What triggers the emergence of grammar?

Luc Steels
*University of Brussels (VUB)
Pleinlaan 2 1050 Brussels
Sony Computer Science Laboratory Paris
`steels@arti.vub.ac.be`

Abstract

The paper proposes that grammar emerges in order to reduce the computational complexity of semantic interpretation and discusses some details of simulations based on Fluid Construction Grammars.

1 Introduction

There has been a flurry of recent theoretical models trying to explain how and why human languages may have evolved grammatical structures (Hashimoto and Ikegami (1996), Nowak and Krakauer (1999)), and there has been a growing series of computer simulations and robotic experiments applying such models to evolve grounded communication systems in artificial agents (Cangelosi and Parisi (2001), Briscoe(2002), Steels(2003)). The problem of the origins of grammar is obviously a key question in research on the origins and evolution of languages, and it is only when we have clear theoretical models that we can hope to reconstruct the ontogenetic and phylogenetic pathways towards grammatical language.

Most research reported so far views the problem of verbal communication as a coding problem where a meaning M is coded by the speaker into an utterance u and then u is decoded by the hearer to reconstruct the same meaning M. Both processes are a function of the lexicon and grammar, further called the language inventory, of the speaker I_s and hearer I_h, so that $code(M, I_s) = u$ and $decode(u, I_h) = M$.

It is common to argue that syntax arises to make both the size of the inventory and the length of the utterance for a given meaning more optimal (Nowak and Krakauer,1999). Language inventory can be minimised by using a compositional as opposed to a holistic coding. Utterance length can be minimised by coding certain aspects of meaning using syntactic means such as word order or hierarchy. A smaller inventory makes it easier to learn the language and it has therefore been argued that the learning bottleneck (i.e. the fact that language learners are only exposed to a limited number of sentences) encourages agents to choose a compositional as opposed to holistic coding (Smith, et.al., 2003). Although it cannot be denied that syntax has this kind of optimising effect, this paper proposes a different explanation for the role and therefore the emergence of grammar. Specifically, I will argue that the first primary function of grammar (but not the only one) is to optimise semantic interpretation. I will also argue that 'true' grammar only arises when there is an intermediary layer of linguistic categories and constructions as opposed to syntactic structure only.

The paper first defines formally the problem of semantic interpretation and characterises its computational complexity. It then reflects on the nature of grammar and argues that grammar only arises when there is an extra intermediary layer of syntactic and semantic categories that mediates between form and meaning. The paper then explores a peer-to-peer negotiation approach to the origins of grammar, in which grammatical categories and constraints on the use of these categories are progressively built and coordinated by the agents, triggered by the need to optimise semantic interpretation.

2 Semantic Interpretation

Assume a set of *agents* $\mathcal{A}$. Each agent $a \in \mathcal{A}$ is defined as a pair $a =< W_a, I_a >$ where W_a is the agent's world model consisting of a set of facts $W_a = \{f_1, ..., f_n\}$ and I_a is the agent's language inventory, whose structure is defined later. Agents take turns being speaker and hearer and we will assume that they use the same interventory both for coding and decoding. Coding and decoding is in the service of a more encompassing process: producing and understanding. Language speakers are not just uttering sentences without any purpose. They do it because they want to achieve an effect in the hearer. Although there are many possible effects, reflected in the type of speech act implied by the sentence, we will here focus on just one very common communicative goal: The speaker draws attention of the hearer to an object or event in the shared world situation.

After having chosen a topic T_s, the speaker must first conceptualise what meaning M he is going to use to draw attention to T_s. Conceptualisation is a complex cognitive process and appears to be to some degree language-dependent (Talmy, 2000). Here I will just assume that the topic is one of the objects in the speaker's world model W_s and that conceptualisation selects a subset of the facts in W_s: $conceptualise(T_s, W_s) = M_s \subset W_s$. M_s should be such that it uniquely circumscribes the intended topic, which will be the case if the constellation of predicates used in M_s is true for the topic but not for any other object in the world model. Given M_s, the speaker then uses the coding function to produce the utterance: $code(M_s, I_s) = u$

The hearer now uses his own inventory to decode the meaning of the utterance: $decode(u, I_h) = M_h$. Usually it is assumed that $M_s = M_h$, however that is too simplistic. What the hearer obtains from decoding u is an expression with the same predicates as M_s but with variables instead of objects for the arguments (assuming the simplifying case where $I_s = I_h$). The hearer next needs to perform semantic interpretation, which is the process whereby the variables in M_h are assigned values by matching M_h against the world model W_h. The topic intended by the speaker can then be retrieved, thus completing semantic interpretation: $interpret(M_h, W_h) = T_h$.

A simple example will make the need for this extra step clearer. Consider the noun phrase "the red ball" which refers (draws attention) to an object, $o1$. The speaker's conceptualisation has selected two facts about $o1$: $red(o1)$ and $ball(o1)$, and we will write M_s as $[o1|red(o1), ball(o1)]$, to mean 'the object $o1$ such that the two predicates red and $ball$ hold'. When the hearer decodes "the red ball", he obviously does not know yet which object is intended. He is only told that there is something which is red, that this thing is a ball, and that this is what the speaker wants to draw attention to. Formally, M_h is therefore equal to M_s with variables: $[X_1|red(X_1), ball(X_1)]$[1]. The hearer's semantic interpretation process must then match this expression against the hearer's world model and finds that the variable X_1 is bound to $o1$.

Communicative success occurs when the topic identified by the hearer is unique and the same entity in the real world as the topic originally chosen by the speaker:

$$\begin{aligned} & conceptualise(T_s, W_s) = M_s \\ & code(M_s, I_s)) = u \\ & decode(u, I_h) = M_h \\ & interpret(M_h, W_h) = T_h \\ & T_s = T_h \end{aligned}$$

Although language learners have been argued to receive no or little direct feedback on the nature of the language inventory, they obviously receive plenty of pragmatic feedback on whether the communication was a success or a failure. For example, if you sit at a table and ask for the plate with salmon by saying "the salmon, please", the success of communication is simply reflected by whether you get the salmon or not.

The problem of semantic interpretation is an instance of a so called constraint satisfaction problem (CSP) which has been widely studied in computer science. Each predicate in M_h can be seen as a constraint on its arguments. The domain of possible values is equal to the entities in the world model. A predicate $p_i(X_1, X_n)$ is satisfied for a particular assignment iff the fact obtained by instantiating the variables is part of the given world model. For example, $p_i(X_1, X_2)$ is satisfied for $\{X_1 = o1, X_2 = o1\}$ iff $p_i(o1, o1)$ is an element of W_h. A possible interpretation of M_h is equal to a complete assignment where all variables in M_h are bound in a way that satisfies all the constraints.

The computational complexity of CSP has been thoroughly studied and this allows us to define the computational complexity of interpreting a meaning structure M_h with respect to a world model W_h. Concretely, we are dealing here with a discrete CSP and assume (simplifying) that the number of possible objects in the world model is finite, hence the set of possible assignments of variables d is finite as well. The maximum number of possible assignments for a given meaning M_h with m variables is therefore $O(d^m)$. Searching through this set to find the assignment(s) that are compatible with W_h is exponential in the number of variables.

The following example makes this more concrete. Suppose that the hearer's world model W_h contains the facts:

> ball(o1), ball(o2), hit(o1, o3), hit(o2,o4), box(o3), box(o4), nextto(o3,o6), nextto(o4,o7), green(o6), green(o2), cube(o6), cube(o7), blue(o5), blue(o7)

---

[1] Variables start with an upper case letter and values with a lower case one.

and that he hears the utterance: "The ball that hit the box next to the green cube". Suppose furthermore that the hearer has a lexicon that maps the content words in this phrase to the corresponding predicates. For example, "ball" adds $ball(X_1)$ to M_h, "hit" adds $hit(X_2, X_3)$ to M_h, etc., so that the phrase is decoded as:

> [X_1 | ball(X_1), hit(X_2,X_3), box(X_5), nextto(X_6,X_7), green(X_8), cube(X_4)]

There are 7 objects in W_h, and 8 variables in M_h, which makes the set of possible assignments equal to $7^8 = 5764801$, a very large number. Many language sentences feature a much larger set of words and involve situations that involve a lot more than 7 objects. So unless a more intelligent method is found for semantic interpretation, communication is not viable.

A first obvious step is to choose an algorithm that does not search by enumerating the set of possible assignments for each of the variables but starts from the predicates in M_h and enumerates only those assignments that actually occur in the world model W_h for each predicate. The computational complexity of semantic interpretation can then be defined in terms of the number of facts in which the same predicate occurs. Let k be the maximum number of facts in the world model that use the same predicate, then the computational complexity of semantic interpretation is $O(k^m)$. This is still exponential in the number of variables, but, assuming a relatively small size of the world model, will be a much smaller number. Concretely, for the example world model given earlier, k is only 2. (There are two boxes, two balls, two hit events, etc.) And so we get $2^8 = 256$ possibilities for M_h. However for realistic world models this is again going to become very large.

3 The role of grammar

The computational complexity of semantic interpretation can be reduced further either (1) by reducing the number of variables in M_h, or (2) by shrinking the set of objects and facts in the world model, which reduces k. Human language users use quite a few devices (linguistic and extra-linguistic) to restrict the context of a conversation and this reduces the domain of the variables and the maximum number of facts that have the same predicate, but I will not elaborate on that aspect here. Instead I focus on the first question, namely how can speakers and hearers reduce the number of variables in the decoded meaning structure? This is precisely where grammar becomes essential.

The key point of this paper is that *the first purpose of grammar is to reduce the number of variables in a decoded meaning structure and hence reduce the computational complexity of its interpretation*. Going back to the example phrase "The ball that hit the box next to the green cube", we see that there is a lot of additional information in this phrase, beyond the lexicon, that communicates equalities between some of the variables:

- "Green cube" forms a noun phrase so that the hearer knows that the predicate green applies to the same object as the predicate cube, $X_8 = X_4$.

- "The ball hit the box ... " is a verb phrase with "the ball" in subject and "the box" in direct object position. This indicates the roles referents play in the hit-event, leading to the conclusion that $X_1 = X_2$, $X_7 = X_8 = X_4$ and $X_6 = X_5 = X3$.

So we have a reduction from 8 to 3 variables and computational complexity of semantic interpretation reduces from *O*(256) to *O*(8). Variables which are constrained to refer to the same object are called equalities.

The issue is not only complexity. Without the additional information that some of the variables introduced by the lexicon have to be assigned to the same values, there would be several semantic interpretations which are all complete. Going back to the example phrase, we see that, there are in fact 2^8 of them (because I constructed the example so that there are two possible assignments for each predicate). However, when taking the additional constraints on variable equalities communicated by syntax into account, only one interpretation remains. So the secondary effect or grammar is also to reduce the number of possible interpretations so that only a unique complete assignment of the variables remains.

4 From Syntax to Grammar

The next question is how natural languages communicate variable equalities. One way is through syntactic structures, based on word order or extra markings. For example, combining the words "red" and "ball" into "red ball" implies that the variables used in $red(X_1)$ and $ball(X_2)$ are equal, $X_1 = X_2$, so that the meaning becomes $[X|red(X), ball(X)]$. Such a patterning could at first be completely ad hoc, which is the case for example in programming languages. To specify the arguments of a procedure or function, programming languages or logic use ordering. For example, the procedure $DrawWindow(W, x, y, z)$, requires 4 arguments to be supplied in a particular order. Note that ad hoc syntactic structures could already have recursive structure, if a group which forms a unit (like “red ball”) is combined into a larger structure (“red ball next to green ball”). In a programming language, there is no further systematicity in syntax. When defining another procedure like MoveWindow, there could be a totally different ordering: $MoveWindow(x2, y2, z2, W)$ or $move - window(W, z2, y2, x2)$, etc., depending on the programmer’s wish. Of course a good programmer will introduce some systematicity in the syntax he is using but the interpreter and compiler know nothing about this.

An experiment in the emergence of syntax in this sense has been carried out by Batali (2002). His syntactic combination rules contain ‘argument maps’ to specify the variable equalities. They are created in an ad hoc fashion as exemplars. Thus, using numbers for the arguments, the individual words usifala, [(snake 1)(sang 1)] and ozoj [(chased 1 2)], are combined into “usifala ozoj” to express (snake 1) (sang 1) (chased 1 2), with the mapping 1:1 for the first word, and 1:1, 2:2, for the second one. Agents negotiate the use of exemplars, based on a lateral inhibition dynamics: Success reinforces the use of certain exemplars and failures discourages their use. Exemplars are reused as much as possible which implicitly creates at least some systematicity but this systematicity is not captured in rules.

Natural languages however impose an additional layer in between the meaning to be conveyed and the final syntactic form. The meaning is reconceptualised in terms of semantic frames such as a TRANSFER-TO-TARGET frame with agent, target and patient and the form is categorised in terms of syntactic categories (like noun, article, etc.), grammatical relations (like subject, determiner), and syntactic patterns (like a Subject-Verb-Direct-Object pattern). The combination of a semantic frame and a syntactic pattern is known as a grammatical construction (see figure 1) (Goldberg,1995). It is only when such a layer of grammatical constructions with syntactic and semantic categories that one can speak about true grammar. It has the obvious advantage of economy and greater expressive power. Constructions in natural language clearly have different degrees of specifity (i.e. idiomaticity), ranging from very idiomatic constructions built around a particular noun or verb, to very general constructions with wide applicability, such as Subject+Predicate+DirectObject (as in ”John gives a book”). Constructions thus form networks where more specific constructions inherit from more general ones and combine with each other to achieve high expressive power. Moreover empirical observations of actual language use shows that the inventory of constructions used by an individual (including adults) is constantly changing. Constructions capture conventionalised patterns of usage, but new patterns develop all the time and others may go out of fashion.

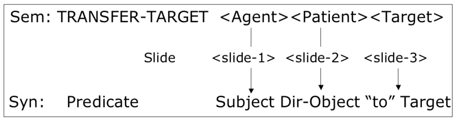

Figure 1: A construction relates a syntactic pattern such as Subject+Predicate+DirectObject+PrepObject with a semantic frame such as TRANSFER-TO-TARGET+Agent+Patient+Target.

To implement all this, we need a formalism which can explicate semantic categorisation rules for reconceptualising meanings into semantic frames, and syntactic categorisation rules that categorise words and syntactic structures. There must also be an explicit representation of grammatical constructions, i.e. associations between semantic frames and syntactic patterns. These constructions should still establish equalities between variables (as Batali’s argument maps) but they will now be more generic and hence applicable to a wide range of situations. If syntactic and semantic categorisations and constructions are explicitly represented in the grammar, then it follows that the agents must have operators for inventing them (as speaker when there are equalities that need to be eliminated), for adopting them (as hearers when there are equalities that the speaker has eliminated) and for aligning them to ensure that the categories and rules of different agents become similar. The next section provides a bit more technical detail on how we have implemented these various aspects.

5 Fluid Construction Grammars

The formalism we have implemented for representing emergent grammars is called Fluid Construction Grammar (FCG) and is related to other computational implementations of construction grammar such as ECG (Bergen and Chang,2003), as well as standard techniques of unification-based grammar employed in computational linguistics today (Pollard and Sag, 1994). Specifically, syntactic and semantic structures are represented as typed feature structures, as shown in figure 2 and 3 for the sentence “Jill slides Jack the block”. Fluidity refers to the goal of being extremely flexible in parsing and production, including when there is no or insufficient grammar or when some rules are violated. In FCG, all rules are bi-directional so that they can be used both for producition (i.e. constructing an utterance that expresses specific meanings derived through a conceptualisation process from a grounded world model) and for parsing (reconstructing the meaning of an utterance and mapping it back into reality by way of the grounded world model). This is a tough technical requirement which is achieved by viewing grammar rules as constraints and language processing as constraint propagation.

```
unit1
 syn-cat: {sentence,SVOtoO-sentence}
 syn-subunits: {unit2,unit3,unit4,unit6}
unit2
 syn-cat: {person(3d,natural),number(singular,natural),
          subject(unit1,unit2)}
 form: {stem(unit2,"jill")}
unit3
 syn-cat: {predicate(unit1,unit3)}
 form: {stem(unit3,"slide")}
unit4
 syn-cat: {person(3d,natural),number(plural,natural),
          direct-object(unit1,unit3)}
 form: {stem(unit5,"block")}
unit6
 syn-cat: {person(3d,natural),number(singular,natural)
          prep-object(unit1,unit6)}
 form: {stem(unit4,"jack")}
```

Figure 2: Syntactic structure after application of the TRANSFER-TO-TARGET construction. There is a unit for each word and for combinations of words. The syntactic categories as well as the properties of the surface form are represented as predicates over units.

```
unit1
 sem-subunits: {unit2,unit3,unit4,unit6}
unit2
  referent: {obj1}
  meaning {jack(obj1),status(obj1,single-object),
           discourse-role(obj1,external)}
unit3
  referent: {ev1}
  meaning: {slide(ev1,true),slide-1(ev1,obj1),
            slide-2(ev1,obj2),slide-3(ev1,obj3)}
  sem-cat: {transfer-to-target(ev1),agent(ev1,obj1),
            patient(ev1,obj2),target(ev1,obj3)}
unit4
  referent: {obj2}
  meaning: {block(obj2),status(obj2,single-object),
            discourse-role(obj2,external)}
unit6
  referent: {obj3}
  meaning: {jill(obj3),status(obj3,single-object),
            discourse-role(obj3,external)}
```

Figure 3: Semantic structure built up alongside the syntactic structure shown in the previous figure. It contains bits of meaning as well as semantic categorisations necessary for the application of the grammatical construction (in the slot SEM-CAT).

FCG rules contain a left pole and a right pole and are activated and applied through unification. An example of a grammatical construction is shown in figure 4. The left pole constrains the semantic side and the right pole the syntactic side. Other rules will expand the semantic and the syntactic structure with descriptions so that this rule can be applied. For example, there will be a semantic categorisation rule that re-conceptualises a slide-event with its various roles (as in John slides the book to Mary) into a TRANSFER-TO-TARGET event. Producing and parsing are totally analogous, the only thing which changes is the direction of rule application.

```
def-cons transfer-to-target-construction
  ?top-unit
    sem-subunits:
      ?event-unit,?agent-unit,?target-unit,?patient-unit
  ?event-unit
    referent: ?event
    sem-cat: transfer-to-target(?event),agent(?event,?agent)
       patient(?event,?patient),target(?event,?recipient)
   ?agent-unit
     referent: ?agent
   ?patient-unit
     referent: ?patient
   ?target-unit
     referent: ?recipient
  <-->
   ?top-unit
     syn-cat: SVOtoO-sentence
     syn-subunits:
      ?event-unit,?agent-unit,?patient-unit,?target-unit
    ?event-unit
      syn-cat: predicate(?top-unit,?event-unit)
    ?agent-unit
      syn-cat: subject(?top-unit,?agent-unit)
    ?patient-unit
      syn-cat: direct-object(?top-unit,?patient-unit)
    ?target-unit
      syn-cat: prep-object(?top-unit,?target-unit)
```

Figure 4: Example of a construction which relates a TRANSFER-TO-TARGET frame to a Subject+Verb+Direct-Object+to+Prep-Object pattern

Agents in our simulations of grammar emergence create categories and constructions in order to reduce the computational complexity of semantic interpretation and align these categories and constructions based on the outcome of the language game. We summarise the main principles of these simulations and refer to Steels (2005) for more detail.

Suppose the speaker has a target meaning M_s which he wants to communicate to refer to a topic, and he can use his lexicon (and maybe already a partial grammar) to code that meaning into an utterance u. But before sending u to the hearer, the speaker can first determine the complexity of semantic interpretation by *re-entrance*: The speaker decodes u (using his own lexicon and grammar) to yield a meaning M_s', and then tries to interpret M_s' against his own world model W_h. This gives a set of possible bindings and possibly a set of equalities. If there are equalities, the speaker knows that additional grammar should be added. Conversely, if the hearer attempts to interpret his interpretation of an utterance u and obtains a possible referent T_h (possibly after additional interaction with the speaker if there was a failure), then he also has a set of bindings and a set of equalities. If there are equalities, the hearer can interpret the additional syntactic information present in the utterance as a reasonable hypothesis that this information is intended to show how the equalities can be resolved.

We discuss first an example how a specific idiomatic construction is generated. Suppose that the speaker wants to express the following fall event: ‘fall(ev1), fall-1(ev1,obj1), ball(obj1)’. Assume that the speaker has already lexical rules for “fall” and “ball”, leading to the semantic and syntactic structure in figure 5. No grammar is involved yet.

```
unit1
 sem-subunits: {unit2,unit3}
unit2
  referent: {obj1}
  meaning: {ball(obj1)}
unit3
  referent: {ev1}
  meaning: {fall(ev1),fall(ev1,obj1)}
```

```
unit1
 syn-subunits: {unit2,unit3}
unit2
  form: string(unit2,“ball”)}
unit3
  form: string(unit3,”fall”)}
```

Figure 5: Semantic (left) and syntactic (right) structure after applying lexical rules for “ball fall”.

If the speaker re-interprets himself the resulting sentence “ball fall” using his own lexicon, he comes up with the following meaning: ‘fall(?ev1), fall-1(?ev1, ?obj1), ball(?obj2)’. If this is matched against the original meaning, the equality ?obj1 = ?obj2 becomes apparent. So if this equality would become expressed grammatically, the communication would become more precise and the risk of failure decreases. The speaker invents a construction for this purpose in two steps.

The first step is to combine the structures derived from the lexicon, introduce variables for all units and entities involved, and add the precedence relation occurring in the sentence, which was arbitrary but now becomes rule-governed. Slots need to contain the specified elements but also could contain other ones. This gives the result shown in figure 6. Note that the variable used with the predicate ball (i.e. ?obj1) is the same as in fall-1. This is the way that the equality will get established when the rule is applied.

This construction does the job in the sense that when “ball fall” is seen, the lexicon contributes the various predicates to the meaning and the construction establishes the right equality. However it is completely ad hoc, so a more general operation should take place, which generalises the meaning and the form by stating the constraints in terms of semantic and syntactic categorisations. The result is shown in figure 7.

The relation between the semantic categorisations and the meaning predicates now needs to be translated into a sem-rules (shown in figure 8). These rules are easily constructed by taking the relevant part of the meaning slot in the original semantic structure and linking it to the sem-cat slot in the construction. These categories are still ad hoc in the sense that they have only one member, but the categories progressively become richer as new elements are declared to be members of them, so that the extent of sem-cat1 becomes something like ‘the set of objects which can participate in physical movement events’, sem-cat2 becomes ‘the set of events that involve such physical movement’, and sem-cat3 ‘the patient involved in this physical movement’.

```
def-con ball-fall
 ?unit1
   sem-subunits: ?unit2,?unit3
 ?unit2
   referent: ?obj1
   meaning: ball(?obj1)
 ?unit3
   referent: ?ev1
   meaning: fall(?ev1),fall-1(?ev1,?obj1)
<-->
 ?unit1
  syn-subunits: ?unit2,?unit3
  form: precedes(?unit2,?unit3)
 ?unit2
   form: string(?unit2,“ball”)
 ?unit3
   form: string(?unit3,”fall”)
```

Figure 6: The first step in inventing a construction is to perform a kind of lambda-abstraction, introducing variables for units and entities.

```
def-con object-move
 ?unit1
   sem-subunits: ?unit2,?unit3
 ?unit2
   referent: ?obj1
   sem-cat: sem-cat1(?obj1)
 ?unit3
   referent: ?ev1
   sem-cat: sem-cat2(?ev1),sem-cat3(?ev1,?obj1)
<-->
 ?unit1
  syn-subunits: ?unit2,?unit3
  form: precedes(?unit2,?unit3)
 ?unit2
   syn-cat: syn-cat1(?unit2)
 ?unit3
   syn-cat: syn-cat2(?unit3)
```

Figure 7: The second step in inventing a construction is to replace specific meaning and form predicates with semantic and syntactic categorisations.

```
def-sem sem-cat2+3
 ?unit
   referent: ?ev1
   meaning: fall(?ev1),fall-1(?ev1,?obj1)
 -->
 ?unit
   sem-cat: sem-cat2(?ev1),sem-cat3(?ev1,?obj1)
```

Figure 8: A semantic categorisation rule for some of the semantic categories in the construction shown in figure 7

The relation between the syntactic categories and the predicates describing aspects of form is expressed in syn-rules. An example is shown in figure 9. The form constraint is repeated in the left pole to make the rule bi-directional. Again these categories are at this moment ad hoc, having the specific words “ball” or “fall” as only members, but as the construction gets re-used, the category becomes broader and they will become similar to the parts of speech in natural languages.

```
def-morph “ball”
  ?unit
    syn-cat: syn-cat1
    form: string(?unit, “ball”)
 <-->
  ?unit
    form: string(?unit, “ball”)
```

Figure 9: A syntactic categorisation rule for “fall”

The hearer goes through exactly the same sort of operations for constructing his own grammatical rules. The hearer detects equalities based on the predicates supplied by the lexicon which are matched against the specific situation in the shared environment to yield a set of bindings and possibly equalities. Every rule in FCG has a strength which reflects how much success the rule has had in the past. The strength is updated using lateral inhibition dynamics, already used for the lexicon Steels(1996): Successful application reinforces the rule and failure causes damping. This leads to a gradual self-organised coherence of the agents’ repertoires.

Language users should try to optimise their inventories by re-using as much as possible existing constructions to cover new situations. This has two advantages: economy of memory, because fewer rules need to be stored, and optimisation of processing because fewer rules need to be considered. But re-use is also beneficial to speed up learning. If there is already a construction which is more or less doing the job, then the hearer can use that construction as a basis to help guess the meaning and learn more about the syntactic and semantic categories of the speaker. In line with the (embodied) cognitive linguistics approach, we argue that grounding should play a major role in deciding to re-use a construction.

Some cases are relatively straightforward. If there is another fall event but now involving another object, say a block, then “block” can simply be categorised as syn-cat1 and the predicate block as sem-cat1, so that the OBJECT-MOVE construction shown in figure 7 becomes applicable. However other cases are not so straightforward. Suppose that a new event has to be categorised (e.g. ‘slide(?ev5), slide-1(?ev5,?obj6), slide-2(?ev5,?obj7)’). The already existing instances of a category (in the example above this is so far only the fall-event with sem-cat2 and sem-cat3) can be compared to the new event by examining the state transition networks that are used for the recognition of each event. The primitive events and event combinations of each event are paired together with the entities that play specific roles in each event. Based on this comparison, a measure of category membership can be computed in a straightforward manner, to find the category whose instances are closest to the new event to be expressed. Thus, besides a ’patient’ that is undergoing movement (sem-cat3), an agent is involved in a slide-event. The entity playing this role participates in different primitive events than the patient and hence the corresponding predicate would not fit very well with sem-cat3. There are still other ways in which constructions can be re-used. For example, if there is already a construction like the one shown in figure 7 it could be specialised with additional roles, e.g. to express the agent of the move-event or the manner of movement.

These learning mechanism proposed are ’constructivist’ Tomasello and Brooks(1999) in the sense that they are not derived from statistical clustering but imposed by language users and then possibly adopted as consensus. At first the categories are ad hoc and have only a single entity as its member, but as constructions are re-used, more instances are added to the category and so they are getting a richer content. The instance-based learning of categorisation results in the prototype behavior also seen with the linguistic categories found in human natural languages.

6 Conclusions

This paper argued that reducing the computational complexity of semantic interpretation, and hence the chance of communicative success, can be the main driving force for getting a population of agents to develop grammar. It argued also that ’true’ grammar only arises when syntactic and semantic categories are used and grammatical constructions to have a more abstract mapping between form and meaning. We do definitely not argue that this is the only use of grammar. In fact when second order predicates become used (i.e. predicates that have other predicates as argument, such as ”very” in ”very good”) there is a second important reason for introducing grammar, namely that the grammar specifies how a predicate needs to be used. Much further work needs to be done to carry the computational simulations forward, and there is no doubt that the operators we have used so far need to be extended with more powerful mechanisms for the invention of new grammar. Chang and colleagues (Chang and Maia,2003) have recently presented computer simulations of such learning processes based on empirical data of child language acquisition and Bayesian learning mechanisms. The perspective adopted here is along similar lines, although we use an abductive learning approach.

Research funded by Sony CSL with additional FWO funding in the ESF-OMLL program, and the EU FET-ECAgents project. The author is indebted to Joachim de Beule, Nicolas Neubauer, and Joris Van Looveren, who have made important contributions, particularly to implementation issues.

References

[1] Batali, J. (2002) The negotiation and acquisition of recursive grammars as a result of competition among exemplars. In (3).

[2] Bergen, B.K. and N.C. Chang. Embodied Construction Grammar in Simulation-Based Language Understanding. Technical Report 02-004, International Computer Science Institute, Berkeley.

[3] Briscoe, T. (ed.) Linguistic Evolution through Language Acquisition: Formal and Computational Models. Cambridge University Press, Cambridge, UK.

[4] Cangelosi, A. and D. Parisi (eds.) (2001) Simulating the Evolution of Language. Springer-Verlag, Berlin.

[5] Chang, N. C. and Maia, T. V. (2001) Grounded Learning of Grammatical Constructions. In 2001 AAAI Spring Symposium on Learning Grounded Representations.

[6] Goldberg, A.E. (1995) Constructions: A construction grammar approach to argument structure. University of Chicago Press, Chicago.

[7] Hashimoto, T. and T. Ikegami (1995) Emergence of netgrammar in communicating agents. Biosystems 38, 1-14.

[8] Nowak, M.A. and D.C. Krakauer (1999) The evolution of language. Proc.Natl.Acad.Scie. USA. Vol 96, pp. 8028-8033, July 1999.

[9] Pollard, C. and I. Sag 1994. *Head-driven phrase structure grammar.* Center for the Study of Language and Information of Stanford University. University of Chicago Press, Chicago.

[10] Smith, K., Kirby, S., and Brighton, H. (2003). Iterated Learning: a framework for the emergence of language. Artificial Life, 9(4):371–386.

[11] Steels, L. (1996) Emergent Adaptive Lexicons. In: Maes, P. (ed.) (1996) *From Animals to Animats 4: Proceedings of the Fourth International Conference On Simulation of Adaptive Behavior*, The MIT Press, Cambridge Ma.

[12] Steels, L. (1998) The origins of syntax in visually grounded robotic agents. Artificial Intelligence, 103:1–24.

[13] Steels, L. (2003) Evolving grounded communication for robots. Trends in Cognitive Science. Volume 7, Issue 7, July 2003 , pp. 308-312.

[14] Steels, L. (2005) The role of Construction Grammar in Language Grounding. Artificial Intelligence Journal, Vol 31.

[15] Talmy, L. (2000) Toward a Cognitive Semantics: Concept Structuring Systems (Language, Speech, and Communication) The MIT Press, Cambridge Ma.

[16] Tomasello, M. and P.J. Brooks 1999. *Early syntactic development: A Construction Grammar approach* In: Barrett, M. (ed.) (1999) *The Development of Language* Psychology Press, London. pp. 161-190.