Framing Fluid Construction Grammar

Vanessa Micelli (vanessa@csl.sony.fr)
Sony Computer Science Laboratory Paris
6 Rue Amyot, 75005 Paris, France

Remi van Trijp (remi@csl.sony.fr)
Sony Computer Science Laboratory Paris
6 Rue Amyot, 75005 Paris, France

Joachim De Beule (joachim@arti.vub.ac.be)
VUB Artificial Intelligence Laboratory Vrije Universiteit Brussel
Pleinlaan 2, 1050 Brussels, Belgium

Abstract

In this paper, we propose a concrete operationalization which incorporates data from the FrameNet database into Fluid Construction Grammar, currently the only computational implementation of construction grammar that can achieve both production and parsing using the same set of constructions. As a proof of concept, we selected an annotated sentence from the FrameNet database and transcribed its frame annotation analysis into an FCG grammar. The paper illustrates the proposed constructions and discusses the value and results of these formalization efforts.

Keywords: Fluid Construction Grammar; FrameNet; Frame Semantics; Grammar Formalism

Introduction

Construction Grammar (CG) and Frame Semantics (FS) are considered to be sister theories in cognitive linguistics. FS investigates the frames of semantic knowledge that a language user needs in order to produce and comprehend words successfully, whereas CG tries to describe the constructions (i.e. meaning-form mappings) of a language. However, even though many construction grammarians subscribe their work to FS, most analyses only focus on the "skeletal meanings" (Goldberg, 1995, p. 28) that underlie grammatical constructions, and leave the exact integration of FS and CG underspecified. This gap may cause inconsistencies in the development of both theories and leaves a lot of crucial issues unaddressed within cognitive linguistics.

In this paper, we therefore propose a concrete operationalization that overcomes this problem by using data from the FrameNet project (Baker, Fillmore, & Lowe, 1998) and Fluid Construction Grammar (FCG) (De Beule & Steels, 2005; Steels & De Beule, 2006), a computational formalism that allows researchers to test their hypotheses for both production and parsing. More specifically, we focus on a hand-made example[1] that serves as a proof-of-concept for our approach, and we discuss possible research avenues for the future such as the automatic incorporation of FrameNet data into FCG. We believe that a computational formalization is a crucial aspect of empirical science because it makes the sometimes fuzzy aspects of linguistic theories explicit and because it reveals consequences of a theory that would have otherwise been overlooked.

This paper is structured as follows: in the next section, we briefly introduce the FrameNet project and Fluid Construction Grammar and explain our motivations for coupling the two with each other. We then discuss the example sentence that we implemented and show what we had to add to its annotation to arrive at an operational result. We then look at some of the lexical and grammatical constructions that we implemented and how they are processed by FCG. Finally, we discuss the obtained results and insights and we briefly touch upon future efforts.

[1]Given the space limitations of this paper and the elaborate annotation of the example sentence, it is impossible to show a complete trace of production or parsing. Interested readers can therefore check the complete and interactive demonstration of the example at `www.fcg-net.org/framenet/`.

Tying the knot between FrameNet and FCG

Since Construction Grammar and Frame Semantics are both based on the same theoretical foundations, it is only natural to investigate how their existing implementations could possibly profit from each other and what would be the best and most advantageous way of doing so. In this section, we briefly introduce FrameNet and FCG, and we motivate why the combination of both forms a powerful tool for investigating both the semantic and grammatical properties of constructions.

FrameNet

The FrameNet database (Baker et al., 1998) presents a huge online database containing more than 10.000 English lexical units (LUs) and, more recently, similar efforts for several other languages such as German and Japanese. All lexical units are annotated with their semantic frames based on example sentences in which the respective frames and frame elements are marked. The corpus-oriented approach of the FrameNet project makes that it is tightly connected to empirical observations. Unfortunately, no computational implementation exists of how these annotated frames can be processed in either production or parsing.

Fluid Construction Grammar (FCG)

Fluid Construction Grammar (De Beule & Steels, 2005; Steels & De Beule, 2006) is a fully operational grammar formalism that has been explicitly designed for capturing the emergent and living aspects of language (Steels, 2000, 2004, 2005). This focus on evolutionary linguistics sets it apart from other grammar formalisms such as Head-Driven Phrase Structure Grammar and Lexical Functional Grammar, and is visible in the use of techniques such as explicit directionality, entrenchment scores and a meta-layer of learning operators. Its linguistic perspective is compatible with cognitive linguistics and construction grammar (Goldberg, 1995; Langacker, 2000) and like many other contemporary theories it is feature structure- and unification-based. FCG's explicit directionality tightly couples linguistic competence to performance, as opposed to non-directional declarative formalisms that are less concerned with performance issues. Moreover, FCG is capable of processing the same constructions in both production and parsing, whereas non-directional grammars typically need to compile a grammar into separate generator and parsing procedures. So far, FCG has mainly been applied in research on the emergence and evolution of grammatical phenomena (Steels, 2004; van Trijp, 2008), but unfortunately, there is no large database yet of lexical and grammatical constructions for natural language processing.

Motivation (and gains)

From the above two subsections, it is clear that FrameNet is in need of a computational formalism for testing its frame annotations in actual language processing, and that FCG could use a large linguistic inventory for broadening its scope of application. The two, therefore, seem to be a perfect fit, especially given the fact that FCG uses feature structures for representing linguistic knowledge, which is highly compatible with how semantics is encoded in the FrameNet database. Our aim in this paper is therefore to show how the combination of both offers linguists a very valuable tool for exploring theoretical issues and challenges in applied linguistics that require text understanding or production facilities.

This paper thus offers a proof-of-concept of our approach through an example taken from the FrameNet database. In the next sections, we will discuss this example and show how we implemented it in FCG. This formalization effort makes clear which aspects were still missing in the annotation for making the sentence operational and indicates which steps need to be taken for an automated integration of FrameNet and FCG.

The river forms a natural line

As a case study, we picked the following sentence:

- *The river forms a natural line between the north and south sections of the city.*[2]

In the remainder of this paper, we solely consider those frames and their respective frame elements that were used in the online annotation of the example sentence rather than exhaustively listing every frame contributing to the full semantics of the sentence. In this section, we first give an overview of the annotation as provided by the FrameNet project. Next, we discuss the changes and additions that were necessary to arrive at an operational implementation of the example.

FrameNet's annotation

Figure 1 displays which frames are evoked by the lexical units. Frames are represented as boxes attached to the LUs that evoke them in capital letters. For example, the whole sentence is governed by the CREATING frame, which is evoked by *forms*. This frame contains two frame elements (presented by the boxes that are connected to their respective frame through dotted lines and in which in general only the first letter is capitalized): a Cause (*the river*) and a Created_Entity (*a natural line between the north and south sections of the city*).

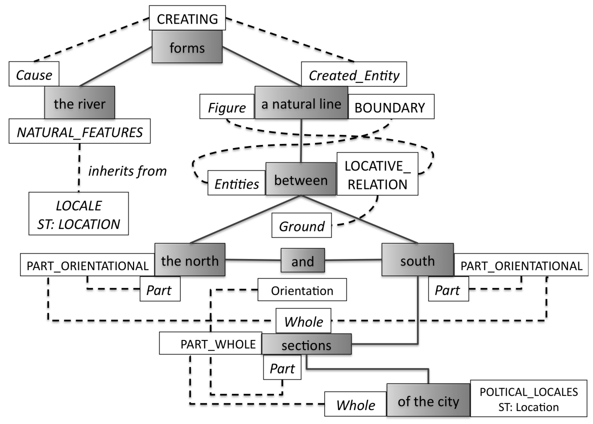

Figure 1: Annotation of the example sentence with the frames that contribute to its meaning. Frames (capital letters) and their frame elements (small letters) are connected through dotted lines.

Additions and changes to the annotation

While modeling the meaning poles of the constructions, we tried to stay as close as possible to the annotations as suggested by the FrameNet project's annotators. However, every attempt at operationalizing an analysis has the great advantage that it will always reveal missing bits or aspects that need to be modified. This is also the case for the current example. For instance, implementing the example into FCG required an explicit representation of determiners, prepositions and of all adjectives (which were partly neglected in the annotation). So our modifications include also an operationalization of the words *the, a, natural, and,* and *of*. Additionally, we left out the PART_ORIENTATIONAL frame because the values of its frame elements Part and Whole were already provided by the PART_WHOLE frame. Secondly, the frames that were provided by the FrameNet annotations only represent the 'super-

[2]This sentence and its annotation can be found at `http://framenet.icsi.berkeley.edu/index.php?option=com_wrapper&Itemid=84` in Section 6 of the annotated text called *Intro of Dublin*.

ficial' meaning of the sentence. In order to arrive at more general and abstract constructions, however, we also needed to include some of the higher level frames from which a particular frame inherits. For example, the word *river* not only evokes the frame NATURAL_FEATURES, but also the frame ENTITY. This is necessary because grammatical constructions abstract away from the fine-grained details of specific frames and only select on the more general frames.

Formalizing the example

In this section, we go deeper into the FCG constructions that formalize the FrameNet example. The main point of this section is not to offer the best or most general description of any particular construction, but rather how this particular example can fit into the FCG framework. We provide concrete constructions for completeness' sake, but it falls outside the scope of this paper to explain all technical details of the formalism[3].

Lexical entries

Lexical entries are form-meaning mappings for particular words. In production, a lexical entry is triggered by a certain meaning. For example, the lexical entry for *river* looks whether among the meanings that the speaker wishes to express (stored in the so-called top-unit), there is a meaning *(river ?x)* in which the logic variable *?x* (indicated by a question mark) can be bound to the particular river the speaker wants to talk about. The construction 'tags' this meaning, and – if the meaning is found among the meanings that need to be expressed – removes it from the top-unit and encapsulates it into a newly created unit (using the J-operator, see De Beule & Steels, 2005). Additionally, the construction includes the semantic frames of this unit: NATURAL_FEATURES and ENTITY. If the unification of the semantic pole was successful, the syntactic pole is merged with the linguistic structure that is being built by the speaker:

```
(def-cxn river
 ((?top
   (TAG ?meaning
        (meaning (== (river ?x))))
   (sem-cat (==0 (frames (== natural-features)))))
  ((J ?river-unit ?top)
   (referent ?x)
   (sem-cat
    ((frames (natural-features entity))
     (semantic-type (location))))
   ?meaning))
 <-->
 ((?top
   (TAG ?form
        (form (== (string ?river-unit "river"))))
   (syn-cat (==0 (pos (== noun)))))
  ((J ?river-unit ?top)
   ?form
   (syn-cat
    ((pos (noun common-noun))
    (type (nominal)))))))
```

[3]The reader is not expected to understand every part of the presented code. For more technical details, please see the mentioned FCG-related documents.

In parsing, FCG can use the same lexical entry but this time applies it in the opposite direction. First the syntactic pole is unified with the linguistic structure that the hearer wants to parse. This lexical entry looks for the word *river* in the observed utterance, and if found, creates again a separate unit for it using the J-operator, adding syntactic information such as the word's part of speech (noun). If the unification of the syntactic pole is successful, the information of the semantic pole is merged with the linguistic structure as well. All other constructions in our formalisation, including grammatical ones, work in exactly the same way.

Through the application of constructions, the speaker (in production) or hearer (in parsing) builds up an increasingly complex linguistic structure until an utterance is licensed or until sufficient meanings have been parsed for interpreting the utterance, as illustrated in the following Figure 2.

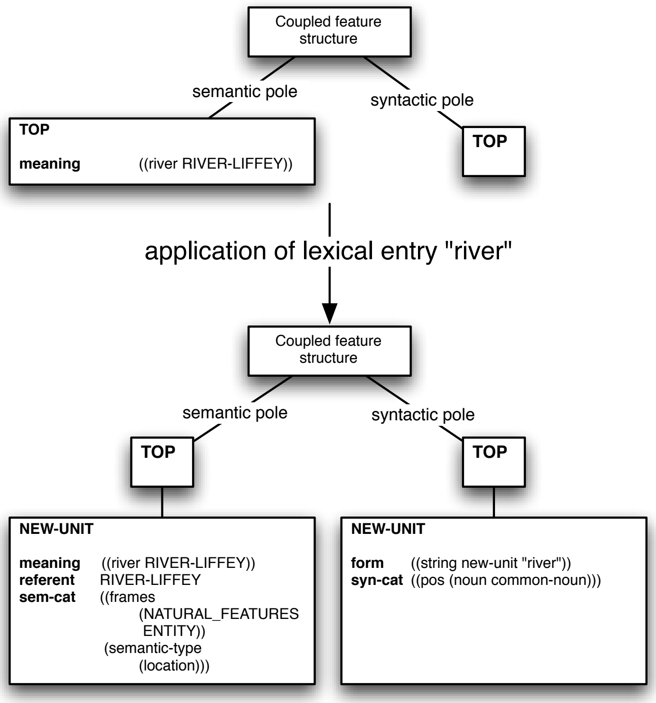
Coupled feature structure
semantic pole
syntactic pole
TOP
meaning ((river RIVER-LIFFEY))
TOP
application of lexical entry "river"
Coupled feature structure
semantic pole
syntactic pole
TOP
TOP
NEW-UNIT
meaning ((river RIVER-LIFFEY))
referent RIVER-LIFFEY
sem-cat ((frames (NATURAL_FEATURES ENTITY)) (semantic-type (location)))
NEW-UNIT
form ((string new-unit "river"))
syn-cat ((pos (noun common-noun)))

Figure 2: This Figure shows how the linguistic structure (the coupled feature structure) can be expanded and modified by applying a construction. In this production example, the lexical entry "river" is triggered by the presence of the meaning (river RIVER-LIFFEY) in the top-unit. The construction then creates a separate unit for this word and adds semantic and syntactic information, such as its frames, form and part-of-speech (pos).

Grammatical constructions

As an example of a grammatical construction, we have chosen the argument structure construction that is triggered by the CREATING frame, which governs the entire sentence. As with the lexical constructions, the creating-construction is triggered by the presence of a certain meaning in the top-unit of the semantic pole of the linguistic structure. In line with most construction grammar approaches, the argument structure construction looks for a 'skeletal meaning' rather than a rich lexical meaning:

(creating ?crt)
(causer ?crt ?agent)
(created-entity ?crt ?theme)

This meaning states that the construction requires (or adds through coercion in parsing) a creating-event (*forms*), a causer (*the river*) and the created entity (the rest of the utterance).

As can be seen below, the creating-construction is more complex than a lexical construction because it not only has to unify with the top-unit of the linguistic structure, but also with some of the units that were created before by lexical or other grammatical constructions. Here, the creating-construction specifies that (for production) it needs two units which contain the frame REFERENT and one unit which contains the frame CREATING. The REFERENT frame is evoked by things like proper names and determined noun phrases (which make REFERENTS from ENTITIES). If all constraints are satisfied and unification is successful, the construction will again create a new unit (using the J-operator) and move the selected meaning from the top-unit to this new unit. Next, the construction also specifies that the three units that either evoked the CREATING frame or that carry the relevant frame elements are subunits of this newly created unit. Finally, the syntactic pole is merged with the linguistic structure, which (among other things) specifies word order constraints among the units.

```
(def-cxn creating-cxn
 ((?top
   (sem-subunits
    (== ?causer-unit ?event-unit ?theme-unit))
   (sem-cat (==0 (frames (== creating))))
   (TAG ?meaning
        (meaning (== (creating ?crt)
                     (causer ?crt ?agent)
                     (created-entity ?crt ?theme)))))
  (?causer-unit
   (referent ?agent)
   (sem-cat (==1 (frames (== referent)))))
  (?event-unit
   (referent ?crt)
   (sem-cat (==1 (frames (== creating)))))
  (?theme-unit
   (referent ?theme)
   (sem-cat (==1 (frames (== referent)))))
  ((J ?creation-unit ?top
     (?causer-unit ?event-unit ?theme-unit))
   (referent ?t)
   ?meaning
   (sem-cat (==1 (frames (== creating))))))
 <-->
 ((?top
   (syn-subunits
    (== ?causer-unit ?event-unit ?theme-unit))
   (syn-cat (==0 (phrase (== declarative))))
   (TAG ?form
        (form (== (meets ?causer-unit ?event-unit)
                  (meets ?event-unit ?theme-unit)))))
  (?causer-unit
   (syn-cat (==1 (type (== nominal))
                 (phrase (== det-noun-phrase)))))
  (?event-unit
   (TAG ?pos (syn-cat (==1 (pos (verb))))))
  (?theme-unit
   (syn-cat (==1 (type (== nominal))
    (phrase (== det-noun-phrase)))))
  ((J ?creation-unit ?top
     (?causer-unit ?event-unit ?theme-unit))
   ?form
   (syn-cat (==1 (phrase (== declarative)))))))
```

Note that the construction uses logic variables for disambiguating which unit is the Causer and which unit is the Created-Entity, rather than merely selecting on units that contain the frame REFERENT (see Steels, De Beule, & Neubauer, 2005, for a more elaborate explanation of the role of logic variables in FCG). For example, the logic variable *?agent*, which is bound to the river (Liffey in Dublin) in our example, occurs both in the meaning of the creating-event and in the referent feature of one of the subunits (the ?causer-unit). Repeating the variable in both places represents the fact that the referent of the ?causer-unit needs to be bound to the same object in reality as the causer of the creating-construction. Likewise, the repetition of the variable *?theme* in two places indicates that the filler of the frame element Created_Entity needs to be bound to the same referent as the one of the ?theme-unit.

The application of the creating-construction can perhaps be better illustrated using a visual representation. Because of space limitations, we will use a partial tree for this. Before the construction is triggered, other constructions have already introduced several lexical and grammatical units into the linguistic structure, which can be roughly represented as follows:

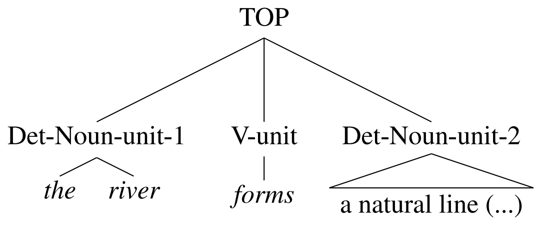

Figure 3: A schematic representation of the linguistic structure before applying the creating-construction.

As explained before, the creating-construction will create a new subunit that hangs directly from the top, and which takes the three current subunits as its own subunits. Through the use of co-occurrences of the logic variables, the construction can also assign the correct frame element to each unit. The resulting structure is shown in Figure 4.

Formalizing the additions

As already mentioned earlier, we had to make several additions to the FrameNet annotation such as including an ex-

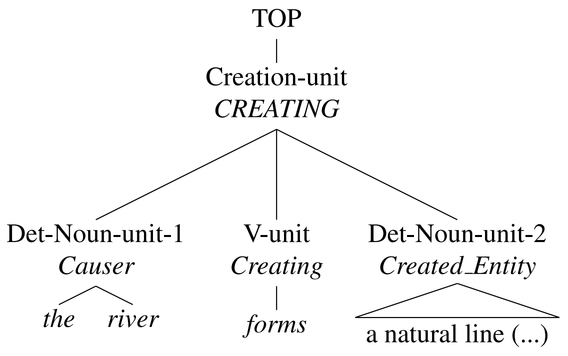

Figure 4: A schematic representation of the linguistic structure after applying the creating-construction.

plicit representation of determiners, prepositions and adjectives. Here, we will briefly discuss their formalization and the consequences of this approach for the grammatical constructions.

Our first example concerns the lexical entry for *natural*. This construction looks very much like a lexical entry for a noun, except that its meaning is a bit different. Instead of a meaning such as *(line ?entity)*, in which *?entity* is bound to the referent of the word *line*, the adjective 'returns' the modified referent of the noun. For example, in the meaning *(natural ?x ?entity)*, the variable *?x* will be bound to the referent that remains after modifying the referent of *?entity* according to the category 'natural'. The construction for *natural* looks as follows:

```
(def-cxn natural
 ((?top
   (TAG ?meaning
             (meaning (== (natural ?x ?entity))))
   (sem-cat (==0 (frames (== natural-frame)))))
  ((J ?modify-unit ?top)
   (referent ?x)
   ?meaning
   (sem-cat
    ((frames
      (natural-frame (modifying ?x ?entity)))))))
 <-->
 ((?top
   (TAG
    ?form (form (== (string ?modify-unit "natural"))))
   (syn-cat (==0 (pos (== adjective)))))
  ((J ?modify-unit ?top)
   ?form
   (syn-cat ((pos (adjective))
             (type (adjectival)))))))
```

Similar to *natural*, the words *north* and *south* also modify another entity: the combination *the north sections (of the city)* would therefore be bound to a different referent than just *the sections (of the city)*. The lexical construction for *and*, on the other hand, does not modify the referent of a noun but rather combines two referents with each other. In this example, the referent of *the north and south sections (of the city)* is therefore the union of the referent of *the north sections* and the referent of *the south sections* (of the city).

Next to modifiers, there are also the prepositions *between* and *of* which define relations between words. For example, the preposition *of* makes the part-whole relation between *the north and south sections* (Part) and *the city* (Whole) explicit. The preposition *between* evokes a LOCATIVE_RELATION between the two *sections of the city*.

The more complex and relational meanings of these words have some important consequences for their semantic frames. As can be seen in the semantic pole of the lexical entry for *natural*, the construction not only evokes the word-specific NATURAL-FRAME, but also the more abstract frame (MODIFYING ?X ?ENTITY), in which the variables *?x* and *?entity* can be considered to be frame elements much like the CREATING frame consists of the frame elements Causer and Created_Entity. This in turn is important for grammatical constructions such as the Adjective-Noun construction, which select on the presence of these frame elements in production.

In the following section we present the Adjective-Noun construction, which looks for any unit with the frame (MODIFYING ?x ?entity) (with *?entity* being bound to the referent of the entity that needs to be modified) and another unit with the frame ENTITY. In order to know which modifier goes with which entity, the variable *?entity* must be repeated in the referent feature of the modified noun. In parsing, the construction triggers on the occurrence of an adjective right in front of a noun. Important to notice is that the Adjective-Noun construction creates a new unit which has the frame ENTITY and the modified referent on the semantic pole, and the syntactic type 'nominal' in the syntactic pole. This allows for a recursive application of the construction in case of multiple adjectives.

```
(def-cxn Adjective-Noun-cxn
 ((?top
   (sem-subunits (== ?modify-unit ?entity-unit))
   (sem-cat (==0 (frames (== entity)))))
  (?modify-unit
   (referent ?x)
   (sem-cat (==1 (frames (== (modifying ?x ?entity))))))
  (?entity-unit
   (referent ?entity))
  ((J ?adj-noun-unit ?top
     (?modify-unit ?entity-unit))
   (referent ?x)
   (sem-cat (==1 (frames (== entity))))))
 <-->
 ((?top
   (syn-subunits (== ?modify-unit ?entity-unit))
   (syn-cat (==0 (phrase (== adj-noun-phrase))))
   (TAG ?form
       (form (== (meets ?modify-unit ?entity-unit)))))
  (?modify-unit (syn-cat (==1 (type (== adjectival)))))
  (?entity-unit (syn-cat (==1 (type (== nominal)))))
  ((J ?adj-noun-unit ?top (?modify-unit ?entity-unit))
   ?form
   (syn-cat (== (phrase (== phrase adj-noun-phrase))
                (type (nominal)))))))
```

Overview

In total there are 13 lexical constructions and 6 grammatical constructions (Determiner-Noun, Adjective-Noun, Adding-Parts, Part-Whole, Locative and the Creating construction). When applied together, they yield the linguistic structure that was shown in Figure 1. The complete and interactive demonstration of the example (both production and parsing) can be tested at `www.fcg-net.org/framenet/`.

Future work

Recently, we have started to automatize the transcription process from FrameNet frames to FCG rules. So far, we were able to achieve this for most lexical units included in the FrameNet database, resulting in an FCG grammar containing approximately 10.000 (lexical) constructions. Further work includes investigating whether this grammar can be used as a seed in a bootstrapping process to learn progressively more abstract constructions like, for example, argument structure constructions.

Discussion and conclusions

In this paper, we offered a proof-of-concept example that shows how annotations from the FrameNet database can be coupled to the computational grammar formalism Fluid Construction Grammar. We argued that this coupling not only operationalizes the integration of Frame Semantics and Construction Grammar, but that it also allows for a more rigorous investigation of the hypotheses put forward by cognitive linguistics. Additionally, our analysis shows the importance of integrating also the semantics contributed by all lexical units as, for instance in our example, adjectives or prepositions which are more or less neglected in the FrameNet annotations.

We strongly believe that the formalization of Frame Semantics and Construction Grammar in concrete models, in concord with empirical data from various subfields of cognitive linguistics, will open up interesting research avenues in the future and allow for a deeper understanding of linguistic processing and human cognition.

Acknowledgments

The research presented in this paper was funded by the European project ALEAR, the Sony Computer Science Laboratory Paris, and the VUB AI Lab at the Vrije Universiteit Brussel. The authors wish to thank all team members for their insightful comments, and in particular Luc Steels, director of both laboratories.

References

Baker, C. F., Fillmore, C. J., & Lowe, J. B. (1998). The berkeley framenet project. In *Proceedings of the 17th international conference on computational linguistics.* Morristown, NJ, USA: Association for Computational Linguistics.

De Beule, J., & Steels, L. (2005). Hierarchy in Fluid Construction Grammar. In U. Furbach (Ed.), *Ki 2005: Advances in artificial intelligence. proceedings of the 28th german conference on ai* (Vol. 3698). Koblenz: Springer.

Goldberg, A. E. (1995). *Constructions: A Construction Grammar approach to argument structure*. Chicago, USA: University of Chicago Press.

Langacker, R. (2000). *Grammar and conceptualization*. Den Haag: Mouton de Gruyter.

Steels, L. (2000). Language as a complex adaptive system. In M. Schoenauer (Ed.), *Proceedings of ppsn vi: Lectur notes in computer science* (pp. 17–26). Berlin: Springer-Verlag.

Steels, L. (2004). Constructivist development of grounded construction grammars. In W. Daelemans (Ed.), *Proceedings annual meeting of association for computational linguistics.* Barcelona: ACL.

Steels, L. (2005). The emergence and evolution of linguistic structure: From lexical to grammatical communication systems. *Connection Science*, *17*(3-4), 213–230.

Steels, L., & De Beule, J. (2006). Unify and merge in Fluid Construction Grammar. In P. Vogt, Y. Sugita, E. Tuci, & C. Nehaniv (Eds.), *Symbol grounding and beyond.* (pp. 197–223). Berlin: Springer.

Steels, L., De Beule, J., & Neubauer, N. (2005). Linking in Fluid Construction Grammars. In *Proceedings of the 17th belgium-netherlands conference on artificial intelligence (bnaic '05). transactions of the belgian royal society of arts and sciences* (pp. 11–20). Brussels: Belgian Royal Society of Arts and Sciences.

van Trijp, R. (2008). The emergence of semantic roles in Fluid Construction Grammar. In A. D. Smith, K. Smith, & R. Ferrer i Cancho (Eds.), *The evolution of language. proceedings of the 7th international conference (evolang 7)* (pp. 346–353). Singapore: World Scientific Press.