings that belong to both traditional conjugational paradigms, rather than just the definite conjugation as one would expect (definite object, accusative case). It is only Example (2) that has a definite suffix. This ending is a combination of the *-l-* morpheme and the *-ek* morpheme, referring to the direct object and the subject of the verb respectively. In this sense, *-lek* is thus a *poly-personal* marker. The *S* in the examples stands for "Subject", whereas the *DO* denotes "Direct Object".

(1) *Szeret-sz en-gem.*
love-(2sg.**S**) me-ACC
'You love me.'

(2) *Szeret-l-ek té-ged.*
love-(2sg.**DO**-1sg.**S**) you-ACC
'I love you.'

Because both direct objects are definite referents either in the physical or in the discourse context, the distinction between poly- and mono-personal conjugation needs to be found elsewhere. The definiteness criterion seems not to be sufficient.

When the action takes place between 2nd and 3rd person event participants, the following verbal behavior results:

(3) *Szeret-ed* *őt.*
love-(2sg.**S**.def**DO**) he-ACC
'You love him.'

(4) *Szeret-∅* *té-ged.*
love-(3sg.**S**) you-ACC
'He loves you.'

Whereas the ending in Example (4) is clearly an instance of the mono-personal conjugation, the *-ed* morpheme in (3) marks the verb with the person feature of the subject and the definiteness feature of the direct object. For clarity, these instances of the traditional definite conjugation are also referred to as poly-personal. Again, a similar deviation between mono- and poly-personal conjugation is found in events with 1/2 or 2/3 person event participants.

We claim that the key to the agreement puzzle lies in the difference in verbal ending between sentences (1) and (3): a mono-personal ending occurs when the object is 1st person singular, whereas a 3rd person singular object requires a poly-personal ending. The pattern that becomes visible here is related to the deictic relations between the participants in the action. In deictic terms, the speaker can be referred to as the *origo* of the discourse (Bühler, 1934). No matter who the participants in the action are, the relationship between them always has to be calibrated on the origo, origo-near, origo-far axis, illustrated in Figure 1. In other words, poly-personal agreement endings are found when the event action takes place in an *outward* fashion (directionally lower right in Figure 1), that is, away from the origo, whereas mono-personal endings are found when the action moves *inward* (directionally upper left in Figure 1). When both participants are situated in the same deictic field (reflexive actions or actions in the third person domain), both participants are marked, and poly-personal endings are used.

2.2. Syntax

In Hungarian, there is not just one case that always expresses the direct object of the action, but it is dependent upon being fully or only partially involved in the action expressed by the verb. Partial involvement is expressed by a partitive

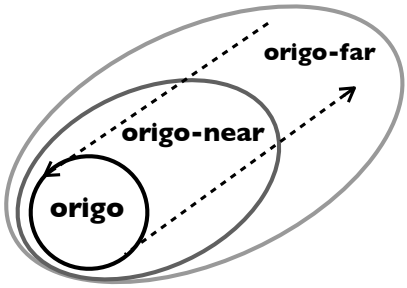


Figure 1. *Directions in the interaction between two participants*

case marker, whereas complete involvement requires the accusative. The following example illustrates that the choice of the syntactic case feature is vital for the realization of the verbal ending. While the accusative ending in (5) causes a polypersonal verb ending, the use of the partitive ending in (6) does not have the same consequence.

(5) *Et-te* *a* *sütemény-t.*
eat-past-(3sg.**S**.def**DO**) the pastry-**ACC**
'He/She ate the pastry.'

(6) *Ev-ett* *a* *sütemény-ből.*
eat-(3sg.**S**)-past the pastry-**PART**
'He/She ate some of the pastry.'

2.3. Phonology

The rich morphology that characterizes the Hungarian language stems partially from the phonological properties that the language displays. The choice of which morphological ending to attach to a particular noun or verb depends on two phonological properties: the stem's last phoneme(s) and the stem's main vowel. The phonological processes that are directly affected by these properties are *assimilation* and *vowel harmony* . They are discussed in the following sections.

2.3.1. *Assimilation*

The choice of a verbal ending that fits a particular grammatical agreement pattern (e.g. `[definite poly-personal accusative]`) is also related to the phonological structure of the verb stem itself. The three major classes to which *verbal stems* belong in Hungarian are:

1. regular stems (e.g. *szeret*, 'to like')
2. stems ending on a sibilant (e.g. *olva**s***, 'to read')
3. stems ending on a consonant cluster (e.g. *csu**kl**-ik*, 'to hiccup')

The verbal suffix is always chosen in accordance with the morphological class to which the verbal stem belongs, as illustrated by the following example.

(7) *Szeret-**sz*** *egy könyv-et.*
like-(2sg.**S**) a book-ACC
'You like a book.'

(8) *Olvas-**ol*** *egy könyv-et.*
read-(2sg.**S**) a book-ACC
'You read a book.'

Furthermore *nominal stems* are characterized by slightly different, stem inherent morphological properties. Nominal stems can belong to one of three kinds:

1. regular stems (e.g. 'virág**ot**', 'flower-ACC')
2. stems ending on a vowel *or* a vowel followed by a sibilant/nasal/glide (e.g. 'tortá**t**', 'cake-ACC')
3. stems containing a lowering vowel (e.g.'könyv**et**', 'book-ACC')

Lowering stems do not themselves change, but they do cause some irregularities in the choice of suffix variants. These stems are followed by two irregularities: (i) the suffix-initial *unstable* vowel is *-a/e* instead of the regular *-o/e/ö*, and (ii) the *unstable* vowel of the accusative is retained even after stem-final consonants that otherwise cause the deletion of unstable vowels. Compare for instance the stem consisting of a vowel followed by a nasal: *pénz**t*** "money + ACC" and *tehen**et*** "cow + ACC", where the latter contains a lowering vowel in its stem. The list of lowering stems is arbitrary and contains very common words.

2.3.2. *Vowel harmony*

The phonological system of the Hungarian language is characterized by so-called vowel harmony, meaning that most suffixes harmonize with the stem to which they are attached. Consequently, most suffixes exist in two or three alternative forms that differ in the suffix vowel, and the selection of the suffix form is determined by the stem vowel(s). Vowels either belong to the front set (i, í, ü, ű, e, é, ö, ő) or the back set (u, ú, o, ó, a, á), where front vowels can be either rounded (ü, ű, ö, ő) or unrounded (i, í, e, é). Suffixes may be non-harmonic (i.e. just one form, e.g. *-kor*, 'at') and may have a harmonic two-form (e.g. *-ban/ben* 'in') or a harmonic three-form (i.e. including the lip rounding distinction: *-hoz/-hez/-höz*, 'to').

The following examples illustrate the importance of vowel harmony since the same semantic and syntactic conditions do not automatically lead to exactly the same formal ending. Within the poly-personal paradigm, additional morphological, as well as phonological decisions, have to be made.

(9) *Szeret-em* *a* *tortá-t.*
like-**front-unrounded**-(1sg.S.defDO) the cake-ACC
'I like the cake.'

(10) *Utál-om* *a* *tortá-t.*
hate-**back**-(1sg.S.defDO) the cake-ACC
'I hate the cake.'

This section has shown that there are two main aspects related to the three modules that have to be taken into consideration when implementing the Hungarian agreement system in FCG. Firstly, there are different semantic agreement patterns that occur according to the definiteness of the object and the direction of the action and that are related to the syntactic case constraints imposed on the object. Secondly, once the decision as to which conjugational paradigm to use has been made, a whole range of morpho-phonological constraints needs to be taken into account so that the appropriate morpheme can be selected. In order to simplify the discussion, we first consider the choice of the agreement pattern using abstract morphemes as summarized in Table 2. Only later will we look at how these abstract morphemes are turned into concrete morphemes using the kind of morpho-phonological constraints found in Hungarian.

	mono-personal		poly-personal	
	sg	pl	sg	pl
1	*-mono1sg*	*-mono1pl*	*-poly1sg*	*-poly1pl*
2	*-mono2sg*	*-mono2pl*	*-poly2sg*	*-poly2pl*
3	*-mono3sg*	*-mono3pl*	*-poly3sg*	*-poly3pl*

Table 2. *Abstract conjugational endings for present events that take mono- or poly-personal agreement.*

3. Operationalization - Part I

The constructions needed for the first steps in processing are going to be grouped into the following different subsets:

1. The *lexical construction set* groups all lexical constructions. The same templates (`def-lex-cxn`, `def-lex-skeleton`, `def-lex-cat`, etc.) are used as already seen in earlier chapters (Steels, 2011).

2. The *functional construction set* groups all functional constructions that map lexical categories and syntactic types to syntactic and semantic functions. They also use the same templates as introduced before.

3. The *grammatical construction set* groups all phrasal constructions that build nominal phrases and sentences using templates like `def-phrasal-cxn`, `def-phrasal-agreement`, etc. As grammatical constructions concern all constructions that deal with more than one unit in the transient structure, both phrasal and argument structure constructions are considered here.

4. The *morphological construction set* groups constructions work out the morphological consequences of argument structure and agreement choices.

A construction can be added to a particular set in two ways. You can either explicitly state the construction set you want to add a construction to (by using the keyword `:cxn-set` in the `def-lex-skeleton`) or you can rely on its instantiating template definition. For example, the `def-lex-cxn` template that starts the building of a new lexical construction puts this construction in the lexical construction set,

the `def-morph-cxn` puts the new morphological construction in the morphological construction set, etc. A configuration setting in the general `def-constructions` template allows you to enable the use of construction sets in the search process.

When you decide to make use of construction sets, you should be aware that there is a particular ordering that is imposed on the sets, which can be different depending on the direction of processing (production or parsing). In parsing, the construction sets are considered in the following order:

`lexical` → `morphological` → `functional` → `grammatical`

In production, the processing order is similar, only the morphological constructions are now moved to the end of the processing pipeline, as it is the grammatical constructions that fill in the feature attributes (such as number, person, etc.) they need. The order thus looks as follows :

`lexical` → `functional` → `grammatical` → `morphological`

Each of these construction sets is now briefly introduced using the example sentence in (11). The verbal ending - `poly3sg` belongs to the poly-personal conjugation paradigm because the object is *definite*. It is situated on the same deictic space (*3rd person*) as the subject, and it is marked with the *accusative* case.

(11) *János szeret-poly3sg a torta-acc*
John like-(3sg.S.defDO) the cake-ACC
'John likes the cake.'

3.1. Lexical Construction Set

Lexical constructions map predicate argument expressions onto words in production and the other way around when parsing. The `def-lex-cxn` template, as defined in Steels (2011), is re-used here. An explanation of the correct reading of the meaning predicates used here can also be found in Steels (2011), while similar verbal meaning predicates are used in van Trijp (2011). The lexical skeleton of each lexical item is already expanded with category information. The example lexicon is created by another template `def-constructions`:

```
(def-constructions example-lexicon
 (def-lex-cxn john-cxn
         (def-lex-skeleton john-cxn :cxn-set lex
    :meaning (== (john ?john-set ?base-set))
    :string "janos")
  (def-add-lex-cat john-cxn
    :sem-cat (==1 (individual person))
    :syn-cat (==1 (lex-cat proper-noun))))
  (def-lex-cxn the-cxn
          (def-lex-skeleton the-cxn :cxn-set lex
    :meaning (== (unique-definite ?indiv ?base-set))
    :string "a")
   (def-add-lex-cat the-cxn
    :syn-cat (==1 (lex-cat article) (is-definite +))
    :sem-cat (==1 (determination definite))))
  (def-lex-cxn cake-cxn
   (def-lex-skeleton cake-cxn :cxn-set lex
    :meaning (== (cake ?cake-set ?base-set))
    :string "torta")
   (def-add-lex-cat cake-cxn
    :syn-cat (==1 (lex-cat noun))
    :sem-cat (==1 (class object))))
 (def-lex-cxn like-cxn
   (def-lex-skeleton like-cxn :cxn-set lex
            :meaning (== (like ?event ?base-set)
                 (like-arg1 ?event ?agent)
                 (like-arg2 ?event ?object))
            :string "szeret")
   (def-add-lex-cat like-cxn
            :syn-cat (==1 (lex-cat verb))
    :sem-cat (==1 (sem-function relator)))))
```

The processing of the example sentence (see (12)) with an inventory of constructions is already possible but clearly not yet complete:

1. The morphological endings are not processed, because their form is not yet part of the construction inventory.

2. The meaning that is returned after parsing does not express the participant relations that are necessary to interpret the poly-personal agreement operation. The variables of the `like-arg1` and `like-arg2` predicates are not equal to those of the `john` and `cake` predicates respectively:

```
((john ?john-set-11 ?base-set-297)
 (cake ?cake-set-84 ?base-set-299))
 (like ?event-117 ?base-set-300)
 (like-arg1 ?event-117  ?agent-117)
 (like-arg2 ?event-117 ?object-29)
 (unique-definite ?indiv-68 ?base-set-298))
```

3. The resulting structure does not yet incorporate the phrasal structure present in the determined noun phrase "a torta" causing sentences such as "szeret a janos torta" or "torta a janos szeret" (lit. "the john loves cake/cake loves the john") to occur frequently in production.

The next section expands the construction inventory to deal with these issues.

3.2. Functional and Grammatical Construction Sets

Functional constructions map available category information into functional information, which then provides the input that is required by grammatical constructions. An example of a template is included below for the `proper-noun-cxn`. Similar to word items that are characterized by the `(lex-cat noun)` attribute, proper nouns also translate into nominal syn-functions. This fact allows these constructions to be treated in a syntactically similar way later on.

```
(def-fun-cxn proper-noun-cxn :cxn-set fun
    :syn-cat (==1 (lex-cat proper-noun))
    :syn-function nominal
    :sem-cat (==1 (individual person))
    :sem-function unique-identifier)
```

Nominal phrases are referring expressions (e.g. "john", "the book") and they are used as input for other grammatical constructions dealing with argument structure and agreement. The phrasal skeleton that is used to create for instance a `proper-nominal-phrase-cxn` builds on the semantic and syntactic functions that were provided by the functional construction set and instantiates

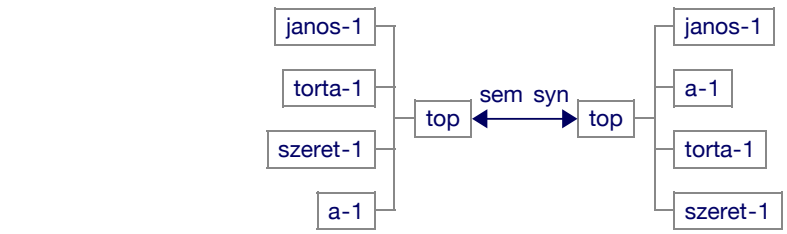
a phrasal "parent" unit (e.g. `?nominal-phrase` in the template below). The other phrasal construction that is needed to reproduce the example sentence is the `determiner-nominal-phrase-cxn`, which does not only contain sem-function and phrase-type information in this case but also a specification of the order of the constituent units.

```
(def-phrasal-cxn proper-nominal-phrase-cxn
    (def-phrasal-skeleton proper-nominal-phrase-cxn
        :cxn-set gram
        :phrase
        (?nominal-phrase
         :sem-function referring
         :phrase-type nominal-phrase)
        :constituents
        ((?proper-nominal-unit
          :sem-function unique-identifier
          :syn-function nominal)))))
```

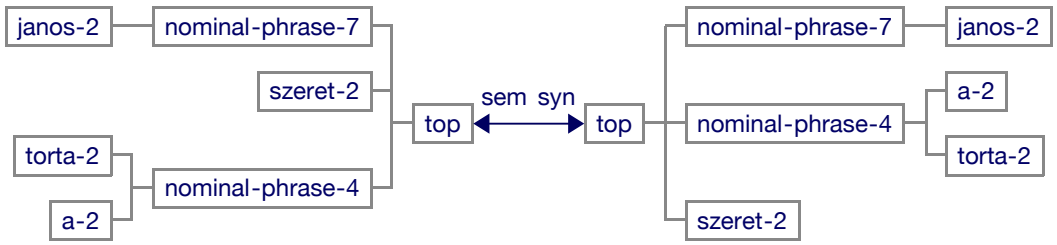
```
(def-phrasal-cxn determiner-nominal-phrase-cxn
    (def-phrasal-skeleton determiner-nominal-phrase-cxn
        :cxn-set gram
        :phrase
        (?nominal-phrase
         :cxn-form (== (meets ?determiner-unit
                              ?nominal-unit))
         :sem-function referring
         :phrase-type nominal-phrase)
        :constituents
        ((?determiner-unit
          :sem-function reference
          :syn-function determiner)
         (?nominal-unit
          :sem-function identifier
          :syn-function nominal))))
```

The resulting linguistic structure after re-parsing the example sentence (see (12)) shows that this step was able to deal with the issue of phrasal structures (see Figure 2). The two remaining issues are handled by the next two steps. First, the variable

equalities that were missing in the meaning predicates are tackled by the introduction of a transitive construction (also belonging to the grammatical construction set). Second, morphological constructions are added to the construction inventory so that the remaining forms can be processed.



(a) only lexical cxns



(b) lexical, functional and phrasal cxns

Figure 2. *Resulting linguistic structures after implementing Step 1 and Step 2, respectively.*

3.3. The Grammatical Construction Set (cont.)

Because Part 1 only takes syntactico-semantic agreement patterns into account (i.e. still excluding morpho-phonological ones), there are just four feature attributes to consider here: `person-direction`, `definiteness`, `number` and `case`. Therefore, every time a transitive meaning needs to be expressed, the transient verb unit collects all the necessary values from these feature attributes so the final verb form follows the appropriate agreement paradigm, be it mono- or poly-personal. There is thus again a copying operation taking place between units on the same level in the transient structure. This time, attributes from the subject (and possibly the object) are copied to the verb.

The lexical construction for a verb does not provide these attributes a priori. Because a verb is not necessarily conjugated (e.g. infinitive), we need another con-

struction to merge the conjugational features into the transient structure, which is precisely what the argument structure construction does. At this stage, the current transient structure consists of three subunits (see Figure 2b): one verb unit and two nominal units. As the values of the three agreement attributes are extracted from both nominal units (the `definiteness` and `case` values from the direct object, and the `person-direction` values from both subject and direct object and the `number` value from the subject only), the semantic roles of these units must be detected before the percolation takes place.

1. The appropriate determination of the semantic roles is done by the template `def-add-roles`, which can be used within the familiar `def-phrasal-cxn` template. The introduction of the `args` feature in the lexicon assures the correct linking of the semantic valency (`sem-val`) information and the lexical meaning of the units (Steels et al., 2005; van Trijp, 2011). The `def-add-phrasal-linking` template (introduced in (Steels, 2011)) is reused here to enforce the recruitment of the event participants from one and the same context.

   The syntactic valency (`syn-val`) attribute has a similar task in parsing by linking the unit names of the subject and direct object fillers with the predicate unit. The added case attribute in the nominal units assures the correct mapping between a particular semantic role, such as agent, and the corresponding syntactic role, namely the subject role.

2. The correct handling of the agreement attributes needed in the verb unit is regulated by the `def-add-phrasal-agreement` template. As mentioned above, a lateral percolation from the subject and direct object units towards the predicate unit is needed here. The predicate unit gets its number feature from the subject unit, its definiteness and case features from the direct object unit and the values of person- direction feature from both. The order in the person-direction feature is important: the action always runs from the subject towards the direct object. The `transitive-cxn` is in fact restricted to handling the accusative direct- object.

```
(def-phrasal-cxn transitive-cxn
  (def-phrasal-skeleton transitive-cxn
      :cxn-set gram
      :phrase
      (?sentence
       :sem-function description
       :phrase-type sentence)
      :constituents
      ((?subject
        :sem-function referring
        :phrase-type nominal-phrase)
       (?predicate
        :sem-function relator
        :lex-cat verb)
       (?direct-object
        :sem-function referring
        :phrase-type nominal-phrase)))
  (def-add-phrasal-agreement transitive-cxn
    (?subject
     :syn-cat (==1 (number ?number)
                   (person ?person-subject)))
    (?direct-object
     :syn-cat (==1 (is-definite ?definiteness)
                   (person ?person-object)
                   (case accusative)))
    (?predicate
     :syn-cat (==1 (number ?number)
                   (is-definite ?definiteness)
                   (person-direction ?person-subject
                                     ?person-object))))
  (def-add-phrasal-linking transitive-cxn
    (?subject
     :args (?agent ?context))
    (?predicate
     :args (?event ?context))
    (?direct-object
     :args (?patient ?context)))
```

```
(def-add-roles transitive-cxn
  (?subject
   :sem-role agent
   :syn-role (==1 (is-subject +)
                  (is-direct-object -)))
  (?predicate
   :syn-val (==1 (subject-filler ?subject)
                 (direct-object-filler
                  ?direct-object))
   :sem-val (==1 (agent ?event ?agent)
                 (patient ?event ?patient)))
  (?direct-object
   :sem-role patient
   :syn-role (==1 (is-subject -)
                  (is-direct-object +)))))
```

Parsing the example sentence “janos szeret-subj3sg a torta-acc” (abstract endings, no accents) now returns the following meaning:

```
((john ?agent-54 ?base-set-124)
 (unique-definite ?patient-41 ?set-8)
 (cake ?set-8 ?base-set-124)
 (like ?event-54 ?base-set-124)
 (like-arg1 ?event-54 ?agent-54)
 (like-arg2 ?event-54 ?patient-41))
```

Elements such as `john`, `cake` and `like` are all part of the same context (`base-set-124`). The determiner selects one individual element from an already filtered set (in this example, a set with cakes). The verb relates the agent and patient variables through its threefold predicate meaning.

3.4. The Morphological Construction Set

Although both the verb unit and nominal units now contain all the semantic and syntactic information that is necessary to carry out the necessary agreement operations, no suffixes have yet been added to the final linguistic structure. In order to express this agreement information in production or to extract the necessary attribute values in parsing (i.e. when the meaning predicates do not determine the

semantic agent and patient roles), morphological constructions need to be added to the construction inventory. These morphological constructions are needed for verbal as well as nominal suffixes and are created with the `def-morph-cxn` template as follows:

```
(def-morph-cxn poly3sg-cxn :cxn-set morph
               :suffix"-poly3sg"
               :stem
               (?verb-unit
                :syn-cat (==1 (is-definite +)
                              (person-direction 3 3)
                              (number (+))
                              (case (==1 (acc +) (nom -) (part -)))
                              (lex-cat verb))))
```

```
(def-morph-cxn accusative-cxn :cxn-set morph
               :suffix "-acc"
               :stem
               (?noun-unit
                :syn-cat (==1 (case (==1 (acc +) (nom -) (part -)))
                              (syn-function nominal))))
```

Morphological constructions consist of two syntactic poles instead of a semantic and a syntactic one (see (Gerasymova, 2012)). The `syn-cat` information that is supplied in the above template belongs to the matching unit of the left pole of the expanded morphological construction. Consequently, this agreement information should already be fully specified in the verb unit in production (i.e. when processing runs from left to right pole). The string is added to the top unit of the right pole as a tag.

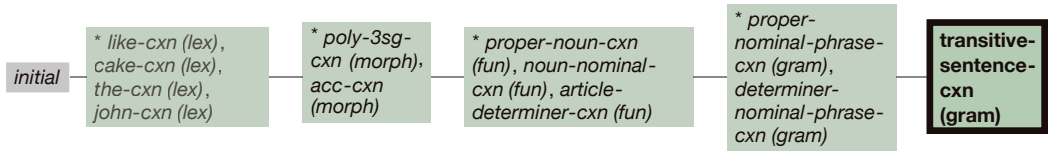
The `def-morph-cxn` template contains the four major agreement features: `is-definite`, `person-direction`, `number` and `case`. The nominal lexical constructions are therefore expanded with corresponding `definiteness`, `person` and `number` attributes, so that the `def-add-lex-cat` template now looks as follows for the lexical items that provide agreement information:

```
(def-add-lex-cat john-cxn
    :syn-cat (==1 (lex-cat proper-noun)
                  (number (+)) (is-definite +) (person 3))
    :sem-cat (==1 (individual person)))
(def-add-lex-cat the-cxn
    :syn-cat (==1 (lex-cat article) (is-definite +))
    :sem-cat (==1 (determination definite)))
(def-add-lex-cat cake-cxn
    :syn-cat (==1 (lex-cat noun)
                  (number (+)) (person 3))
    :sem-cat (==1 (class object)))
```

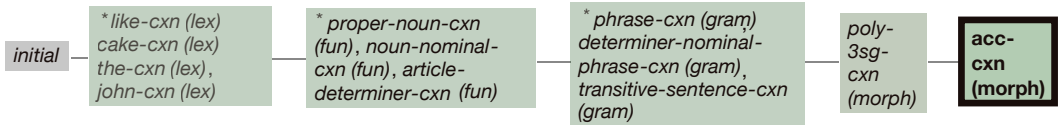
Because nouns are always unmarked for definiteness, the noun phrase to which they belong receives its definiteness feature from another unit such as the determiner. It is the `def-add-phrasal-agreement` template, incorporated into the `def-phrasal-cxn template`, that regulates this kind of feature percolation between phrase and word level. The following snippet is added on top of the phrasal skeleton of the already existing `determiner-nominal-phrase-cxn`.

```
(def-add-phrasal-agreement
  determiner-nominal-phrase-cxn
   (?nominal-phrase
     :syn-cat (==1 (is-definite ?definiteness)
                   (number ?number)
                                    (person ?person)))
   (?determiner-unit
     :syn-cat (==1 (is-definite ?definiteness)))
   (?nominal-unit
     :syn-cat (==1 (number ?number)
                                    (person ?person)))))
```

The construction inventory now contains 12 constructions that are all used to process the example sentence. Figure 3 shows that all constructions are needed in parsing as well as production, although in a slightly different order. Whereas the morphological constructions apply second in parsing, they all occur at the end of the production pipeline. This explicit order between different sets of constructions is enforced in the way that morphological constructions trigger as soon as a lexical



(a) parsing



(b) production

Figure 3. *The resulting application processes with a complete construction inventory.*

item appears in the transient structure and then fill in agreement information that is provided by the `transitive-cxn`.

4. Operationalization - Part II

The first part of the operationalization stage took only the semantic and syntactic constraints that were needed to actualize verbal agreement in Hungarian into consideration. This second part adds now morpho-phonological constraints so that by the end of the section, the FCG system will process complete Hungarian sentences. We start Part 2 of the implementation from within the morphological inventory, as the abstract suffixes from Part 1 are this time replaced by a new range of context sensitive morphemes. Moreover, the lexicon is also further extended so that the verbal and nominal lexemes can host the new array of suffixes.

The example sentence that will be considered now looks as follows:

(12) *János szeret-i* *a* *torta-t*
John like-(3sg.S.defDO) the cake-ACC
'John likes the cake.'

	mono-personal		poly-personal	
	sg	pl	sg	pl
1	*-ok/-ek/-ök*	*-unk/-ünk*	*-om/-em/-öm*	*-juk/-jük*
			-lok/-lek/-lök	*-⋆uk/-⋆ük*
2	*-sz*	*-tok/-tek/-tök/*	*-od/-ed/-öd*	*-játok/-itek*
	-ol/-el/-öl/-asz/-esz	*-otok/-etek/-ötök*		*-⋆átok*
3	∅	*-nak/-nek/-anak/-enek*	*-ja/-i*	*-ják/-ik*
			-⋆a	*-⋆ák*

Table 3. *Present tense verb endings. The ⋆ indicates that the previous consonant has to be doubled.*

4.1. Re-defining the Morphological Constructions

The initial abstract verbal suffixes (cf. Table 2) get replaced by an extensive conjugational table containing no less than 44 morphemes (cf. Table 3). This number is due to the fact that each grammatical feature bundle `[person/number/conjugation-type]` has almost always multiple allomorphic realizations. The first row in the table contains suffix variations that are present due to the vowel harmony that needs to be maintained between the stem and the suffix vowel. The second row in the table illustrates additional variation due to morpho-phonological properties of the verb stem. The stars in the table indicate that the final consonant needs to be doubled, which occurs when speakers are confronted with phonetic assimilation . For example, instead of writing ”olva**s-j**átok” Hungarian spelling has adopted itself to the phonetic writing “olva**s-s**átok”, when expressing the 2nd person plural of the poly-personal conjugation of “to read”.

Additionally, the introduction of accusative case markers is characterized by a rapid increase in the number of endings that are used in processing. Taking into account the vowel-harmony and the morpho-phonological properties of the stem, Table 4.1 includes five different physical endings that are representing not less than nine different morpho-phonological constraints. More information on the morpho-phonological details of Hungarian verbal and nominal stems is included in Sections 2.3 and 2.3.2.

Real Hungarian morphology is introduced into the construction inventory by (i) replacing the abstract ending with its Hungarian equivalent (e.g. *-poly3sg* $\rightarrow$

	regular stem	ends on vowel/vowel+sng	lowering stem
back	*-ot*	*-t*	*-at*
front-unrounded	*-et*		*-et*
front-rounded	*-öt*		

Table 4. *Accusative case markers. Vowel-harmony feature represented by the rows, stem properties by the columns.*

-i) and (ii) adding the appropriate phonological properties by means of two attributes that function as values of the `phon-cat` feature: `vowel-harmony` and `stem-properties`. The templates below illustrate this addition.

```
(def-morph-cxn poly-i-cxn
    :suffix "-i"
    :stem
    (?verb-unit
     :syn-cat
     (==1 (is-definite +)
          (person-direction 3 3)
          (number (+))
          (lex-cat verb))
     :phon-cat
     (==1 (vowel-harmony
           (==1 (back -)
                (front-unrounded ?i-front-unrounded)
                (front-rounded ?i-front-rounded)))
          (stem-properties
           (==1 (regular-stem +)
                (ends-on-sibilant -)
                (ends-on-consonant-cluster -))))))
```

```
(def-morph-cxn acc-t-cxn
    :suffix "-t"
    :stem
    (?noun-unit
     :syn-cat
     (==1 (syn-role (==1 (is-subject -)
                         (is-direct-object +)))
          (case accusative)
          (syn-function nominal))
     :phon-cat
     (==1 (vowel-harmony
           (==1  (back ?acc-back)
                 (front-unrounded ?acc-front-unrounded)
                 (front-rounded ?acc-front-rounded)))
          (stem-properties
           (==1 (regular-stem -)
                (ends-on-vowel/vowel+sng +)
                (lowering-stem -))))))
```

The values of the new attributes have the form of feature matrices (van Trijp, 2011), which means that some values might remain undetermined, such as whether the stem vowel is front-rounded or front-unrounded (cf. template for `poly-i-cxn`). The complete value is filled in by the particular verb stem to which the suffix gets attached.

4.2. Adapting the Lexicon

Before the example sentence is processed again with this enriched morphology, the lexical entries for nominal and verbal stems also need to be expanded with `phon-cat` information. In contrast with morphological constructions, the values of the `vowel-harmony` and the `stem-properties` attributes are always fully specified in lexical constructions, since a particular noun or verb can only have one main stem vowel (back/front-unrounded/front-rounded) and one main morpho-phonological stem property (regular/ends-on-vowel/ vowel+sng/lower-stem). The following template shows this constraint.

```
(def-add-lex-cat cake-cxn
      :phon-cat
      (==1 (vowel-harmony
            (==1 (back +)
                 (front-unrounded -)
                 (front-rounded -)))
           (stem-properties
            (==1 (regular-stem -)
                 (ends-on-vowel/vowel+sng +)
                 (lowering-stem -)))))
```

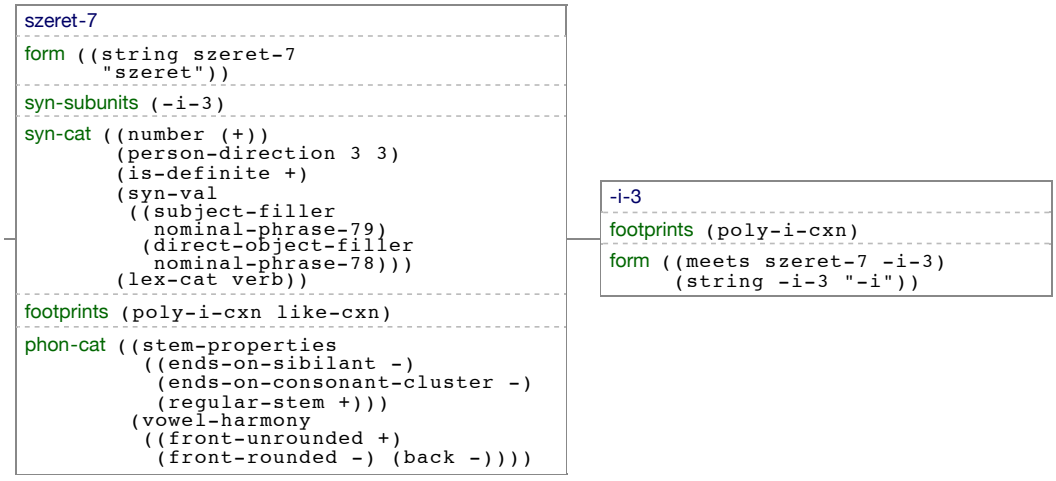


Figure 4. *The syntactic pole of the predicate unit after parsing "janos szeret -i a torta -t" .*

Figure 4 shows that parsing the example sentence is successful, and the predicate unit has nicely absorbed the syntactic and phonological information coming from the morpheme in order to construct a meaning on the semantic pole. Furthermore, in production, the inherent phonological properties of the nominal units (i.e. verb and nouns) (i.e. `phon-cat` values), together with the grammatical relations added by the argument structure construction (i.e. `syn-cat` values), define the suffixes that get chosen in the conjugation of the verb and the inflection of the direct object.

Just like the "invisible" nominative case marker, the mono-personal suffix that is complementary to the 3rd person poly-personal "-i" is a so-called empty string. No

special morphological construction was introduced to cover this suffix, since there is no formal string that needs to be attached to a verb unit. The lack of such a construction poses a problem in parsing, as the necessary verbal agreement attributes are not added to the verb unit. Section 5 is devoted to the issue of *unmarked forms*.

5. Unmarked Forms

FCG designers typically encounter two types of relationships in the grammar they are formalizing:

(i) Construction-1 *requires* the application of Construction-2 in order to match the transient structure.

(ii) Construction-1 *precedes* Construction-2 in the queue, i.e. it is prioritized in the search process.

The first relationship is the one that has been discussed above. It is operationalized by the introduction of an explicit processing order between construction sets, such as morphological constructions (e.g. the `accusative-cxn`) relying on the application of grammatical constructions (e.g. the `argument-structure-cxn`) in production. In such cases, Construction-1 is a prerequisite for the correct application of Construction-2.

Sometimes it can be useful to make more fine-grained distinctions within one construction set. This is the case with unmarked forms, i.e. nouns or verbs that lack a suffix but are used in their bare lexical form. Think about the English *he walks* vs. *you walk*. In the latter case there is no explicit ending that indicates person/number agreement with the subject. The unmarked forms embody the second relationship that is often found in grammar formalization.

Unmarked forms are a big topic in mainstream generative linguistics where abstract representations of language include many "invisible" nodes and even invisible words. Because constructionist approaches typically adopt a "what you see is what you get" model of language (Goldberg & Suttle, 2010), no such "silent" syntactic units are added to the transient structure. Moreover, the addition of multiple unmarked constructions all containing empty strings would cause a major enlargement of the search space at every single processing step. Furthermore, a solution is needed that deals with semantic units that lack a surface form realization, that is, a syntactic counter part. The previous sections already pointed to the need for the following two kinds of unmarked constructions:

1. **Morphological unmarked constructions** mark either a nominative case noun unit or a 3rd person singular present verb form. Again, the `transitive-cxn` only matches in parsing on nominal units that carry a case feature, whereas in production these constructions would not affect the transient structure.

2. **Phrasal unmarked constructions** create asymmetric transient structures to cover the very frequent phenomenon of so-called "dropped" subjects. As these constructions modify the transient structure on both semantic and syntactic poles, they are functional both in production and parsing.

The most crucial question posed, when a set of unmarked constructions is explicitly added to the construction inventory, concerns the exact moment in the application process *when* they should apply. Only if there are lexical items that are not immediately followed by morphemes or if the subject of a verb form is explicitly left unpronounced, do the unmarked constructions come into play. The current implementation solves this issue by putting these unmarked constructions in a separate set that is processed after the construction set to which they are most related. For instance, the set of grammatical unmarked constructions comes after the construction set that contains all grammatical constructions in the processing pipeline. Note that although the unmarked constructions are only useful in parsing, they are nevertheless added to the production pipeline as well, in order to keep the processing bi-directional. Unmarked elements in production are inherently present in the transient structure and need not be encoded explicitly.

5.1. Unmarked Morphology

The first group of unmarked constructions, namely those that deal with morphologically unmarked forms such as the nominative and the 3rd person singular mono-personal verb form, requires the introduction of a new template `def-unmarked-morph-cxn`. This template automatically creates a new construction set (recognized by the label `unmarked-morph`) that is always processed after the regular morphological construction set, in production as well as in parsing.

Because an unmarked morphological construction typically applies when there is already a lexical unit present (either verb or nominal) that does not contain case or agreement information yet, the keyword `conditional-syn-cat` specifies the attributes that should be part of the existing `syn-cat`, and the keyword `unmarked-syn-cat` containing the `syn-cat` information is added by the unmarked construction. The construction set information regulates the order of application.

```
(def-unmarked-morph-cxn 3sg-mono-cxn
  :conditional-syn-cat (==1 (lex-cat verb))
  :unmarked-syn-cat  (==1 (person-direction 3 3)
                          (number (+))
                          (is-definite -)))
```

```
(def-unmarked-morph-cxn nominative-cxn
  :conditional-syn-cat (==1 (syn-function nominal))
  :unmarked-syn-cat  (==1 (case (==1 (nom +) (acc -)
                                     (part -)))))
```

In order to provide a better understanding of how these unmarked morphological constructions really work, the complete construction for an unmarked 3rd person singular verb form is included below. The `unmarked-syn-cat` occurs in the left pole, which means that it is information present in production but merged during parsing. In production this construction only applies, without altering the existing structure of the verb unit.

```
((?top-unit
   (syn-subunits (== ?verb-unit)))
 (?verb-unit
   (syn-cat (==1 (person-direction 3 3)
                 (number (+))
                 (is-definite -)))
   (footprints (==0 unmarked-morph-cxn)))
 ((J ?verb-unit)
  (footprints (==1 unmarked-morph-cxn))))
 <-->
((?top-unit
  (syn-subunits (== ?verb-unit)))
 (?verb-unit
  (syn-cat (==1 (lex-cat verb)))))
```

When the example sentence is parsed once more, there is one unmarked construction that shows up in the application process: the `nominative-cxn`. Figure 5 illustrates the complete process. The `nominative-cxn` applies after the other morphological constructions and the marked `transitive-cxn` before its unmarked versions. The actual order in which sets are processed in parsing is thus: `lex` $\rightarrow$

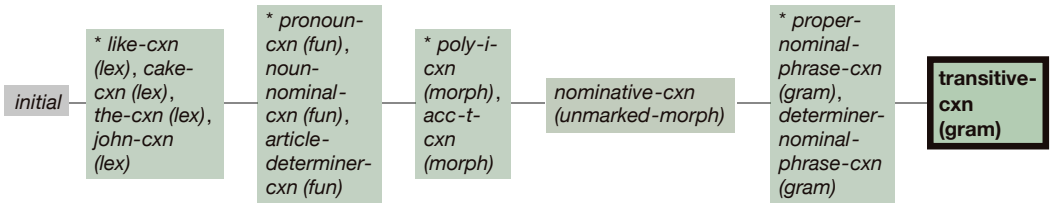


Figure 5. *The final application process after re-parsing "janos szeret -i a torta -t". The different construction sets are grouped together in search nodes.*

`fun` → `morph` → `unmarked-morph` → `gram`. Note that this time the functional construction set is processed before the morphological construction sets as the unmarked set requires the presence of a syntactic function feature attribute (namely `:conditional-syn-cat (==1 (syn-function nominal))`).

5.2. Unmarked Argument Structure

The solution for introducing a construction for the unmarked argument structure constructions differs from the unmarked constructions that manipulate just a single unit (verb or nominal unit), in that the argument structure construction actually adds a new semantic unit to the transient structure and takes care of essential semantic linking. The real issue with the unmarked argument structure constructions is that they trigger too soon in parsing. As soon as there are two subunits that fulfill the matching conditions specified on their syntactic pole, they apply and create a sentence, even though there might be form left in the top unit. Of course, now that the unmarked grammatical constructions form an additional construction set they are processed after the regular grammatical constructions so the `transitive-cxn` is always handled before its unmarked versions.

Figure 6 shows the resulting transient structure after parsing a sentence that lacks an overt subject:

(13) *Szeret* *egy torta-t*
like-(3sg.S.indefDO) a cake-ACC
'He/she likes a cake.'

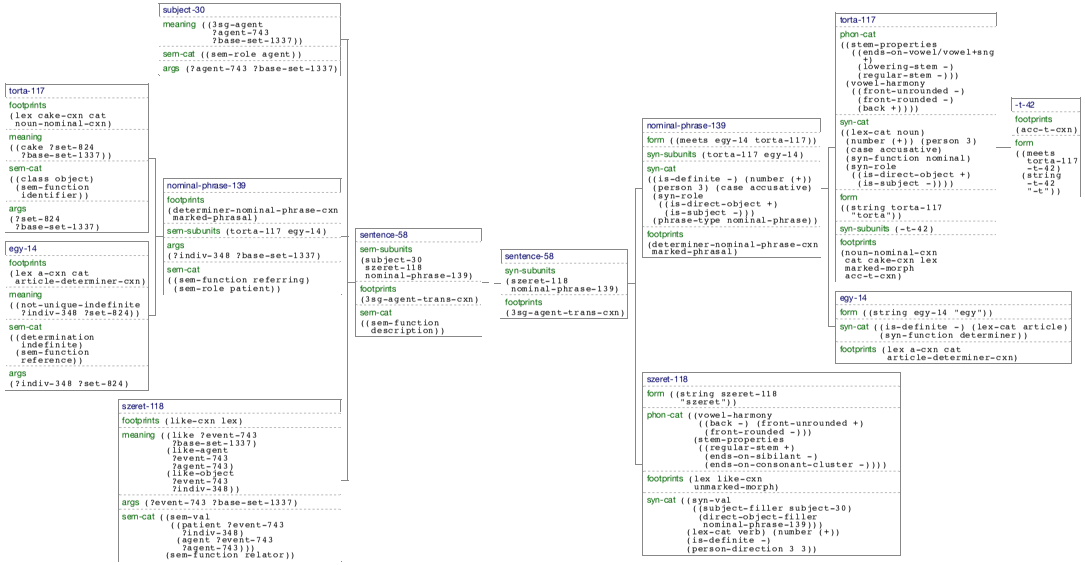


Figure 6. *The transient structure after a final parse of the example sentence without a subject and with an indefinite object: "szeret" "egy" "torta" "-t". The top unit and sentence unit are left out for clarity issues. Case and agreement information is present in all three syntactic subunits (right pole) and translates into corresponding semantic roles and valency linking on the semantic pole (left).*

The asymmetric structure indicates the presence of an agent unit on the semantic side while a subject unit is lacking on the syntactic side. Both the unmarked constructions for the subjectless transitive construction as the one for the 3rd person singular mono-personal conjugation applied here. As no agent unit could be found in the transient structure, the `transitive-cxn` could not apply here (as opposed to Figure 5) and the unmarked grammatical constructions had a chance to apply.

5.3. Footprints

As the parsing process in Figure 5 shows, the `transitive-cxn` triggers at the end when all necessary case and agreement information has been provided by constructions belonging to other sets. Moreover, the `unmarked-3sg-agent-trans-cxn` is prevented from applying because it belongs to a set that is processed later. The explicit ordering of sets functions thus as a kind of *block* that prevents the unmarked construction from applying too early.

The idea of blocking constructions comes very close to the notion of the footprints that the construction application process leaves behind. In order to *block* the future application of the same construction at exactly the same spot in the transient structure, constructions add footprints. In this sense, when multiple constructions would share the same footprint, the application of the first construction would block all remaining constructions. The `==0` operator makes this exclusion clear. On the other hand, footprints are also sometimes used to indicate that a particular construction needs to have applied before another one will match the transient structure. This is done with the `==1` operator. By explicitly saying that a footprint needs to be present in the footprint list of a unit, you achieve some kind of precedence link.

However, footprints are only added to the transient structure when a construction could apply and thereby change the structure of previous transient structure. In the case of the different argument structure constructions, this would not be enough as the constructions do not constitute some kind of prerequisites for each other but should be seen in an exclusive relationship. The transitive construction should be *tried before* the intransitive construction but is not a prerequisite for the latter.

6. Conclusions

On the one hand, this paper has introduced an important design pattern for dealing with unmarked forms in FCG. By delaying the actual application of unmarked constructions in parsing, they can fill in the necessary information that has not been provided by the form that is being processed. On the other hand, this delay can

only be achieved by splitting up the inventory of constructions into multiple construction sets. The driving force behind the introduction of construction sets into the FCG-system is two-fold. The importance of efficiency in processing together with effectiveness in grammar design brought the notion of construction sets into life. Their practical workings were illustrated here by a case study that dealt with the non-modular processing of grammatical agreement constraints in Hungarian.

Acknowledgements

This research was conducted at the Vrije Universiteit Brussel, financed by a strategic basic research grant (IWT-489) from the agency for Innovation by Science and Technology (IWT). Additional funding came from the European research project ALEAR (FP7, ICT-214856). Apart from the members of our team in Brussels and at the Sony CSL lab in Paris, I especially want to thank István Zachar and Tünde Marusnik for their constructive comments on previous versions of this paper.

References

Bühler, Karl (1934). *Sprachtheorie: die Darstellungsfunktion der Sprache*. Jena, Fischer.

Croft, William, Allan Cruse (2004). *Cognitive Linguistics*. Cambridge Textbooks in Linguistics. Cambridge University Press.

Gerasymova, Kateryna (2012). Expressing grammatical meaning - a case study for Russian aspect. In Luc Steels (Ed.), *Computational Issues in Fluid Construction Grammar*, Lecture Notes in Artificial Intelligence. Berlin: Springer.

Gerasymova, Kateryna, Luc Steels, Remi van Trijp (2009). Aspectual morphology of Russian verbs in Fluid Construction Grammar. In N.A. Taatgen, H. van Rijn (Eds.), *Proceedings of the 31th Annual Conference of the Cognitive Science Society*, 1370–1375. Cognitive Science Society.

Goldberg, Adele, Laura Suttle (2010). Construction grammar. *Wiley Interdisciplinary Reviews: Cognitive Science*, 1(4), 468–477.

Kiss, Katalin (2002). *The syntax of Hungarian*. Cambridge syntax guides. Cambridge University Press.

MacWhinney, Brian, Csaba Pléh (1997). Double agreement: Role identification in Hungarian. *Language and Cognitive Processes*, 12(1), 67–102.

Siewierska, Anna, Dik Bakker (1996). The distribution of subject and object agreement and word order type. *Studies in Language*, 20(1), 115–161.

Steels, Luc (2011). A design pattern for phrasal constructions. In Luc Steels (Ed.), *Design Patterns in Fluid Construction Grammar*. Amsterdam: John Benjamins.

Steels, Luc, Joachim De Beule, Nicolas Neubauer (2005). Linking in fluid construction grammars. In *Proceedings of BNAIC*, 11–18. Brussels: Transactions of the Belgian Royal Society of Arts and Sciences.

Steels, Luc, Remi van Trijp (2011). How to make Construction Grammars fluid and robust. In Luc Steels (Ed.), *Design Patterns in Fluid Construction Grammar*. Amsterdam: John Benjamins.

Steels, Luc, Pieter Wellens (2006). How grammar emerges to dampen combinatorial search in parsing. In P. Vogt, Y. Sugita, E. Tuci, C. Nehaniv (Eds.), *Symbol Grounding and Beyond: Proceedings of the Third International Workshop on the Emergence and Evolution of Linguistic Commun*, LNAI 4211, 76–88. Berlin: Springer-Verlag.

Törkenczy, Miklós (2005). *Practical Hungarian Grammar*. Corvina Books Ltd., 2nd edn.

van Trijp, Remi (2011). Feature matrices and agreement: A case study for German case. In Luc Steels (Ed.), *Design Patterns in Fluid Construction Grammar*. Amsterdam: John Benjamins.

Wellens, Pieter (2011). Organizing constructions in networks. In Luc Steels (Ed.), *Design Patterns in Fluid Construction Grammar*. Amsterdam: John Benjamins.