Disambiguating Stickers
Grammatical Agreement as a Design Pattern

Katrien Beuls [a]

[a] *Vrije Universiteit Brussel, Pleinlaan 2, B-1050 Brussels*

Abstract

Many of the world's languages exhibit grammatical agreement, meaning that at least two words in a sentence match in some way. For example, the words in the Spanish utterance "las casas altas" (the tall houses) all share the same ending "-as", which indicates the grammatical gender and number features of the houses the speaker is referring to. In this paper, I will address the function of what seems to be redundant marking by presenting the results of multi-agent language games. Faced with the ecological need to reduce parsing effort, agents are forced to group words that belong together. This is done through morphological markers such as "-as" which unlike their Spanish counterparts do not refer to any grammatical feature but only impose a linguistic sticker onto words that form a group. I will demonstrate how such stickers can propagate in a population of software agents based on their token frequency. The interactive web demonstration that accompanies this paper can be found on: `http://arti.vub.ac.be/~katrien/disambiguating-stickers/index.xhtml`.

1 Introduction

Grammatical agreement is present in many of the world's languages today and has become an essential feature that guides linguistic processing. When two words in a sentence are said to "agree", this means that they share certain features such as *gender*, *number*, *person* or others. The primary hypothesis of this paper is that marking agreement within one linguistic phrase reduces processing effort as word groups can more easily be recognized.

Imagine you would hear a sentence such as "big chair ball red black". You would not immediately know how the different words are related unless they would be uttered in a context where you see a big black ball and a red chair. Natural language usually employs syntax in order to avoid redundant ambiguities so that communication is streamlined. There are commonly three strategies found in languages around the world that support this solution: (i) word order, (ii) prosody and (iii) morphological markers [4]. Often, combinations of these strategies are applied. In French for example, a word group follows the default nominal word order schema [determiner + noun + adjective] and is marked on every word for number (and gender when singular), e.g. *les ballons rouges*, "the red balls".

This paper presents the results of evolutionary simulations that investigate the third strategy. The simulations rely on the *language game* model, as first proposed by Steels [6], that has been used extensively for studying the dynamics of linguistic interactions between agents, be it humans or artificial agents [12, 14] and their collective learning process [10]. The series of games that is presented here has been named *the sticker games*.

The sticker games examine the most 'naive' morphological marking strategy, namely by means of mere formal markers whose exclusive function is to signal the word group members, without including references to any semantic or formal features that characterize the group. Such markers are comparable to colored stickers that one attaches to books that fall into the same category or to uniforms people wear to distinguish themselves from others. Since the experiments investigate linguistic communication systems, stickers take the form of labeled markers such as *-x*, *-y*, *z*, etc. The general schema that is imposed on multiple words belonging to the same group, therefore referring to the same referent in the meaning space, looks as follows:

$$[word1_x word2_x \dots wordn_x] \quad (1)$$

The paper is structured as follows. Section 2 explains the basic architecture of the sticker game. It provides details on the situation generator and the problem solving module that monitors the regular game flow. Section 3 shows the results of the introduction of formal stickers on the population level as well as the individual agent level. Section 4 discusses these results and asks the question why natural languages do not display such a sticker marking behavior. Virtually all natural languages that employ the agreement strategy use markers with a semantic underpinning of some kind, some more vaguely than others. Section 5 concludes the paper.

2 The Sticker Game

Every language game follows a routinized interaction script that is designed by the experimenter of the game. The interaction script for the sticker games is included in Figure 1. At the start of one game, a speaker and a listener are randomly selected from the population and a new situation (or scene) is generated automatically. The speaker's actions are represented by the rectangles, the listener's by the oval shaped boxes. The two dark shaded areas designate the meta-level actions, that is the actions that monitor the course of the game. Section 2.2 discusses these actions in further detail. The situation generator is explained in Section 2.1. At the end of a game, the listener increases the frequency scores of all used markers (e.g. *-x*). This score is used in production when the decision is made which marker to use to mark a word group. When *-x* has been used more often in past games than *-y*, the speaker will prefer *-x* over *-y* to mark a new word group.

Before the start of the first game, the agents are initialized with a English-like lexicon (covering all possible meaning predicates that can be generated), together with a basic grammatical architecture to link words in the lexicon into word groups. In a script without the meta-level actions, all agents would be able to communicate with each other with utterances that lack markers, although the communication cost would be high in terms of cognitive effort (see Section 2.2). The grammatical component that supports the agents' production and parsing processes has been implemented in Fluid Construction Grammar (FCG) [9]. More details on the grammatical implementation are explained in the Appendix as well as the interactive web demonstration that accompanies this paper: `http://arti.vub.ac.be/~katrien/disambiguating-stickers/index.xhtml`.

The sticker game agents are all software agents that are built inside the Babel2 environment [5].

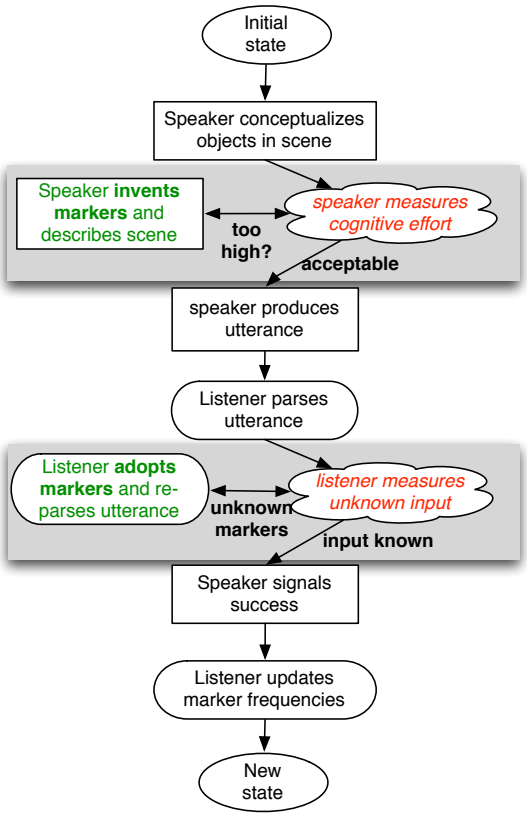


Figure 1: The sticker game's interaction script

2.1 Situation Generator

The situation S contains a subset of the world W, represented as a set of predicates and arguments in prefix notation. For example, the following situation involves two objects (labeled `ref-1` and `ref-2`) which each have a series of properties, such as big, ball, chair, etc.

```
(big ref-1) (ball ref-1) (red ref-1)
(chair ref-2) (black ref-2)
```

These situations are automatically generated by a script but they could be derived from real world situations. For the sticker games, there is no uncertainty in perception, i.e. both agents are assumed to have perceived the situation in exactly the same way. Moreover the situation always contains at least two objects and at most five. Also the number of properties ranges from two to five.

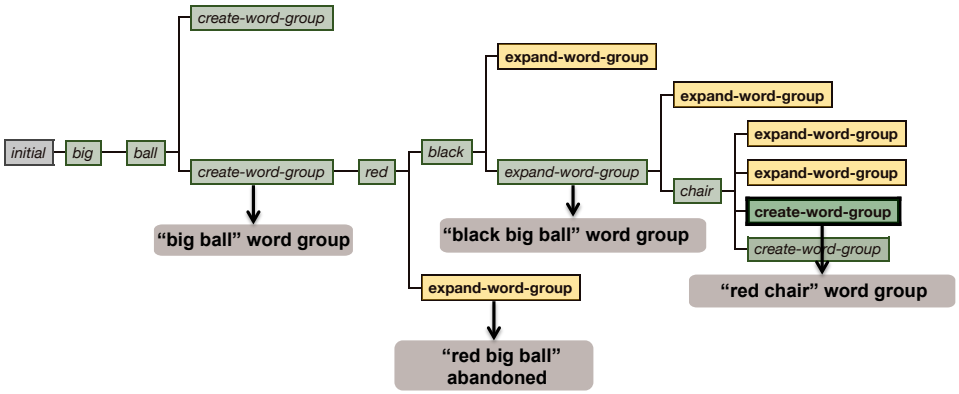


Figure 2: Annotated screen shot of a parse tree when parsing the utterance "black big ball red chair". The explanatory panels provide information on the creation of word groups and the failed search nodes. In these nodes the current situation does not match the meaning parsed so far. The cognitive effort (number of failed nodes) would be five in this parse tree.

2.2 Monitoring the Game Flow

As Figure 1 has already shown, there are two levels in the interaction script: a level with basic actions related to the problem domain (i.e. to describe objects in the situation) and a second level with reflective actions that monitor the flow of the game's basic level. The double layered architecture is an inherent feature of Babel2 [3]. For the speaker, the following meta operators have been implemented:

- `Detect-cognitive-effort`: Before the speaker sends his utterance to the listener, he verifies the degree of cognitive effort the listener would experience when he parses this utterance, taking himself as a model. This process is called re-entrance [7]. When the cognitive effort is too high, the speaker decides to invent formal stickers in order to reduce it and then goes through the production process again. Cognitive effort is calculated as the number of failed search nodes in the parse tree. Nodes fail when their parsed meaning does not match the current situation. Figure 2 explains this in further detail. Figure 3 shows the population's cognitive effort in a series of baseline games that lack the meta-level operator that detects cognitive effort.

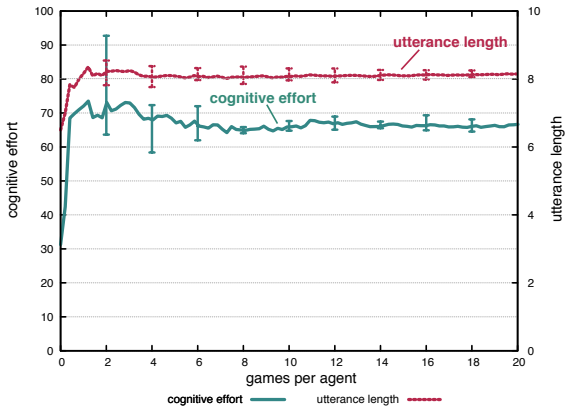
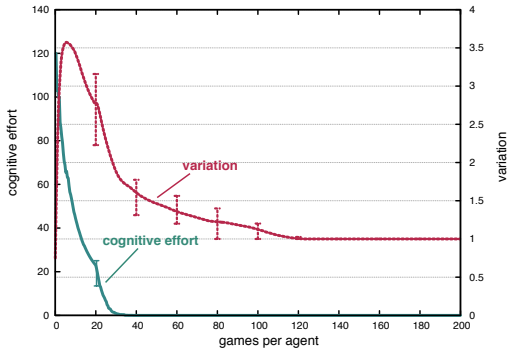
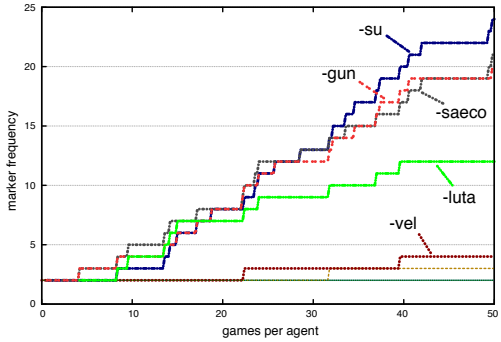


Figure 3: The baseline cognitive effort (y1 axis) plotted in function of the number of games per agent (x axis). Without introducing sticker markers, the cognitive effort (number of failed nodes) remains more or less stable (65), given an average utterance length of 8 words. Settings: 10 agents, 100 language games, 4 game series, maximum number of objects = 5.

- `Invent-formal-sticker`: When cognitive effort is diagnosed, the speaker contests this with the invention of formal sticker markers that group words that belong to the same individual by attaching them the same suffix. The speaker creates a new form by assembling a random combination of



(a) The cognitive effort (listener) and the variation in sticker usage is plotted in function of the number of games played per agent. Settings: 10 agents, 1000 language games, 4 game series, maximum number of objects = 5.



(b) The development of markers in the inventory of a specific agent. The agent uses five markers most frequently: -su, -saeco, -gun, -luta and -vel. The lower ranked markers only occur when enough objects appear in the utterance.

Figure 4: Comparing the results of adding formal stickers on the level of the population (a) and one individual agent (b).

consonants (C) and vowels (V) according to three possible patterns: CV, CVC or CVCV. When he then revises his earlier description he will now produce "big-xu chair-ba ball-xu black-ba" instead of the plain utterance "big chair call black". Note that the word order remains random.

The listener disposes of the following two complementary operators:

- `Detect-unknown-input`: Any input in the transmitted utterance that cannot be traced back to words that are part of the agent's grammar are caught. A problem is created in the meta level that contains all unknown input strings.
- `Adopt-formal-sticker`: The unknown input in the recently created problem is used now to create formal sticker markers that link objects together by means of suffixes. These markers are added to the grammatical inventory of the listener. They are formally the same as the one the speaker has added (see Appendix).

3 Results

Figure 4a shows the outcome of an experiment with 10 agents. The cognitive effort indeed drops dramatically as soon as agents start to use the formal sticker strategy. We are also interested to see whether agents reach a shared inventory, i.e. whether their preferences for the usage of stickers is converging so that the inventory is easier to learn for new incoming agents, less memory is needed to store them, and lookup is faster. We therefore track the variation in the population by measuring the average number of markers per feature. Variation is equal to $\frac{T_s}{M}$ where T_s is the total number of sticker markers in use in the population and M the maximum number of objects. If the variation is equal to 1, then there is only one marker per object and hence agents have reached total convergence. Figure 4a shows that this is also happening.

A snapshot of the top five highest frequency sticker constructions of two different agents in one experiment is shown in Table 1. The markers that belong to the top five are the same for both agents, but the frequency is different for constructions on rank three and four. In practice, this means that when agent 1 is describing a situation with three individuals he will use for the third individual `-gun`, whereas agent 2 would prefer `-luta`.

When we examine the marker frequencies of individual agents, a clear tendency becomes visible where the markers that are used more will rapidly increase even further. Thus Figure 4b shows the development of a single agent's markers in terms of their frequency scores within the first 50 games of the first game series. There are five markers that distinguish themselves from the rest. The difference in frequency between these five can be explained by looking at the size of the meaning spaces that are generated. These contain at least two and at most five objects, and thus require always two and occasionally five markers.

Table 1: Top five highest frequency sticker markers for two agents after 250 games (50 games/agent). Both agents share the same top five markers but they differ in the internal ranking of these markers.

	agent 1		**agent 2**	
rank	marker	freq.	marker	freq.
1	-su	24	-su	18
2	-saeco	21	-saeco	18
3	**-gun**	20	**-luta**	14
4	**-luta**	12	**-gun**	12
5	-vel	4	-vel	8

4 Discussion

In terms of evaluation, one immediately spots the drop down in parsing effort within the first few games an agent engages himself in. The marking strategy does seem to be an efficient means to dam in the number of parsing solutions to a minimum, resulting in a single solution that leads to the correct interpretation (see also [13]). There are however some issues to consider when interpreting this result.

First, the rate of convergence on a shared set of 5 stickers is rather slow. This is probably due to the fact that frequency and not communicative success is the measure that drives the spread of markers. At the end of each game, the listener increases the frequency of the used markers with 1. As a result of the absence of a semantic interpretation of a formal sticker (e.g. ?? $\Longleftrightarrow$ -*x*) , the lateral inhibition dynamics as normally used in lexicon formation cannot be employed here (see for instance [11]). In lateral inhibition, all markers would carry a score between 0 and 1 which would be increased if the marker was used and decreased if the marker was a competitor of a used one.

Second, the number of markers in use will rise linearly with the maximum number of objects in the meaning space. The convergence time in turn rises exponentially. As soon as the maximum number of objects in one situation expands, new markers need to be invented. Such an infinite marker inventory size is never encountered in natural languages, which are characterized by closed inflectional classes, only affected by morpho-phonological changes once established. Arbitrary marker invention might look efficient in the current experiments but will actually put a burden on the memory of an agent.

One can see several advantages for using meaningful stickers as a basis of an agreement system [2]:

1. If full fledged words are used as initial stickers, their meaning can be immediately inferred.
2. No additional stickers are needed for words acting as sources (such as words referring to female persons when feminine is chosen as agreement feature).
3. Human memory works best when items to be memorized have existing semantic associations. Meaningful stickers are therefore easier to remember compared to formal stickers.

Experiments on how such meaningful marker systems can arise in a multi-agent population have been reported in [1] and [2]. [1] also give an account of different feature selection methods, which become indispensable once semantic features need to be recruited to build meaningful stickers.

5 Conclusion

In this paper I have shown how a population of agents can impose a morphological grouping pattern onto single words in order to help the listener in disambiguating the relationships between words in an utterance. Words that belong to the same group are thereby marked with the same suffix. By doing so, these word groups show a preliminary type of agreement that still lacks the notion of shared semantic or formal features or internal hierarchy. The basic function of agreement can be seen as a design pattern that guides the listener in targeting the right parsing solution. However, the arbitrary markers increase the processing load in that any marker can be applied to group a randomly selected word group and there is no external motivation for selecting a marker other than its frequency score. This hints at the use of agreement features that are motivated either semantically or formally by the words that constitute a group. Preliminary experiments that go in this direction have been reported in [1] and [2]. Future work should further investigate the rise of semantic markers and the grammaticalization processes they undergo.

Acknowledgements

This research has been conducted at the VUB AI Lab in Brussels and has been funded by a strategic basic research grant from the agency for Innovation by Science and Technology (IWT). I would like to thank Luc Steels, director VUB AI Lab, for his great ideas, support and feedback. Any remaining errors or terminological obscurities are of course my own.

References

[1] Katrien Beuls and Sebastian Höfer. Simulating the emergence of grammatical agreement in multi-agent language games. In AAAI, editor, *Proceedings of the Twenty-second International Joint Conference on Artificial Intelligence*, 2011.

[2] Katrien Beuls, Luc Steels, and Sebastian Höfer. The emergence of internal agreement systems. In Luc Steels, editor, *Experiments in Language Evolution*. John Benjamins, Amsterdam, 2012.

[3] Katrien Beuls, Remi van Trijp, and Pieter Wellens. Diagnostics and repairs in fluid construction grammar. In *Language Grounding in Robots*. Springer, Berlin, 2012.

[4] Greville G. Corbett. *Agreement*. Cambridge, Cambridge Textbooks in Linguistics, 2006.

[5] Martin Loetzsch, Pieter Wellens, Joachim De Beule, Joris Bleys, and Remi van Trijp. The Babel2 manual. Technical Report AI-Memo 01-08, AI-Lab VUB, Brussels, 2008.

[6] Luc Steels. A self-organizing spatial vocabulary. *Artificial Life Journal*, 2(3):319–332, 1995.

[7] Luc Steels. Language re-entrance and the 'Inner Voice'. *Journal of Consciousness Studies*, 10(4-5):173–185, 2003.

[8] Luc Steels. A design pattern for phrasal constructions. In Luc Steels, editor, *Design Patterns in Fluid Construction Grammar*. John Benjamins, Amsterdam, 2011.

[9] Luc Steels, editor. *Design Patterns in Fluid Construction Grammar*. John Benjamins, 2011.

[10] Luc Steels and Frederic Kaplan. Collective learning and semiotic dynamics. In D. Floreano, J-D Nicoud, and F. Mondada, editors, *Advances in Artificial Life: 5th European Conference (ECAL 99)*, Lecture Notes in Artificial Intelligence 1674, pages 679–688, Berlin, 1999. Springer-Verlag.

[11] Luc Steels and Martin Loetzsch. The grounded naming game. In Luc Steels, editor, *Experiments in Cultural Language Evolution*. John Benjamins, Amsterdam, 2012.

[12] Luc Steels and Paul Vogt. Grounding adaptive language games in robotic agents. In I. Harvey and P. Husbands, editors, *Proceedings of the 4th European Conference on Artificial Life*, pages 474–482, Cambridge, MA, 1997. The MIT Press.

[13] Luc Steels and Pieter Wellens. How grammar emerges to dampen combinatorial search in parsing. In Paul Vogt, editor, *EELC 2: Symbol Grounding and Beyond*, volume 4211 of *LNAI*, pages 76–88, Berlin Heidelberg, 2006. Springer Verlag.

[14] Pieter Wellens, Martin Loetzsch, and Luc Steels. Flexible word meaning in embodied agents. *Connect. Sci*, 20:173–191, June 2008.

Appendix

The language systems discussed in this paper are all implemented in Fluid Construction Grammar (FCG) [9]. This appendix defines in more technical detail how the constructions involved the sticker games have been defined.

The agents' grammar is scaffolded with a lexicon and two phrasal constructions that link multiple words into a word-group.

Lexical constructions A lexical construction is a mapping between meaning and form. The meaning is a set of predicates (although in this experiment we use only one for simplicity) and the form is a string. The lexical construction also has to define what the meaning is about. Lexical constructions are defined using templates (see [9]), as in the following example:

```
(def-lex-cxn girl-cxn
  (def-lex-skeleton girl-cxn
      :meaning (== (girl ?x))
      :about (?x)
      :form "girl")
  (def-lex-cat girl-cxn
      :syn-cat (== word)))
```

This builds a construction called `girl-cxn`. It has a lexical skeleton which defines the meaning, the form, and what the meaning is about, and adds a syntactic category (which is very general, namely being a word) and a semantic-category.

Phrasal constructions The initial grammar contains two grammatical constructions. The first one, called `create-word-group` combines two words into a word-group. Which words can be combined is not constrained by their syntactic categorization (they just have to be words) nor their semantic categorization. However the two words have to be about the same referent. The construction is specified using the `def-phrasal-linking` template (discussed in [8]). Note that the word-group as a whole inherits the :about feature from the words.

```
(def-phrasal-cxn create-word-group
  (def-phrasal-skeleton create-word-group
     :phrase
      (?word-group
       :phrase-type word-group)
     :constituents
      ((?word-1
        :syn-cat (==1 word))
       (?word-2
        :syn-cat (==1 word))))
  (def-phrasal-linking create-word-group
     (?word-1
      :about (?referent))
     (?word-2
      :about (?referent))
     (?word-group
      :about (?referent))))
```

The second construction, called `expand-word-group` is entirely similar, except that a word is added to an existing word-group, if they are about the same object:

```
(def-phrasal-cxn expand-word-group
  (def-expand-phrasal expand-word-group
     :existing-phrase
        (?word-group
         :syn-cat (==1 word-group))
     :constituent
       (?word
        :syn-cat (==1 word)))
  (def-phrasal-linking expand-word-group
     (?word-group
```

```
      :about (?referent))
     (?word
      :about (?referent))))
```

Formal stickers are handled by a straightforward expansion of the two phrasal constructions used earlier. The suffixes are added by a new template `def-phrasal-morph` which attaches a suffix to a stem. Here is an example for a sticker "-bla". The sticker is stored in the unit for the word-group so that it can easily be reused when the word-group is expanded later with more words.

```
(def-phrasal-cxn create-word-group-bla
  (def-phrasal-skeleton create-word-group-bla
     :phrase
      (?word-group
       :syn-cat (==1 word-group)
       :sticker "-bla")
     :constituents
      ((?word-1
        :syn-cat (==1 word))
       (?word-2
        :syn-cat (==1 word))))
  (def-phrasal-linking create-word-group-bla
     (?word-group
      :about (?referent))
     (?word-1
      :about (?referent))
     (?word-2
      :about (?referent)))
  (def-phrasal-morph create-word-group-bla
     (?suffix-word-1
       :stem ?word-1
       :string "-bla")
     (?suffix-word-2
       :stem ?word-2
       :string "-bla")))
```

The expansion from `expand-word-group` to `expand-word-group-sticker` is straightforward because the `create-word-group` construction stores the sticker of the word-group at the level of the sticker:

```
(def-phrasal-cxn expand-word-group-sticker
  (def-expand-phrasal expand-word-group-sticker
     :existing-phrase
        (?word-group
        :syn-cat (==1 word-group)
:sticker ?sticker)
     :constituent
       (?word
        :syn-cat (==1 word)))
  (def-phrasal-linking expand-word-group-sticker
     (?word-group
      :about (?referent))
     (?word
      :about (?referent)))
  (def-phrasal-morph create-word-group-bla
     (?suffix-word
       :stem ?word
       :string ?sticker)))
```