Introducing Fluid Construction Grammar

Luc Steels

Abstract

Fluid Construction Grammar (FCG) is a formalism for defining the inventory of lexical and grammatical conventions that language processing requires and the operations with which this inventory is used to parse and produce sentences. This chapter introduces some of the key ideas and basic design principles behind the development of Fluid Construction Grammar.

1. Background

Fluid Construction Grammar (FCG) is designed primarily to allow computational linguists to formally write down the inventory of lexical and grammatical constructions needed in parsing or producing utterances or do experiments in language learning and language evolution. A computational formalism is necessarily based on a particular perspective on language. For FCG, this perspective is inspired by research into cognitive linguistics in general and construction grammar in particular. FCG is not intended to displace existing linguistic proposals for construction grammar, such as those made by (Goldberg, 1995, 2006; Croft, 2001; Kay & Fillmore, 1999; Michaelis & Lambrecht, 1996), a.o. These proposals usually stay at the level of non-formal verbal descriptions and do not take into account processing issues. On the other hand, FCG does not want to commit to specific opinions about how certain grammatical phenomena need to be handled. Instead, it wants to be an open instrument that can be used by construction grammarians who want to formulate their intuitions and data in a precise way and who want to test the implications of their grammar designs for language parsing, production and learning. Throughout this book the term parsing is used for the process of mapping form to meaning and production for the process of mapping meaning to form. Production is not the same as generation (as in generative grammar). Generation is not constrained by semantics.

FCG does not make any claims about psychological validity. The emphasis is on getting working systems, and this is difficult enough. Only when we have been able to come up with possible effective models of language processing does it make sense to inquire which ones of these models might be psychologically or neurologically most plausible.

Language is constantly changing, shows a lot of variation and exhibits a high degree of flexibility in the way it is used in actual discourse. Human natural languages are therefore significantly different from programming languages, logical calculi or mathematical formalisms, because they are open systems. Language users break linguistic conventions as fast as they invent them in order to increase the expressive power of their language for the purpose of dealing with new meanings or for catching the attention of the listener with novel phrases. Undeniably, there is huge variation in language, even among those speaking the same language, and even in the language use of a single individual as he or she is switching between different contexts and interacting between members of different social groups. Formal and computational approaches to language should not ignore these facts, but instead take the challenge of dealing with the 'fluidic' nature of language as one of its objectives. This is what Fluid Construction Grammar tries to do.

Current approaches to language processing try very hard to split entirely issues of efficiency from issues of grammar representation (Sag et al., 2003). There is much to say for this point of view. But FCG does not entirely follow this line of argument. Often the representation of grammar has a profound impact on how efficient language processing with a grammar can be, and so FCG provides a variety of mechanisms that give more procedural control over language processing.

Fluid Construction Grammar has been fully implemented in a system called the *FCG-system* which is made available for free to the research community (http://www.fcg-net.org/). The FCG-system contains a core component (called the *FCG-interpreter*) that performs basic operations needed for parsing and production, as well as various tools to aid linguistic research, such as a tool for browsing through linguistic structures (the *FCG-browser*) and a tool for monitoring the success rate of a grammar when processing a set of test cases (the *FCG-monitor*). The FCG-system should not be seen as a finished product. It has been under development from around 1998 in order to support experiments in modeling language evolution using language games played by autonomous robots (Steels, 1998), and since then it has undergone major revisions and enhancements. The FCG-system is still continuously being adapted and revised today to cope with new linguistic phenomena and new processing challenges, and to improve the ease with which complex lexicons and grammars can be developed and tested. Nevertheless, the system can already be used to tackle sophisticated issues in the representation and processing of language as other contributions to this book abundantly show.

This chapter discusses some of the key concepts behind the development of Fluid Construction Grammar: What are constructions? What does language processing using constructions looks like? Why would we want to use a construction-based organization of linguistic competence? And how does FCG attempt to deal with some of the key problems of language processing such as combating combinatorial explosions in the search space or dealing with the fluidity of language?

2. What are Constructions?

The notion of a construction has been at the core of linguistic theorizing for centuries (Östman & Fried, 2004). A *construction* is a regular pattern of usage in a language, such as a word, a combination of words, an idiom, or a syntactic pattern, which has a conventionalized meaning and function (Goldberg & Suttle, 2010). The term construction is used from now on both to refer to the pattern itself and to the knowledge that a speaker or hearer needs to handle the usage pattern in producing or comprehending utterances. The meaning and functional side of a construction, as well as relevant pragmatic aspects, are captured in a *semantic pole*, and all aspects which relate to form, including syntax, morphology, phonology and phonetics are captured in a *syntactic pole*.

Here are some examples of constructions:

1. Single words, or more precisely lexical stems, like "walk", are covered by *lexical constructions*. They capture a direct association between a string, with a particular stress pattern, (the syntactic pole) and its meaning (the semantic pole). Lexical constructions also introduce additional syntactic and semantic categorizations that are important for later grammatical processing, such as the lexical category (part of speech), number or gender.

2. A *determiner-nominal construction* combines a determiner, such as an article like "the" or "some", with a nominal, such as "table" or "white book", to form a referring expression. The semantic pole of this construction specifies that the nominal introduces a class of objects (e.g. the set of tables) and the

determiner specifies how we get the referent of this set (*a* table, *the* table, *two* tables, etc.). The syntactic pole prescribes a particular ordering of the constituents: the determiner must come first, and the nominal (which possibly consists of a combination of adjectivals and a nominal) must directly follow it. Depending on the language, it also prescribes agreement for number, gender, case or other features between the determiner and the nominal.

3. A *passive construction* prescribes a particular ordering of the constituents in the sentence (Subject Verb Direct Object by+Object), such as “This book was given to me by a famous linguist.” There are two verbal components, the auxiliary “be” and the past participle. The constituent that would normally be the subject in the active counterpart of the sentence is introduced as a prepositional phrase with the preposition “by”, here acting as a grammatical function word. The semantic pole of this construction specifies not only what roles participants play in the event introduced by the main verb, but it also highlights one participant by making it the subject.

4. A *resultative construction* has a particular syntactic pattern of the form: Subject Verb Direct-Object Predicate. It is illustrated with sentences like “Fred watered the plants flat” or “The critics laughed the play off the stage” (Goldberg & Jackendoff, 2004). The semantic pole of this construction specifies that the predicate (for example “flat”) describes the state of the referent of the direct-object (“the plants”) as a side effect of the action described in the main verb (“watered”).

5. A *postposed-genitive construction* (such as “This book of mine”) combines a nominal phrase (”this book”) with a preposition (”of”) and a genitive (”mine”). The semantic pole adds meaning, namely that there is a possessive relation between the referent of the nominal phrase (”this book”) and the referent of the genitive (”mine”). (Lyons, 1985).

Constructions clearly form a continuum between quite abstract grammatical constructions, such as the determiner-nominal construction and so called *item-based constructions*, which are built out of lexical materials and frozen syntactic patterns. They contain open slots in which structures with specific semantic and syntactic properties can fit, as in the “let-alone” construction, underlying a sentence like “Joan is unable to write 5 pages, let alone a whole book”. (Fillmore, et.al. 1988).

Constructions relate meaning to form through the intermediary of semantic and syntactic categorizations (Figure 1). Semantic categorizations are ways in which meaning and function are conceptualized or re-conceptualized for language. For example, in many languages, the specific roles of the participants in an event introduced by a verb are categorized in terms of abstract semantic categorizations (like agent, patient, beneficiary, possessor, location) before they are mapped into abstract syntactic categorizations (like the syntactic cases nominative, dative, accusative, or genitive), which then translate further into surface forms. Using such abstract categorizations is obviously much more efficient than having an idiosyncratic way to express the participant roles of each verb, because fewer constructions are needed, and novel sentences can be partially understood, even if the meaning of the verb is unknown.

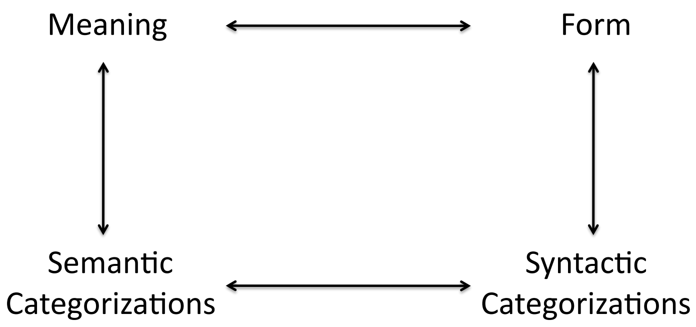

Figure 1. *The grammar square depicts the different associations between meaning and form that constructions establish. Meaning can be directly related to form, as in the case of words, or it is expressed through the intermediary of semantic and syntactic categorizations.*

Constructions typically establish various relations at the same time. For example, lexical constructions associate some meaning directly with a word stem but they also already specify some of the syntactic and semantic categorizations that are associated with the word and its meaning. Some constructions are entirely dedicated to inferring more syntactic and semantic categorizations from those already there. For example, phrasal constructions group units together and determine syntactic and semantic functions of the components. Other constructions focus on establishing mappings between semantic and syntactic categorizations. For example, argument structure constructions map semantic roles like agent, patient or beneficiary to syntactic roles like subject, direct object and indirect object, or to cases like nominative, accusative and dative.

There is a continuum between semantic and syntactic categorizations, because many syntactic categories (such as gender) originally go back to semantic distinctions or functions that have progressively become purely syntactic so that they need to be learned by heart. It is difficult to estimate again how many categorizations natural languages employ, but it is safe to assume that they run into the thousands, particularly if we take all the semantic categorizations into account that play a role in determining syntactic constraints. For example, even for deciding something as simple as the ordering of two color adjectives in a double adjective construction in English (where the first adjective is used adverbially to modify the second one, as in "light green", "bright yellow", or "blue green"), one must take into account a semantic categorization of the second adjective, namely, that it has to express a chromatic color category (i.e. a hue category) as opposed to a non-chromatic one (i.e. expressing brightness (shiny, bright, dull) or lightness (dark, light) dimension). It is incorrect to say "blue light" in order to express a shade of blue that is light, rather "light blue".

It is not easy to estimate the number of constructions in a language because it depends on the 'grain size' of analysis. As a rough estimate, a normal adult speaker probably knows at least 100,000 constructions. 10,000 or more of these are 'abstract' grammatical constructions whereas most of them are lexical. Some researchers argue that the number is much higher because language users store rich, ready-made solutions, even if they can be derived from more abstract constructions. This approach, hence, leans towards *memory-based* or exemplar-based approaches to language processing (Daelemans & Van den Bosch, 2005), which contrasts with the more abstract grammars often proposed in the generative literature. Storing ready-made solutions makes processing faster and explains why so many idiosyncracies exist in language, including idiomatic expressions. If only abstract constructions are stored, they would always take priority over specifics and idiomatic expressions would disappear.

Traditionally, the linguist's task is to describe the constructions found in a language, which means figuring out the form constraints imposed by the syntactic pole and the often subtle meaning and functional distinctions introduced by the semantic pole. In this sense, a linguist is like a naturalist who goes out into nature to find and describe butterflies, plants or other kinds of species, and tries to systematize them. Given that there are so many constructions in a language, this undertaking is incredibly non-trivial. Constructions come in families with slight variations in syntax and semantics, and it is useful to study and systematize these relations as well (Jackendoff & Goldberg, 2004). Today linguists use large data bases of text materials with automatic extraction techniques (Stefanowitsch & Gries, 2003), and they code constructions into computational representations such as FrameNet (Baker et al., 1998) or WordNet (Fellbaum, 1998). The general aim of collecting and describing constructions, however, is essentially the same as it has been for centuries. What has been learned from all these descriptive endeavors?

1. A first insight from empirical research into grammatical constructions is that language is fundamentally *non-modular*, in the sense that a construction can involve any aspect of language and package it together in one usage pattern. Traditionally linguistics makes a distinction between different levels of language analysis:

1. *Pragmatics* focuses on the integration of a sentence in the discourse context, such as how a sentence can make use of information conveyed earlier or in which way certain aspects can be foregrounded.

2. *Semantics* concerns issues related to the meaning of linguistic units. It studies what kind of representational formalism is necessary for defining meaning and how meaning can be interpreted.

3. *Syntax* is concerned with the purely structural constraints on a language, such as what kind of hierarchical sentence pattern may occur in a declarative main sentence.

4. *Morphology* studies the decomposition of individual words in stems and affixes and what determines and limits their combination.

5. *Phonology* focuses on speech but at an abstract level, using features of speech sounds like rounded, voiced, ends-in-consonant-cluster, etc. Languages exhibit many conventionalized regularities on how speech sounds are combined and how they are allowed to influence each other.

6. *Phonetics* focuses on the actual acoustic signature of utterances and the articulatory controls that can produce human speech sounds.

Nowadays, you find linguists specializing in each of these fields, but a real language does of course not care much about divisions in academic disciplines. A construction may cut across all these levels. (See Figure 2.) For example, the suffix added to a Hungarian verb expresses features such as number and gender which are both derived from the subject *and* the direct object (so called poly-personal agreement, see Beuls (2011). The concrete choice of suffix is based on: (i) syntactic considerations, since poly-personal agreement happens only when the direct object is in the

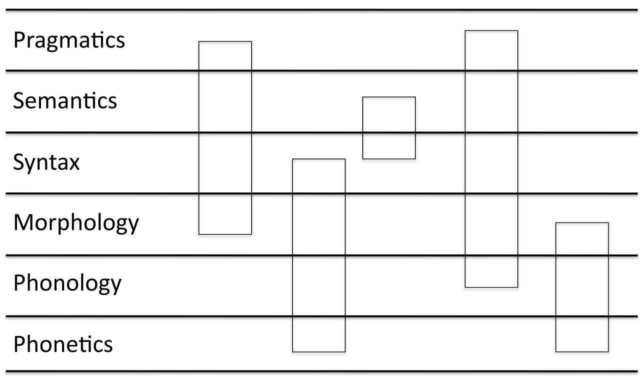

Figure 2. *Constructions package constraints that potentially concern any level of linguistic description, from pragmatics to phonetics. They therefore cut in a vertical way through the traditional levels of linguistic analysis.*

accusative case, (ii) semantic considerations, since the referent of the direct object has to be definite and considered to be further removed from the deictic center than the subject, (iii) morphological considerations, since the morphological structure of the verb determines the choice of suffix, and (iv) phonological considerations since there is vowel harmony between the main vowel in the verb stem and that of the vowel in the suffix. It follows that constructions should be able to have access to whatever level of analysis they need in order to define as clearly as possible all the constraints relevant for a particular step in linguistic decision-making.

Constructions not only integrate aspects downwards from syntax (morphology or phonology), they clearly contribute additional meaning and function which cannot simply be derived from lexical items. Compare for example the following two sentences (Fillmore, 1968):

(1) Bees are swarming in the garden.

(2) The garden is swarming with bees.

These sentences use the same lexical materials for the main constituents, but because they are combined in a different constructional pattern, we get subtle differences in meaning: (1) suggests that there are bees in the garden but possibly only in part of it (or simply fewer bees), whereas (2) suggests that the whole garden is full of bees. The additional meaning is here provided by the grammatical construction and added to the meanings coming from the lexical items. A wealth of other examples how constructions cut vertically across levels are easy to find, in fact it is almost impossible to find a usage pattern in a human language that can be defined purely based on syntactic criteria.

3. A second important insight coming from centuries of empirical research into constructions is that although there are certain trends among the constructions found in languages, particularly those that have common ancestors (such as French and Catalan which both derive from Latin), there are at the same time deep differences, even within regional dialects of the same language or among speakers of the same language (Evans & Levinson, 2009). This phenomenon is similar to biological species. We find that all species of butterflies have wings, but the color patterns, size and shape of the wings may differ greatly, and of course many other insect species have no wings. The differences between languages is not just in what kind of linguistic materials they use (some languages use tones, others intonation, etc.), or how they express information (morphologically, with word order, with intonation or stress). It is also in what information they have lexicalized into words or which usage patterns have become grammaticalized as entrenched constructions and adopted by the linguistic community as a whole.

It usually comes as a surprise, but there are many grammatical systems that occur in one language but are entirely absent from another one. A language may have an elaborate system of aspect and Aktionsart with a dozen distinctions being expressed with elaborate morphological markers (as in Russian), or it may lack such a system altogether (as in Japanese). A language may have a rich set of determiners for indicating the access status of the referents of a nominal phrase (as in English) or lack determiners altogether (as in many Slavic languages). Some languages express the roles of participants in events using morphologically expressed cases like nominative, accusative, dative, etc., which range from a few to close to a hundred different ones. Whereas other languages (like English) express the same information primarily through the ordering of constituents and the use of prepositions. Even languages that have case systems use them in quite different ways. According to typologists like Haspelmath (2007) it does not make sense to talk about ”the” genitive or “the” dative but only about a German dative or a Finnish dative, because they indeed differ profoundly in terms of which semantic roles they express and how they show in the utterance.

The observation that syntactic categorizations and semantic categorizations are language-specific is attested for all areas of grammar. Thus, it also does not make sense to talk about “the” noun or “the” verb because nouns and verbs behave in some languages in very different ways, and concepts lexicalized as verbs in one language may turn up as nouns in another and vice versa. Another very simple example is (syntactic) gender. Gender distinctions go back to the natural gender distinction between males and females, but when it starts to play a role in grammar it has to be expanded to all objects, including non-animate ones. Some languages, like French, make only two gender distinctions (masculine versus feminine), others languages, like German, use three (masculine, feminine, neuter) and some languages, like Japanese, do not have syntactic gender at all. The assignment of gender to inanimate objects also differs from one language to the next. The moon is masculine in German (”der Mond”), but feminine in Italian (”la luna”). Even animate objects can take conflicting and arbitrary genders. A child is neuter in Dutch (”het kind”) regardless of its natural gender. Bantu languages as well as Australian aboriginal languages use classifier systems which can be seen as more elaborate versions of gender systems, in the sense that they use many more than 2 or 3 distinctions. The classes used in classifier systems go back to natural categories, but they are quite non-intuitive to a non-native speaker. A famous example is the category ‘Women, Fire and Dangerous Things’ which is one of the classes that determines morphological markers in Australian aboriginal languages (Lakoff, 1987).

3. A third important insight from empirical linguistic research is that constructions are constantly on the move. New constructions appear, old ones may change or disappear, which is obviously the case for lexical constructions, with new words popping up, words shifting and expanding their meaning or becoming more restricted, and other words disappearing. The same happens for grammatical constructions, such that a language can have no articles (as in Latin), but its descendants, in this case French, Italian, Spanish, all have it. There can also be a stage in a language with no significant expression of aspect, followed by the evolution of sophisticated aspectual markers (as in Russian). Old English had a complex case system comparable to Greek, which then eroded and was replaced in the 14th century by an alternative system relying on the ordering of constituents and prepositions (Van Kemenade, 1987). Not only the constructions but also the categorizations used in constructions may undergo significant change, where new categorizations may come up and old ones disappear. Many linguistic changes occur because the syntactic or semantic categorizations employed in families of grammatical constructions shift and themselves enact various other changes. For example, demonstratives

(such as Latin “ille” or “illa”) often expand their function to become determiners (as in French “le” or “la”) which then gives rise to a distinctive syntactic category of articles with its own syntactic and semantic properties. (Diessel, 1999)

The literature is abundant and contains many concrete examples how grammatical constructions have evolved (Heine & Kuteva, 2002), but language evolution is certainly not a thing of the past. Even within the course of a single conversation, a new construction may come up, either because a speaker needs to stretch the meanings of existing words or the usage pattern of known grammatical constructions, in order to express new shades of meaning, or to capture the attention of the hearer in a novel, hence more forceful, way. This novel use may then be picked up by the hearer and possibly start to propagate further in the population, or it may die immediately, never to be used again (Garrod & Anderson, 1987).

3. Construction-based Language Processing

A constructional approach to language has abundantly proven its worth in descriptive linguistics and is also used almost universally in second language teaching. Moreover empirical evidence from child language acquisition shows that language learning can be understood by the progressive usage-based acquisition of constructions (Lieven & Tomasello, 2008). The constructional perspective has also been very productive for historical linguists (Fried, 2009). There is now a large body of clear examples showing how new constructions typically develop from the creative extension of existing constructions by a few individuals to a productive common pattern that is adopted by the linguistic community as a whole (Bybee, 1998). However, in this book we are primarily interested in developing adequate models of language processing, and we will see that the constructional perspective is also highly valuable for this.

3.1. Parsing and Production

Language processing has two sides: A speaker needs to convert meaning (taken in a broad sense to include pragmatic and communicative function) into a full specification of the form of an utterance. And a hearer needs to start from this form (again taken in a broad sense to include intonation, stress pattern or any other feature of the utterance) and reconstruct the most plausible meaning. These transduction processes are driven by input. In production it starts from the meaning that the speaker wants to convey and in comprehension it starts from the forms that could be extracted from the utterance. Clearly, knowledge of the language (linguistic competence) should be represented in such a way as to maximally support the transduction process so that the mappings can take place as fast as possible, even if the given input is incomplete or errorful.

We now know enough about language processing through attempts to model it in computational terms to realize that parsing and production is far from a straightforward, mechanical application of rules. It is a kind of problem solving or inferential process. The production of a sentence can be compared to the creative design, planning and manufacturing of a tool or a building, and the parsing of a sentence can be compared to the recognition of the structure and functioning of a tool and its subsequent use for achieving some specific purpose. When designing a tool, intermediary structures are formulated by the designer as part of the design and planning process, particularly if the goal is to arrive at a complex object. These intermediary structures are progressively enhanced to take into account more and more constraints until a concrete object can be built or understood. For example, the design of a new car might start from rough sketches which then get progressively refined according to precise engineering diagrams that take into account all functional requirements, the availability of materials and components, technological constraints, customer preferences, the organization of production lines, pricing constraints, etc.

Furthermore, the understanding of a tool (what it can be used for, how it should be used, how it was built) requires the development of rich intermediary representations that progressively clarify the tool and its functioning as well. For example, if we are confronted with a new complex machine, such as a new copier that can also send faxes, scan images, bind papers, etc., it will take a while before we have figured out what all the buttons and menus mean and how the printer has to be controlled from a computer. Gradually we will develop representations about what the machine can do and how its functionality is expressed and controlled through the interface.

All of this helps to clarify the nature of language processing and language learning. We now know that quite complex intermediary structures have to be built which progressively incorporate more details (in production) or recognize and integrate more features of the input (in parsing). In Fluid Construction Grammar these structures are called *transient structures*. They typically consist of a set of units, roughly corresponding to morphemes, words or phrases, as well as a lot of information attached to each of these units in the form of features, which may concern any aspect of language at any level: pragmatic, semantic, syntactic, morphological, phonological. For example, the transient structure for the phrase “the mouse” would contain units for “the” and “mouse”, and for the nominal phrase as a whole. The unit for "mouse" would contain information that its part of speech is noun, the number singular, that the meaning of "mouse" introduces a class of animate objects, etc.

Constructions (now used in the sense of the internal knowledge structures coding lexical and grammatical knowledge) package all information relevant to the usage pattern they cover in such a way that it can be used efficiently to expand transient structures from an initial state to a final state. Because a set of constructions are usually needed to build a complete sentence, a chain-like reaction sequence must occur, with a construction applied at each step:

$$T_{init} \rightarrow T_1 \rightarrow T_2 \rightarrow ... \rightarrow T_{final}$$

In language comprehension, the initial transient structure T_{init} contains whatever information can be gleaned from the utterance itself, plus possibly information on the context that already constrains meaning. The final structure T_{final} should contain enough information to enact the interpretation of the utterance. Each inference step then corresponds to the application of a construction, which expands a transient structure by adding more information. In language production, the initial transient structure T_{init} contains information on the meaning to be expressed as well as any other contextual or discourse constraint that might be relevant. The final transient structure T_{final} should contain everything that is needed to fully articulate the utterance.

How many constructions and hence processing steps are typically needed depends on the grain-size of a construction, but a moderate 10 word sentence would easily involve at least 50 constructions, some of them handling individual words and others adding further information to reach the final state.

The process of applying a construction involves a *matching step* to determine whether a construction is 'compatible' with the transient structure under consideration and a *merging step* to expand the transient structure with information contained in the construction. For example, the construction for the definite article "the" would check in parsing whether its form (an occurrence of the string "the") is indeed present in the input, and it would then add in the merge step information to the transient structure about the syntactic and semantic categorizations associated with "the" (for example that it is an article) and about its meaning. Constructions can therefore be thought of as active objects. They are active processes that are constantly on the look out as to whether or not they can do something to a particular transient structure, and, if so, they become active and carry out that change. Sometimes more than one construction can apply in parallel, whereas, at other times, another construction needs to have prepared the ground.

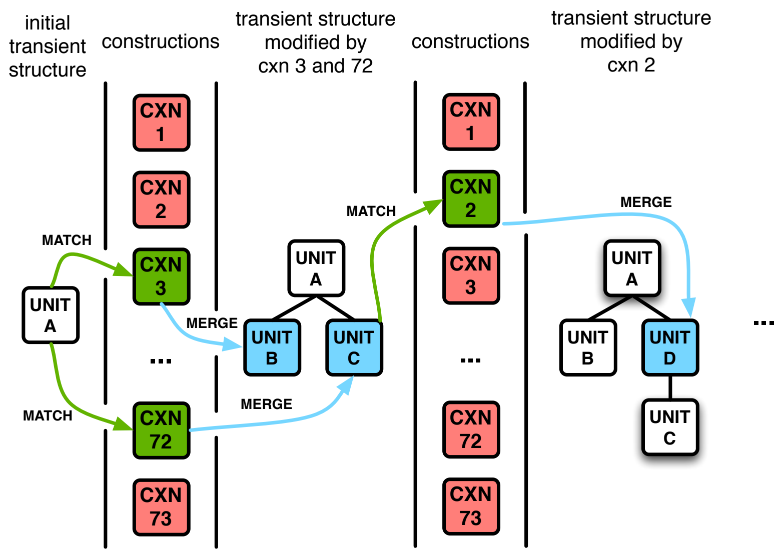
initial transient structure
constructions
transient structure modified by cxn 3 and 72
constructions
transient structure modified by cxn 2
CXN 1
CXN 2
CXN 3
...
CXN 72
CXN 73
MATCH
MATCH
MERGE
MERGE
UNIT A
UNIT B
UNIT C
UNIT D
...

Figure 3. *Constructions first check whether they are compatible with a given transient structure, and if so they extend the structure by building new units or by adding information to existing units, which creates an opportunity for other constructions to become applicable.*

3.2. Rich Chunks Versus Local Rules

Linguistic materials are usually *multi-functional*. Most words and grammatical patterns are ambiguous and polysemous. Hence, other elements in the context are critical for deciding which function is intended in a specific case. Human cognitive processing is remarkably adept for this. The English word "bank" can be used for a variety of things including a financial institution, the side of a river or a place where information is stored. Yet there is no hesitation as to the meaning of the sentence "She went to the bank to deposit a check". The German article "der" can be nominative masculine singular, dative or genitive feminine singular, or genitive masculine or feminine plural, but as soon as a German speaker hears "Der Tisch ist gedeckt" ("The table is ready"), he or she knows instantly that "der" is used as the nominative masculine singular.

The advantage of multiple uses of the same linguistic materials is that a language needs fewer of them and hence there is less load on memory. The German article expresses four cases, two numbers and three genders which gives 24 possible combinations of values, but there are only six articles (”der”, “die”, ”das”, “dem”, “den”, “des”). The disadvantage is that a linear chain of construction applications from initial to final state is no longer possible because at each step a multitude of possible constructions can expand a given transient structure. If a parser encounters the word “der” there are five possible nouns that could be expected (Nominative Masculine Singular, Dative Feminine Singular, Genitive Masculine or Feminine or Neuter Plural).

In general, a *search space* arises in which each of these paths is explored, with most paths leading to dead-ends:

$$\begin{array}{ccccccc} & \rightarrow & T_{1,1} & \rightarrow & ... & & \\ T_{init} & \rightarrow & T_{1,2} & \rightarrow & T_2 & \rightarrow & ... \\ & \rightarrow & T_{1,3} & \rightarrow & ... & & \\ & & & \rightarrow & T_3 & \rightarrow & ... \end{array}$$

Although some search is acceptable, indeed unavoidable, relying too much on search is dangerous because of combinatorial explosions. If every word in a sentence has 10 possible meanings or functions (a low estimate) then a 10 word sentence already requires the consideration of 10^{10}= 10,000,000,000 processing chains, and we have not even taken into account that the same syntactic structure can also have different meanings or functions. Language production requires as much search as language comprehension because the same combination of meanings can often be expressed in a multitude of ways, and the speaker has to find a solution that fits with all the other choices being made to build a complete sentence.

The search problem is *the* major issue in language processing, and certain aspects of grammar become comprehensible only when it is taken into consideration. Often a change in the representation of the grammar can do a lot towards avoiding a combinatorial explosion, as illustrated in the later chapter on agreement systems (van Trijp (2011b)). One of the most remarkable aspects of human language processing is that it goes extremely fast, despite the relatively slow human brain compared to contemporary electronic computers and despite errorful input (van Turennout et al., 1998). This amazing performance suggests that there are plenty of grammatical cues that allow listeners to quickly grasp the meaning of an utterance and allow speakers to rapidly fetch ready-made usage patterns, minimizing search as much as possible.

How does a construction-based organization of linguistic competence help to deal with the search problem? A construction brings together a large number of constraints in a single rich data structure which can consequently be applied in one processing step. Therefore, a lot of search disappears because semantic or pragmatic constraints and constraints on morphology or phonology can be brought to bear immediately. Features of other elements in the context can be taken into account, even if they are very specific or buried deep down somewhere in a linguistic structure.

A construction-based organization also helps to explain *robustness*. Elements are frequently missing from the linguistic input or, even worse, there can be errors, like pronunciation errors, non-standard use of words, half-finished fragments, and so on, that inhibit or complicate further processing steps. Such errors happen very frequently in language, particularly in speech, because some sounds may not have been accurately produced by the speaker, or the hearer's speech system was unable to recognize them. Some words or constructions may have been used inappropriately by the speaker because of sloppiness or due to errors and mistakes in planning (Steels & van Trijp, 2011). Problems may also arise because the language systems of different speakers of the same language are never entirely the same, and so the hearer may be confronted with an ungrammatical sentence which is perfectly grammatical within the ideolect of the speaker.

A construction-based organization is a more promising route to achieve robustness because constructions can be matched in flexible ways, even if some elements are missing. If enough of a construction fits with the current context to be confident that it should apply, it can supply information in a top-down manner about the missing or errorful elements and enable further processing. When structure building operations are purely local, so that the effect of a single construction requires dozens of steps, language processing not only becomes much slower but also very brittle. As soon as a single problem occurs, processing gets stuck. The parser is unable to deal with errors, because it has no access to the bigger picture.

4. Fluid Construction Grammar

Translating the intuitions of construction grammar into a system that is both formal and computationally effective is far from easy. Computational construction grammars need at least the following facilities:

1. A way to represent transient structures: These are the structures created on the fly during parsing and production. FCG uses feature structures for this

purpose. Feature structures consist of a set of units, roughly corresponding to morphemes, words or phrases, and features for each unit with specific values. For example, one feature of a unit may define the set of subunits it has, another one what the meaning is contributed by this unit. A value can itself consist of a set of features and values.

2. A way to represent constructions: FCG also uses features structures for this task, although the features structures are more abstract compared to transient structures. They make more use of variables and contain various operators for partial matching and for grabbing, building and manipulating hierarchical structures.

3. A way to apply constructions to transient structures (that is, to use information from the definition of a construction to expand a given transient structure): This operation forms the core of the FCG-interpreter and is the same for parsing and producing, except that the direction of application reverses. FCG uses variants of unification for this purpose.

4. A way to orchestrate the successive application of constructions: this facility implies mechanisms for setting up a search space and for monitoring progress.

There are many ways to implement these facilities, depending on what representational and computational mechanisms are adopted as underlying foundation. FCG uses techniques now common in formal and computational linguistics, such as the representation of linguistic structures with feature structures (Carpenter, 1992; Copestake, 2002), and the use of unification for applying constructions to expand linguistic structures in language parsing and production, as pioneered in Functional Unification Grammar (Kay, 1986), and also used in Lexical Functional Grammar (Dalrymple et al., 1995), and Head-driven Phrase structure Grammar (Pollard & Sag, 1994; Sag et al., 2003). Like many other computational linguistics efforts, the FCG-system is embedded within a contemporary Common LISP-based programming environment from which it inherits well tested mechanisms for representing and processing complex symbolic structures (Norvig, 1992). Other proposals for operationalizing construction grammar, such as Embodied Construction Grammar (Bergen & Chang, 2005) and Sign-Based Construction Grammar (Michaelis, 2009), draw on mechanisms arising from the same computational tradition but use them in different ways. Given the current state of the field, it is highly beneficial that many approaches are explored in order to discover the best way to formalize and implement construction grammars.

4.1. Basic principles

FCG represents constructions as much as possible in a *declarative way*, which means that constructions take the same form as transient structures with units, features and values, instead of a *procedural way*, which would mean that the construction codes directly the operations to expand a transient structure. Declarative definitions are a general feature of unification grammars (Sag et al., 1986). Using the same representations for constructions as for transient structures has significant advantages. It makes FCG easier to learn and implement as all internal data structures and operations that work for transient structures can automatically be used for constructions, including ways to browse through them. It also opens the way for implementing learning strategies that start from concrete transient structures and develop constructions by making parts of these structures variable.

In addition, there is *no formal distinction* in FCG between different types of constructions, whether they are grammatical or lexical, or whether they are very abstract or item-based. Consequently, there is no a priori architectural division between when and how constructions are supposed to be used in processing. Constructions trigger as soon as they are able. This functionality is in line with the non-modular character of construction grammars and makes possible the integration of all relevant constraints in a construction.

Next, FCG adopts the *reversibility principle* which means that the same constructional definition must be usable without change both in parsing and production and without compromising efficiency or generating unnecessary search. A construction therefore defines a *bi-directional pairing* between aspects of meaning (captured in the semantic pole) and aspects of form (in the syntactic pole). In language production, constructions trigger based on their semantic pole and add information contained in the syntactic pole. In parsing, they trigger based on the syntactic pole and add information contained in the semantic pole.

The reversibility principle has been an important goal of computational linguistics for a long time and was one of the primary motivations for unification-based grammars (Kay, 1986). However, in practice most grammar formalisms focus either on parsing or on producing, but not on both at the same time. Reversibility is hard to achieve because the representation of information usually has a strong impact on the efficiency with which it can be used for a particular purpose. For example, a telephone book allows you to look up the number given the name of a person but is quite useless if you need to find the name of a person, given a telephone number. Similarly, an efficient representation for parsing may not at all be suited for production and vice-versa.

Why nevertheless insist on reversibility? There are two reasons:

1. If there were two separate representations, language users would need to double the memory resources required for storing knowledge of their language, and they would continuously need to coordinate both representations if they are learning new aspects of their language, requiring complex bookkeeping and translation operations.

2. Once constructions are bi-directional it becomes possible to constantly move back and forth between parsing and production: when a sentence is being produced, the FCG-interpreter can monitor progress by constantly re-entering the structures already produced using its repertoire of constructions and parsing them. Conversely, when a sentence is being parsed, the FCG-interpreter can fill in gaps of missing elements or fix errors by switching to a production mode in which the partial structures already derived from the input are expanded by the reverse application of constructions.

Human language learners are often able to produce less than they can understand, and this can be seen as a counter-argument for the bi-directionality of constructions. However, language is an inferential coding system (Sperber & Wilson, 1995). The context and world knowledge are often enough to grasp the meaning, while grammar is in many cases helpful but not absolutely necessary. For example, the phrase "child dog bite", which does not code grammatically information about 'who does what to whom', would normally be understood as "the dog bites the child" and not "the child bites the dog", simply because it would be very unusual for a child to do this. Or, the phrase "he come yesterday" tells us that the event of coming was in the past, even though it is not grammatically marked as it is in "he came tomorrow". On the other hand, language speakers never produce sentences that they would not be able to understand themselves. They constantly monitor and possibly correct or improve their own utterances while they are speaking.

4.2. Efficiency Issues

These basic principles are useful design goals, but we must also ensure that the representations of constructions contain the information necessary to perform parsing and production fast enough, which means that the consideration of constructions must be done as efficiently as possible and search must be avoided. One way, mentioned earlier, is that as many constraints as possible are captured in a construction so that all possible cues can be considered whenever possible.

FCG tackles the efficiency issue by structuring constructions internally, by using footprints and by organizing grammars into sets and networks:

1. Rather than using a single feature structure for a construction, as is typically done in unification grammars, FCG splits the construction definition into *a semantic and a syntactic pole*. The two poles have different roles in parsing and production. In parsing, the syntactic pole is the conditional pole, and the semantic pole the contributing one. For example, the determiner-nominal construction will only apply if a determiner and a nominal can be found and if they exhibit other syntactic properties, like the determiner comes before the nominal and they agree in number. The syntactic pole therefore acts as a kind of first check whether it is useful to consider the construction at all. This first check is done by a matching operation, which is easier and faster than the full merging operation, which is done only when the first check succeeded. In production, the semantic pole is the conditional pole, and the syntactic pole is the contributing one. For example, the meaning associated with a lexical construction will be matched against the meaning that must be expressed and only if that is the case, the word form and various other categorizations are added. Again, it helps to determine whether a construction should be considered before doing more complex operations. The split into two poles is not totally strict because the conditional pole may also contain information that will have to be merged with the transient structure and the contributing pole may still block the application of a construction if some of its features are incompatible. Nevertheless, this division of labor is an important instrument for making construction application more efficient.

2. There is often a need to prevent a construction from being considered, either because it has already been applied, or there are more specific constructions which have already done their job and so it is no longer necessary to apply more general constructions. FCG uses *footprints* to deal with these issues. Footprints are markers which are added by a construction when it has been able to apply. Constructions can then test for the presence of these footprints and thus quickly determine whether it make sense to try those constructions.

3. A third way to improve efficiency is by organizing the construction inventory into sets and networks. Once *construction sets* have been defined, each set can be invoked and applied separately, and an ordering can be defined over when a particular set has to be considered. Thus, rather than putting all constructions

in a single pot, lexical constructions are able to be separated from grammatical constructions and applied first, which obviously minimizes the time needed to find a construction that might possibly apply. A more fine-grained way to exercise control over which constructions should be considered is by organizing them into *construction networks*, which encodes relations between constructions and can potentially be exploited to prioritize which constructions should be considered (Wellens, 2011). FCG supports the definition of many different types of networks with each a specific utility. For example, a specialized construction should be considered before its more generic counterpart, even though the more generic construction triggers whenever the specialized construction triggers. Thus, by defining specialization relations between constructions, this functionality is automatically achieved.

4.3. Damping search

The problem of search comes primarily from the fact that most linguistic elements have multiple functions. The same word may belong to multiple lexical categories (adjective, adverb, preposition), or it may express competing bundles of syntactic features. A verb may have different case patterns (valences) so that it can be used in a variety of constructions. The same spatial relation may be expressed through different words depending on the syntactic context. The consequence of this phenomenon is that there is often not enough information available in the transient structure to take the next step. At the same time, language processing should not halt because it is only when more information is derived that a decision can eventually be made. Of course, it is always possible to set up a search space in which the different possibilities are explored, and, indeed, this is often assumed by grammar writers. However, for realistic grammars, there are too many possibilities so that the search space becomes very large. Just assuming that search will do the job is not a viable approach to explain why human language processing is so fast.

Although in FCG, every effort is made to avoid search, this is not always possible, simply because there are not enough constraints to make a decision on local criteria only. FCG therefore supports the representation of open decisions and multiple possibilities as part of the transient structure. There are three main mechanisms in FCG for doing that:

1. It is possible to leave choices open by using *variables*. These variables become bound whenever information is available to resolve them. Many constraints can already be expressed and enforced if that is not the case. For

example, although there is agreement between the article and the noun for the number in a nominal phrase, in a phrase like “the sheep” there is, nevertheless, not enough information on the number because both “the” and “sheep” can be plural as well as singular. The nominal phrase as a whole gets the number feature of its constituents, but if we do not know what number is the case, a specific value cannot percolate. However, by using the same variable for the number value of the article and the noun, the agreement relation can already be enforced, and, by using again the same variable for the number feature of the nominal phrase, we can express that the value for number percolates, even though we do not know yet what the number is. When more information becomes available, for example through the agreement between subject and verb as in the sentence ”the sheep lives in a pen”, the number ambiguity can be resolved and the right value propagates through variable sharing to all units.

2. The use of variables is made more sophisticated by organizing them in *feature matrices* (van Trijp, 2011b). These matrices combine a number of dimensions that typically co-occur and interfere with each other. For example, decisions on number, gender and case all interact in establishing the morphological form of the German article, and constraints to help make a decision for each of these feature dimensions may come from many different sources. For example, case decisions may come from the case pattern of the verb or from a marking of the noun, number may come from semantic properties of the referent of the nominal phrase and gender from the chosen noun. Feature matrices contain variables when a value is still unknown, and + and - values when the feature is or is not present. Identical variables are used to express relations between feature values without knowing yet what their values are. Interestingly, the ‘normal’ unification operations can be used to combine these matrices and progressively resolve uncertainty (as discussed in more detail later (van Trijp, 2011b)).

3. Yet another technique commonly used in FCG to avoid search is to make a distinction between the *potential* values of a feature (for example, the different lexical categories that a word may have) and the *actual* value. As long as the actual value could not yet be determined, the set of potential values is represented as a disjunction in the transient structure. Some constructions may further restrict the potential, but it is only when all information is available that a construction will come along to determine the actual value. Various

examples of this design pattern are described later in this book (Spranger & Loetzsch, 2011; van Trijp, 2011a).

4.4. Flexibility and Fluidity

The next critical issue for language processing is to deal with the fluid, open-ended nature of language and the fact that utterances in normal discourse may actually be ungrammatical due to hesitations or errors but also due to the unreliability of speech recognition. Realistic utterances usually are only fragments of sentences and may contain missing words, words that are not perfectly recognizable, grammatical errors and so on. One approach to these problems is to make language processing probabilistic (Bod et al., 2003), which means that all constructions or aspects of constructions have probabilities associated with them so that parsing becomes a matter of calculating probabilities rather than determining with certainty how a sentence should be parsed. Even a sentence that is to some extent ungrammatical could still be analyzed, although it would get a low score.

FCG goes somewhat in this direction because each construction has an associated *score* (in line with earlier proposals, by Jurafsky (1998), a.o.). Scores play a role in the search process. Constructions with a higher score are preferentially explored. The approach is not entirely probabilistic, however, because the scores are assumed to reflect success in communication rather than frequency. There is of course a relation between the probability that an utterance may occur or be used by a speaker and the success that the speaker is expected to have with the constructions used in that utterance. However, the two are not the same.

Probabilistic grammars and scoring helps to deal with variation in language, because competing constructions can exist side by side in the inventory. The speaker prefers one way of speaking (the one with the highest score) but can still parse sentences that are based on constructional variants. Yet, this mechanism does not yet help in dealing with incomplete sentences or unknown lexical or grammatical elements.

The FCG-interpreter uses two levels of processing: (i) A routine layer at which constructions are applied transforming transient structures to map form to meaning (in parsing) or meaning to form (in production). The FCG-interpreter is not concerned with establishing grammaticality but with applying as many constructions as possible. (ii) A meta-layer at which diagnostics are run and repair strategies possibly triggered. Diagnostics test for example whether all words in the input could be integrated into the sentence as a whole, or whether all the meaning that the speaker wanted to express are actually part of the final utterance. Repair strategies then try to fix this situation, possibly by ignoring some of the input

Here are two examples which are illustrated elsewhere in more detail (Steels & van Trijp, 2011):

1. It is possible to introduce units for unknown words or words that could not be recognized by the speech system. These units are initially empty, but when constructions start to trigger, they begin to fill in aspects of the unknown word thanks to the rest of the context, up to a point where it is sometimes possible to actually reconstruct the word form, particularly if the FCG-system is embedded in a more encompassing language system that has also strong semantic capacities. In case the word is unknown, this process leads to strong hypotheses about what a missing construction should look like.

2. It is possible to stretch the usage of existing constructions in production by *coercion*. For example, comprehending the famous sentence "Mario sneezed the napkin off the table" requires permitting an intransitive verb ("sneeze") to obtain a causative meaning because it is used in a trivalent pattern that is typically used for causative verbs as in "he pushed the box off the table". Normal routine application of constructions would not be able to handle this, but, by imposing the trivalent argument structure construction, an appropriate interpretation can nevertheless be obtained.

5. Design patterns and Templates

Writing down constructions is not at all an easy task, even for an experienced computational linguist. The grammar designer needs to consider not only whether all the right linguistic constraints have been captured properly, both for parsing *and* for production, but also when and how a construction is best applied and how to avoid an explosion of the search space. Many design and engineering fields have developed approaches and methodologies for coping with complexity, and this is also the case in grammar engineering (Bender et al., 2002). In Fluid Construction Grammar, two techniques are used to manage design complexity: design patterns and templates.

5.1. Design Patterns

The notion of a design pattern comes from architecture (Alexander, 1979) but is now widely used in software engineering (Gamma et al., 1995). An architectural design pattern is for instance a dome structure for spanning a very large space (such as the Santa Maria del Fiore Duomo in Florence built by Bruneschelli). There are general principles of dome design but specific details will depend on the required size and height of the space that needs to be covered, the available building materials as well as on aesthetic considerations. Moreover the same space could also be covered with another kind of structure, for example by a roof with two slopes.

In the context of grammar, a design pattern circumscribes the core solution to a particular issue of language. A good example is the use of an agreement system to signal which constituents of a nominal phrase hang together. Some languages associate and occasionally explicitly mark various syntactic features of constituents (e.g. number, gender, case). They then exploit these features to signal constituent structure. There are still some remnants in English of an agreement system (e.g. the determiner and nominal have to agree for number) but English prefers word order, however we see rich uses of this design pattern in many other languages, e.g. Slavic languages like Polish, where there will be agremeent for gender, number, and case and occasionally other features like animacy or personhood. Another example is the use of field topology to determine sentence structure. A field is a slot that can be filled with a particular constituent depending on various constraints. For example, most German declarative clauses have a number of fields with the verb often in the second field. The first field can be filled by many other constituents, not just the subject. The German case system is strong enough so that constituent structure does not need to be less based on ordering compared to English. The ordering of constituents is not free but a carrier of additional meaning, particularly information structure. The specific details how a design pattern is instantiated in a particular language may vary considerably and some languages may use certain design patterns which are entirely absent from others. Nevertheless, it is extremely useful to approach the analysis and design of a grammar by first inquiring what kind of design patterns have been adopted by speakers of that language.

5.2. Templates

Another well known technique in computer science for handling complexity is to introduce abstractions that encapsulate complex datastructures and procedures. For example, rather than having to write an implementation of a procedure for sorting the elements of a list each time, a programmer typically pulls a general sorting procedure out of a library and plugs it into his or her code, possibly after setting some parameters such as which function should be used for checking when one element should precede another element. The same approach is advocated for Fluid Construction Grammar. A particular design pattern, such as agreement or field topology, is implemented using a number of templates that hide much of the operational detail so that the grammar designer can focus on the linguistic content.

For example, suppose that we want to define a construction for building (or recognizing) a determiner nominal phrase, like the English phrase "the table". This construction will have to deal with a number of different issues (seed the later chapter on phrasal constructions (Steels, 2011)):

1. It should specify the *functional constraints* on the units and what the phrase type and possible functions are of the phrase as a whole, in this case that the construction is dealing with units that have the syntactic functions of determiner and nominal respectively and the semantic functions of reference and identifier.

2. It should specify what the phrase type and possible functions of the parent phrase are going to be, in this case, that the combination of determiner and nominal yields a unit with the syntactic type nominal phrase and the semantic function of referring expression.

3. It should specify the *agreement relations* between syntactic features associated with each unit, namely that the determiner should agree in number. In other languages, other features would be relevant such as gender (in French) or case (in German).

4. It should specify *semantic constraints* on the units; for example, the countability feature (mass noun vs. count noun) needs to be compatible between the article and the noun.

5. Next there is the *percolation* of some of the features of the constituents to the parent, which, for the determiner nominal phrase, is at least the case for definiteness (coming from the article) and number (coming from both).

6. The construction also needs to contain information on how the meanings supplied by each of the constituents should be *linked together* to form the meaning of the phrase as a whole.

7. Finally, a construction typically introduces *additional meaning* or *additional form* which is not in any of the parts. For example, the determiner nominal

phrase construction would add information on getting the semantic context within which the referring expression operates, and it would impose a particular ordering on the constituents, namely that the determiner precedes the nominal.

All of these different issues might be handled by different templates (the grain size of templates is the choice of the designer). For example, there might be a template specifying how meanings of the constituents should be linked or a template specifying agreement relations. To build the construction itself, each template needs to be supplied with specific information. For example, the agreement template needs to be told which features of which constituents have to agree. The template is then able to expand the skeleton of the construction already built by other templates with the necessary details to handle the issue for which it is responsible. For example, if an agreement template is told that agreement concerns the features *number* and *definiteness* between the determiner and nominal, then the appropriate constraints should be added to the construction to make sure that these agreement relations are enforced.

Templates can express the kind of principles that linguists working in other formal traditions are seeking. For example, the percolation template might simply specify that all features contained in the head of the phrase need to percolate to the parent (Pollard & Sag, 1994). It is then enough to specify which constituent is the head, and the template can add all the necessary elements to the construction that are required to implement this form of percolation. A linguist who does not want to use the head principle might instead use another template in which he or she can specify explicitly which features from which different units percolate.

Fluid Construction Grammar does not make claims about whether there is a universal set of templates shared by all languages or even what the ideal grain-size is for the templates needed for a specific language. The possible set of templates is considered to be open, and new templates can be easily added. Of course, there are clearly commonalities in the set of templates that are needed for different languages, particularly if the languages are coming from the same linguistic family. At the same time, it is truly amazing how profoundly languages can differ.

6. Conclusions

This chapter introduced some of the key ideas behind construction grammars in general and Fluid Construction Grammar in particular. It argued that the construction grammar perspective is not only valuable from a linguistic or psycholinguistic perspective. It clearly yields a more viable approach to language processing because it helps to explain the amazing speed, robustness and flexibility of human language. This chapter then provided a broad discussion of some of the general design principles used in FCG, focusing in particular on how to deal with efficiency issues without giving up on reversibility and declarative definition, how to deal with the fluidity and open-ended nature of human language, and how to achieve a more effective way for designing large grammars by the use of templates.

Acknowledgements

The research reported here was conducted at the Sony Computer Science Laboratory in Paris and the Artificial Intelligence Laboratory of the Free University of Brussels (VUB). It received additional funding from the the EU-FP6 project ECAgents and the EU-FP7 project ALEAR.

References

Alexander, Christopher (1979). *The Timeless Way of Building.* Oxford: OUP.

Baker, Colin, Charles Fillmore, James Lowe (1998). The berkeley framenet project. In *Proceedings of the 17th international conference on Computational linguistics.* Morristown, NJ: Association for Computational Linguistics.

Bender, Emily, Dan Flickinger, Stephan Oepen (2002). The grammar matrix: an open-source starter-kit for the rapid development of cross-linguistically consistent broad-coverage precision grammars. In *Proceedings of the Workshop on Grammar Engineering and Evaluation at the 19th International Conference on Computational Linguistics*, 8–14. Taipei.

Bergen, Benjamin, Nancy Chang (2005). Embodied Construction Grammar. In Jan-Ola Östman, Mirjam Fried (Eds.), *Construction Grammars: Cognitive Grounding and Theoretical Extensions*, 147–190. Amsterdam: John Benjamins.

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.* Amsterdam: John Benjamins.

Bod, Rens, Jennifer Hay, Stefanie Jannedy (Eds.) (2003). *Probabilistic Linguistics.* Cambridge, Ma: MIT Press.

Bybee, Joan (1998). A functionalist approach to grammar. *Evolution of Communication*, 2, 249–278.

Carpenter, Bob (1992). *The Logic of Typed Feature Structures*. Cambridge: Cambridge University Press.

Copestake, Ann (2002). *Implementing Typed Feature Structure Grammars*. Stanford: CSLI Publications.

Croft, William (2001). *Radical Construction Grammar: Syntactic Theory in Typological Perspective*. Oxford: Oxford UP.

Daelemans, Walter, Antal Van den Bosch (2005). *Memory-Based Language Processing*. Studies in Natural Language Processing. Cambridge: Cambridge University Press.

Dalrymple, Mary, Ron Kaplan, John Maxwell, Annie Zaenen (Eds.) (1995). *Formal issues in Lexical-Functional Grammar*. CSLI Lecture Notes 47. Stanford CA: CSLI.

Diessel, Holger (1999). *Demonstratives: Form, Function, and Grammaticalization*. Typological Studies in Language 42. Amsterdam: John Benjamins.

Evans, Nicholas, Stephen Levinson (2009). The myth of language universals. *Behavioral and Brain Sciences*, 32, 429–492.

Fellbaum, Christiane (1998). *WordNet: An electronic lexical database*. Cambridge Ma: MIT Press.

Fillmore, Charles (1968). The case for case. In E. Bach, R. Harms (Eds.), *Universals in Linguistic Theory*, 1–88. New York: Holt, Rhinehart and Winston.

Fried, Mirjam (2009). Construction grammar as a tool for diachronic analysis. *Constructions and Frames*, 1(2), 261–290.

Gamma, Erich, Richard Helm, Ralph Johnson, John Vlissides (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.

Garrod, Simon, Anne Anderson (1987). Saying what you mean in dialogue: A study in conceptual and semantic coordination. *Cognition*, 27, 181–218.

Goldberg, Adele (1995). *A Construction Grammar Approach to Argument Structure*. Chicago: Chicago UP.

Goldberg, Adele (2006). *Constructions At Work: The Nature of Generalization in Language*. Oxford: Oxford University Press.

Goldberg, Adele, Ray Jackendoff (2004). The english resultative as a family of constructions. *Language*, 80(3), 532–568.

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

Haspelmath, Martin (2007). Pre-established categories don't exist. *Linguistic Typology*, 11(1), 119–132.

Heine, Bernd, Tania Kuteva (2002). *World Lexicon of Grammaticalization*. Cambridge University Press.

Jackendoff, Ray, Adele Goldberg (2004). The english resultative as a family of constructions. *Language*, 80(3), 532–568.

Jurafsky, Daniel (1998). A probabilistic model of lexical and syntactic access and disambiguation. *Cognitive Science*, 20(2), 137–194.

Kay, Martin (1986). Parsing in functional unification grammar. In Barbara Grosz, Karin Sparck-Jones, Bonny Webber (Eds.), *Readings in Natural Language Processing*. Morgan Kaufmann.

Kay, Paul, Charles Fillmore (1999). Grammatical constructions and linguistic generalizations: The what's x doing y? construction. *Language*, 75, 1–33.

Lakoff, George (1987). *Women, Fire, and Dangerous Things: What Categories Reveal about the Mind*. Chicago: The University of Chicago Press.

Lieven, Ellen, Michael Tomasello (2008). Children's first language acquistion from a usage-based perspective. In Peter Robinson, Nick Ellis (Eds.), *Handbook of Cognitive Linguistics and Second Language Acquisition*. Routledge.

Lyons, Christopher (1985). The syntax of english genitive constructions. *Linguistics*, 12, 123–143.

Michaelis, Laura (2009). Sign-based construction grammar. In Bernd Heine, Heiko Narrog (Eds.), *The Oxford Handbook of Linguistic Analysis*, 155–176. Oxford: Oxford University Press.

Michaelis, Laura, Knud Lambrecht (1996). Toward a construction-based model of language function : the case of nominal extraposition. *Language*, 72, 215–247.

Norvig, Peter (1992). *Paradigms of Artificial Intelligence Programming. Case Studies in Common Lisp.* San Francisco: Morgan Kauffman.

Östman, Jan-Ola, Mirjam Fried (2004). Historical and intellectual background of construction grammar. In Mirjam Fried, Jan-Ola Östman (Eds.), *Construction Grammar in a Cross-Language Perspective*, 1–10. John Benjamins Publishing Company.

Pollard, Carl, Ivan Sag (1994). *Head-Driven Phrase Structure Grammar.* Chicago: University of Chicago Press.

Sag, Ivan, Ron Kaplan, Lauri Karttunen, Martin Kay, Carl Pollard, Stuart Shieber, Annie Zaenen (1986). Unification and grammatical theory. In *Proceedings of the Fifth Annual Meeting of the West Coast Conference on Formal Linguistics*, 238–254. Stanford SLA, CSLI Publications.

Sag, Ivan, Thomas Wasow, Emily Bender (2003). *Syntactic Theory. A Formal Introduction.* CSLI Publications, second edn.

Sperber, Dan, Deirde Wilson (1995). *Relevance: Communication and Cognition.* Cambridge, MA: Harvard University Press.

Spranger, Michael, Martin Loetzsch (2011). Syntactic indeterminacy and semantic ambiguity: A case study for German spatial phrases. In Luc Steels (Ed.), *Design Patterns in Fluid Construction Grammar*. Amsterdam: John Benjamins.

Steels, Luc (1998). The origins of syntax in visually grounded robotic agents. *Artificial Intelligence*, 103(1-2), 133–156.

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

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.

Stefanowitsch, Anatol, Stefan Gries (2003). Collostructions: Investigating the interaction of words and constructions. *International Journal of Corpus Linguistics*, 2(8), 209–243.

Van Kemenade, Ans (1987). *Syntactic Case and Morphological Case in the History of English*. Dordrecht: Forist Publications.

van Trijp, Remi (2011a). A design pattern for argument structure constructions. In Luc Steels (Ed.), *Design Patterns in Fluid Construction Grammar*. Amsterdam: John Benjamins.

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

van Turennout, Miranda, Peter Hagoort, Colin Brown (1998). Brain activity during speaking: From syntax to phonology in 40 milliseconds. *Science*, 280(5363), 572–574.

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