Mingo, J.M., Aler, R. Evolution of shared grammars for describing simulated spatial scenes with grammatical evolution. Genet Program Evolvable Mach 19, 235–270 (2018).

DOI: 10.1007/s10710-017-9315-y

Noname manuscript No.
(will be inserted by the editor)

Evolution of Shared Grammars for Describing Simulated Spatial Scenes with Grammatical Evolution

Jack Mario Mingo · Ricardo Aler

Received: date / Accepted: date

Abstract One of the prevailing theories about language during the last decades considers human language as a behavior for communication purposes. From this point of view, language use plays a major role in the communication and language development process. On the other hand, if behavior and intention are associated, as it has also been proposed from several disciplines, language can be seen as a behavior aimed at communicating intentions. Following these theoretical foundations, we propose a model based on an evolutionary process combined with a planning process to develop a limited spatial language with a syntactical structure in a team of artificial agents. Syntax is induced by means of a grammar and the grammar itself evolves in order to reach a syntactical agreement in the team. Evolution is implemented by adapting an evolutionary algorithm where each agent in the team manages a population of chromosomes that represent possible grammars. Grammars can be used by agents to generate utterances which are subsequently applied in language games to describe spatial relations. A planning process builds the sentences, but agents select the syntactical alternatives according to their current communicative intentions. Results in two different linguistic task show how a shared grammar can be developed in the group of agents and show how grammars helps to present the language that is being evolved in a way similar to natural language.

Keywords Grammatical Evolution · Dynamics of Artificial Languages · Language Games · Multi-Agent Systems

Jack Mario Mingo
Computer Science Department, Universidad Autónoma de Madrid, Spain
Tel.: +34914972291
E-mail: mario.mingo@uam.es

Ricardo Aler
Computer Science Department, Universidad Carlos III de Madrid, Spain
Tel.: +34916249418
E-mail: aler@inf.uc3m.es

1 Introduction

In the last decades there has been an increasing interest in the study of complex languages, including human languages, by means of computational models. These models can help researchers both to understand important issues on human languages, or they can be used to develop artificial languages for artificial organisms (agents or robots). The model we present in this work falls within the second approach. Our model does not try to explain features of human languages or their origins. Rather, what we do in this work is to select, adapt, and use some hypotheses about human languages in order to propose a complex process of language evolution for a team of artificial agents. Artificial agents (or eventually robots) with a complex communicative system that develops autonomously, without having to be explicitly told how to use a language, could increase the autonomy and flexibility of agents performing diverse tasks. In particular, we study how a team of agents can develop a limited spatial language with a syntactic structure. A spatial language is basically a set of sentences which are focused on communicating information about objects and their relative positions. Spatial sentences share syntactic and semantic properties with other natural language sentences, but usually spatial sentences involve a spatial perspective [1]. Quoting Schober: "the speakers describing locations can speak from their own point of view, from their partner's or from another perspective which avoid the choice" [2].

The rest of the paper is structured as follows: in Section two a review about related works is showed. Section three describes the two evolutionary algorithms which are at the core of the proposed model will be described. Section four describes a basic problem to be solved and the setting is specified. Sections five and six explain the model in depth. Section seven analyzes the results for the basic problem. Section eight proves the model in a more complicated problem and analyzed results in this case. Finally, in Section nine we draw some conclusions and propose some interesting lines of work for the future.

2 Related Work

Initial works in the topic of spatial language were logically devoted to create vocabularies. A pioneer work about a self-organized spatial vocabulary developed by a group of agents was proposed in [3]. In this work, agents have to identify each other using names or simple spatial descriptions in two dimensions. The proposed model follows the hypothesis that language is an autonomous adaptive system which emerges by means of a self-organizing process. In [4], some strategies for the simulation of vocabulary agreement are analyzed in the context of a simple spatial environment, where each agent tries to convince other agents about the words it uses in a social interaction. Vocabulary in this work is limited to concepts such as north, south, east and west. The agents use their words and decide whether to change or to maintain their previous

belief about the word. Steels and Lara's works are interesting here because they treat the problem of spatial vocabularies and they propose models to solve it. Finally, we cite the work of Maravall et al. on emergence of vocabularies because, although they do not focus on spatial languages, we use some of their approaches in our present work. Maravall's works follow two different perspectives: reinforcement learning [5] and evolutionary [6]. In the former, a mapping model between meanings and symbols by means of associative matrices is proposed, that we also use here. Our approach is also evolutionary, as his second work.

The process of emergence of spatial vocabularies is a required step for subsequently developing more complex linguistic structures but in this article, it will be assumed that the agents have developed a set of basic vocabularies for objects and spatial relationships, by the means mentioned in previous paragraphs. The next step implies considering the syntax, which is a central issue regarding language. Syntax for artificial agents was first studied in [7] although it was limited to certain word order constraints which emerged from language games. Steels offers a functional point of view for grammars and he proposes here that syntax is a product of a general cognitive skill. His model is based on frame-like structures and complexity is included as operations in the frames. Some of Steels' collaborators proposed in [8] a model that connects low-level perception with categorization, hierarchical meaning construction, and syntax. A first step is a conceptualization that maps agent inputs into concepts. As the number of input data and events can be potentially huge, a filtering process is carried out by a semantic subsystem. Finally, a categorical grammar is used to transform semantic descriptions into natural language and vice versa. Recently [9] proposed a semantically oriented approach to explain the origins of syntax. Spranger and Steels employ robots with a spatial grammar where a syntactical structure emerges as words grouped in classes (nouns, adjectives, and verbs). Subsequently, word classes allow the model to build more complex noun or verb phrases. In a functional grammar the word classes suggest possible grammatical functions and they can be organized in a network showing a specific communicative goal such as an object reference. In order to build a network, the authors use a formal device known as Fluid Construction Grammar [10] and a representational system called Incremental Recruitment Languages (IRL) which is based on intensional logic and procedural semantics. The meaning of a sentence is a mental program combining a set of semantic functions in the network nodes. Experiments in this work show how having a grammar saves effort to the agents in the evolution of the language. All these models emphasize both the grounded relation between signal and meanings and the scheme for a joint attention task among the agents. Starting from these items, the models suggest a cycle where the speaker utters a sentence and the hearer tries to recognize the sentence during one or several language games. Besides, agents usually employ the signal associated to the meaning, but they can invent or repair sentences if they do not have a sentence or they do not understand it. Finally, the process of alignment is

the final stage in order to share a common language. This alignment process is the step we address in the proposed model.

To finish this review, it is worth noting the Lingodroids project ([11]), which is another proposal to solve grounding problems between symbolic and perceptual worlds in line with the works of Steels or Spranger. The Lingodroids robots are endowed with a kernel with two components: 1) an architecture based on the rodent's hippocampus that allow the robots to build an internal map of their environment and 2) a language learning system based on neural networks. Each robot moves around its environment and builds a cognitive map. Then, they try to develop a symbolic language by means of several language games such as where-are-we, go-to or what-direction, to mention a few ones. Experiments show how robots are able to find each other in a specific place using the developed language.

The commented works are examples about how a group of artificial organism can be endowed with complicated communicative systems. Some models fit in a functionalist point of view and other follow a biological one. But all of them take ideas from other fields such as psychology, linguistic, philosophy or neuroscience. We will also adopt some of the hypotheses and ideas used to explain natural languages in order for a team of agents to develop an artificial language autonomously. Specifically, according to the structure of natural languages, we will use a formalism based on context free grammars. It is well known that this kind of grammar is powerful enough to manage two of the main properties of language: compositionality and recursion. Formal grammars are a valuable tool to generate and recognize sentences in a language. However, we follow Austin and Searle in believing that agents must have a reason to generate the sentences. They envisaged language as a kind of behavior focused in communicative ends and they developed a *Speech Act Theory* [12,13]. Based on the *Speech Act Theory*, other researchers proposed a *Plan-Based Theory of Speech Act* [14,15]. In this theory, speech acts are considered as ordinary acts, in which case a plan can be applied in order to produce the act. Speech Act Theory represents a functional point of view about the natural language and it is the approach we will adopt here. Besides, intentionality is a key factor in this theory because, as we have mentioned, humans speak because they have intentions to communicate with each other. Putting together the structuralist and functionalist language dimensions, we propose here a model that uses an evolutionary algorithm and a planning process. The former is to evolve the grammar that the agents try to share (structure), and the latter is to build sentences based on the evolved grammar. The agent's communicative intentions are essential in the planning process in order to choose the most suitable sentences (function). The evolution of the grammar allows to analyze how language evolves and why some syntactical structures are chosen instead of other ones. The process that will be studied here is that of syntactical alignment within the team of agents, so other phenomena such as invention of new words or the process of understanding other agent's sentences, will be omitted. Syntactic agreement will be analyzed by means of *language games*, in the sense envisaged by Wittgenstein [16]. That is, agents associate linguistic expressions

with spatial concepts and relations by means of their mutual communicative interactions. Therefore, language use is the key factor of its development.

To some extent, main differences between this model and the reviewed works is the formalism based on generative grammars and the inclusion of intentions as bias to guide the evolution of the grammars. When we talk about intentions we mean that linguistics intentionality is represented in some explicit way. In our case the intention is simplified as symbolic functions. Cited works do not considerer so explicitly the intentionality. With regard to the generative grammars we use intensively this kind of formal device as a medium to generate easily sentences. Most approaches that consider syntax in languages for artificial agents or robots do not use explicitly a grammar but they use other formalisms. However, grammars are powerful device and they can potentially generate infinite sentences.

3 Grammatical Evolution and Grammatical Evolution by Grammatical Evolution

This section reviews the evolutionary algorithm that we use in the model. It is important to remark that although this algorithm uses concepts inspired from linguistics (like that of a universal grammar), they should not be interpreted as strictly belonging to linguistic theories. Grammatical Evolution (GE) was proposed by Collins, Ryan and O'Neill [17]. It is an evolutionary algorithm that uses variable-length linear chromosomes. Each chromosome encodes how to generate a candidate solution for a particular optimization problem. It is a grammar-based evolutionary algorithm in the sense that the chromosome is used to generate the candidate solution by applying production rules of a user defined grammar. GE separates clearly genotype and phenotype. The genotype (chromosome) is represented by a binary string while the phenotype (candidate solution) could be whatever structure that can be described by means of the grammar. The mapping process which transforms the genotype into its corresponding phenotype is split in two stages: transcription and translation, that will be described later. In short, genotype and grammar together define the means to build the phenotype. For instance, let us suppose we are interested in evolving a Java computer program to control a robot moving across a corridor avoiding collisions with the wall. In that case, candidate solutions (phenotype) would be such Java programs, the grammar would be the subset of the Java language required to solve the problem, and the chromosomes (genotype) would contain information that can be decoded by means of the grammar to create the candidate solution. This will be explained in more detail later.

The ideas of GE can be applied to evolve the grammar itself. This point of view was adopted in a new algorithm called Grammatical Evolution by Grammatical Evolution (henceforth referred to as GE^2) [18]. In order to evolve the grammar, GE^2 uses a second grammar which specifies the properties that

the original grammar can adopt. Therefore, in GE^2 two different grammars are supported:

- A grammar that the authors of the algorithm call the *universal grammar*, which describes the rules to build grammars. This grammar could represent the concept of grammatical universality as it has been proposed by some linguists and it would act as a *meta-grammar*. In any case, it is important to differentiate the concepts about grammars that the algorithm uses and similar linguistic concepts, so the term *universal grammar* or *meta-grammar* in GE^2 simply refers to its role in order to "dictate the construction of the solution grammar" [18].
- A grammar called the *solution grammar* which describes the rules to build candidate solutions. This grammar would represent the grammar which is traditionally used in standard GE, but in GE^2 it is evolved by the evolutionary algorithm, instead of having to be defined by a user.

GE^2 carries out a double evolution process where two variable-length binary genomes are involved. The first one generates a solution grammar from a meta-grammar while the second one generates candidate solutions from the solution grammar. GE^2 is the evolutionary algorithm adopted in our work because it allows to evolve the most suitable grammar in order to describe the current spatial situation. In this case, a meta-grammar describes the rules to build several solution grammars which in turn allow the model to utter the sentences about the spatial scene. However, sentences are generated from those solution grammars by means of a planning process as we will explain later, instead of applying an evolutionary process as the original GE^2 really does.

The concept of meta-grammar is important to study how an artificial language can arise, because it allows the model to evolve different solution grammars in a self-organized process. This would not be possible if a single solution grammar were previously designed by a human designer. Self-organization here means that the group of agents as a whole and each particular agent develops and evolves its own grammar without human supervision, that is, the meta-grammar only contains generic syntactical rules (a kind of innate knowledge we could say) and they can be applied during the evolutionary process in order to find the solution grammar that best fits the current linguistic situation. This is more flexible than endowing each agent with a hand-made initial solution grammar.

4 Experimental Setup

In this section we describe a simple task, that will be used in the experiment section, but here it will be useful to present the model. In this problem, agents look for a consensus in linguistic terms, in order to describe a spatial situation where several objects maintain a spatial relation among them, such as *left, right, front* or *behind*. The symbolic expressions of the utterances have syntactic structure. This syntax is simple and for now, words such as determiners,

prepositions or verbs are not used. To simplify, we will call this communication system a language, because words are ordered according a grammar, although a language also implies semantic and pragmatic aspects (that we simplify here). We use three simple objects in the experiments: *table, chair* and *wardrobe*. Figure 1 shows the typical configuration used in the experiments:

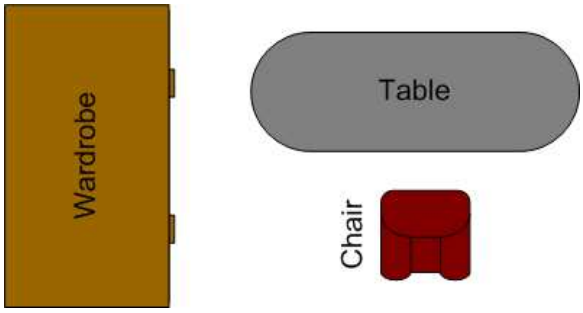

Fig. 1 Scene with the spatial environment used in the experiments

In Figure 1, we can identify a referential scene which includes several linguistic situations. For example, we have spatial relationships such as:

- *the chair is on the right of the wardrobe*,
- *the wardrobe is on the left of the chair*,
- *the chair is in front of the table*,
- *the table is behind of the chair*,
- *the table is on the right of the wardrobe*,
- *the wardrobe is on the left of the table*

The figure defines a perceptual situation to be observed and analyzed by the agents. In psychology, this is called a joint attention scene and it is a conceptual frame where language is developed. Starting from this perception, each agent tries to associate a symbolic representation by means of language. This language will be useful to the agents if they are able to get a consensus about the symbolic representation of the scene. As we are interested here in the cognitive functions of agents, it will be assumed that agents can recognize objects and relations among them by means of methods such as sensory discriminant variables, typical of Pattern Recognition methods. An additional assumption is that all agents have a common lexicon for the objects located in the environment and for their spatial relationships. This way, a basic vocabulary for these components is assumed to be previously created. To simplify, we use Prolog-like functions in order to represent objects, meanings, and symbols for the initial lexicon. Table 1 shows the functions to represent a "chair" object and the "right" spatial relationship at the meaning and symbolic levels.

Having into account this simplification, each agent extracts the object features when it looks at the object, then it associates the corresponding meaning by means of an internal representation, and finally it relates the meaning to its

Table 1 Symbolic functions to represent basic meanings and object

Function Name	Description
$meaning\_of\_chair$	$meaning(chair) :- features(small, rectangular, legs, back)$
$symbol\_of\_chair$	chair
$meaning\_of\_right$	$meaning(right(meaning(A), meaning(B)) :- greaterXCoord(meaning(A), meaning(B)))$
$symbol\_of\_right$	right

symbol. Similar ideas are used for the spatial relationship, but in this case two objects are associated, so we need to compare some feature between them. To simplify, we follow the ideas of [3] and we consider that *"the object A is on the right of the object B"* if the x-coordinate of the object A is greater than the x-coordinate of the object B with regard to the referential system of the agent that is looking both objects. Both Steels' work and this work adopt a relative frame of reference, according to Levinson terminology [19], so we suppose that all the agents in the team watch the scene from the same place and all of them have the same coordinate system.

As we mentioned above, communicative intentions play a main role in this model because they are involved in the process of choosing the best production rules in the grammar as we will explain later. To simplify, we model the intentions as a set of linguistic tasks which are described by means of symbolic functions similar to those that Table 1 shows.

5 Description of the Model

As we commented in the Introduction, the proposed model is based on two computational strategies: *evolution* and *planning*. Evolution drives the grammatical agreement and planning drives the process for generating sentences following the *Plan-Based Theory of Speech Act principles* [15]. Essentially, the model is executed on each agent in the team of agents, as follows:

1. An initial population of candidate solutions representing grammars is randomly generated. Candidate solutions are encoded into variable-length binary genotypes (or chromosomes).
2. An evolutionary algorithm is applied:
   (a) *Transcription* and *translation* processes are applied to each chromosome. During the translation process, the *meta-grammar* or *universal grammar* is used to build a *solution grammar* from the chromosome. The solution grammar includes a set of production rules that are potentially relevant to the informative intention.
   (b) The solution grammar is evaluated in two steps: (i) a sentence is generated from the solution grammar by means of a planning algorithm; (ii) the sentence is uttered in a language game. Both steps are executed with all the possible *intentional sentences* that can be generated starting from the solution grammar under evaluation. With *intentional*

sentence, we refer to a sentence suitable from the agent's intentional point of view, that is, according to the set of linguistic tasks to be described. As a result of evaluation, a fitness value is assigned to the chromosome.

3. The termination criterion is checked:
   (a) If the syntactic agreement is attained or the maximum number of generations is reached, the process ends.
   (b) Otherwise: (i) genetic operators (crossover, mutation, selection, and duplication) are applied to chromosomes in the population and a new population of chromosomes is obtained (a new generation); (ii) return to step 2 and repeat the process.

Figure 2 shows the process graphically.

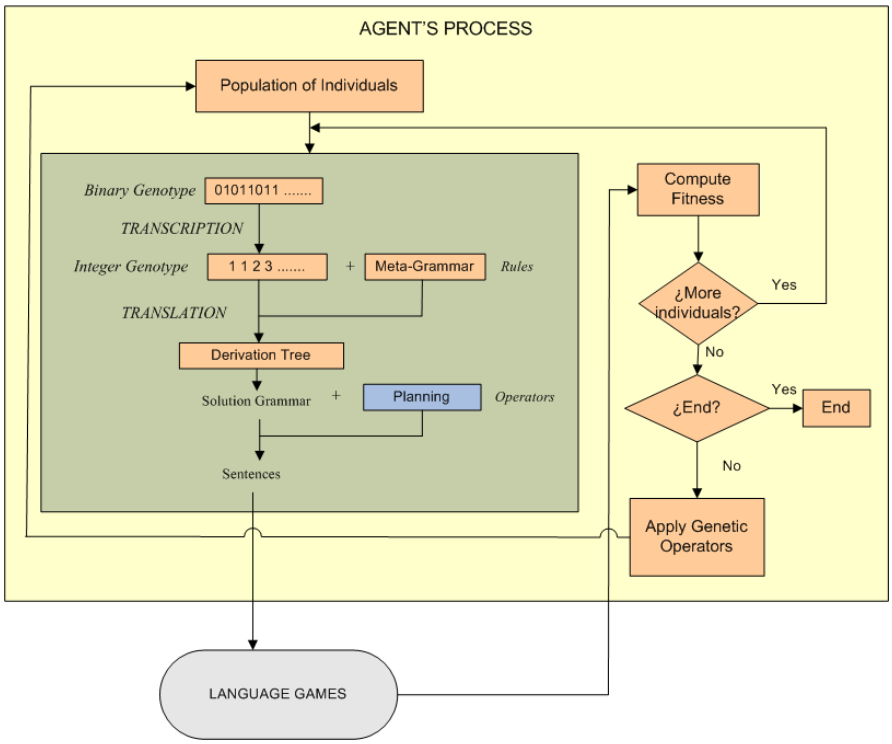

Fig. 2 Agent's Process

In the previous description we have used the terms *team* and *population* and we would like to clarify that in this work, they do not refer to the same concept. The term *team* stands for the group of artificial agents that looks for syntactic agreement. On the other hand, the term *population* is a concept from the evolutionary computation field, and stands for a collection of candidate solutions (encoded as binary chromosomes), which is a component of

the evolutionary algorithm (GE^2 in this case). The process of Figure 2 is executed within each agent in the team. Therefore, a population of chromosomes is stored within each agent.

In order to understand better how the model performs, the main steps will be described next in more detail. First, the process by which a solution grammar is obtained from a meta-grammar by means of an evolutionary process (green rectangle and orange boxes and diamonds in Figure 2), and by which a sentence is generated from a solution grammar by means of a planning process (blue box in the Figure 2). Second, language games and chromosome fitness computation will be explained.

5.1 Generating a Solution Grammar from a Meta-grammar

In order to generate a solution grammar from a chromosome by means of a meta-grammar (see production rules in appendix A), the transcription and translation processes are applied as it is usual for both GE and GE^2:

1. *Transcription.* In this stage, the original binary string which stands for each chromosome in the population is transformed into an integer string. We will show this process through an example. Consider a binary string which is composed of a string of bits or *codons* as they are usually named: 00-01-10-01-00-01-11-01-10-00-00-01-10-00-01. A hyphen is used only to separate each group of codons. After applying the transcription process this binary string is transformed into the equivalent integer string: 0-1-2-1-0-1-3-1-2-0-0-1-2-0-1. In this new string, each integer is simply the decimal representation of the corresponding codon.
2. *Translation.* Starting from the integer string and the meta-grammar rules, a solution grammar can be obtained. In the previous example, the derivation tree associated to this process is shown in Figure 3. Grey nodes represent terminal symbols. Numbers near the nodes represent the production rule number to apply to the non-terminal symbol (white nodes). These numbers are taken from the integer string that we computed in the transcription process and they appear from left to right as the integer string is traversed. In order to determine which production rule has to be applied, the following equation is used: $Rule = CDV \ \% \ NR$; where $\%$ is the modulo operator, CDV is the acronym of *codon integer value* and NR is the acronym for the *maximum number of rules* corresponding to the non-terminal that must be expanded at the current time. The meta-grammar production rules are defined in appendix A. For example, starting from the first node which corresponds to the meta-grammar start symbol (labelled as *grammar* in Figure 3), the production rule is computed as $Rule = 0 \ \% \ 1$ because 0 is the first integer in the integer string and 1 is the maximum number of production rules associated to the meta-grammar start symbol. Result is 0 in this case, so the first (and only) production rule is chosen. The next operation to apply is the expansion of the leftmost non-terminal symbol in Figure 3. That is, the non-terminal <*rel-expression-def*>. In this case

the production rule to apply is computed by $Rule = 1\ \%\ 2$ where 1 is the next integer in the integer string and 2 is the number of production rules for the non-terminal <*rel-expression-def*>. The result is 1, so the second production rule is chosen $< relexpression >' \mid' < relexpressiondef >$ (indexing starts at 0). Translation continues until a solution grammar is built, if possible.

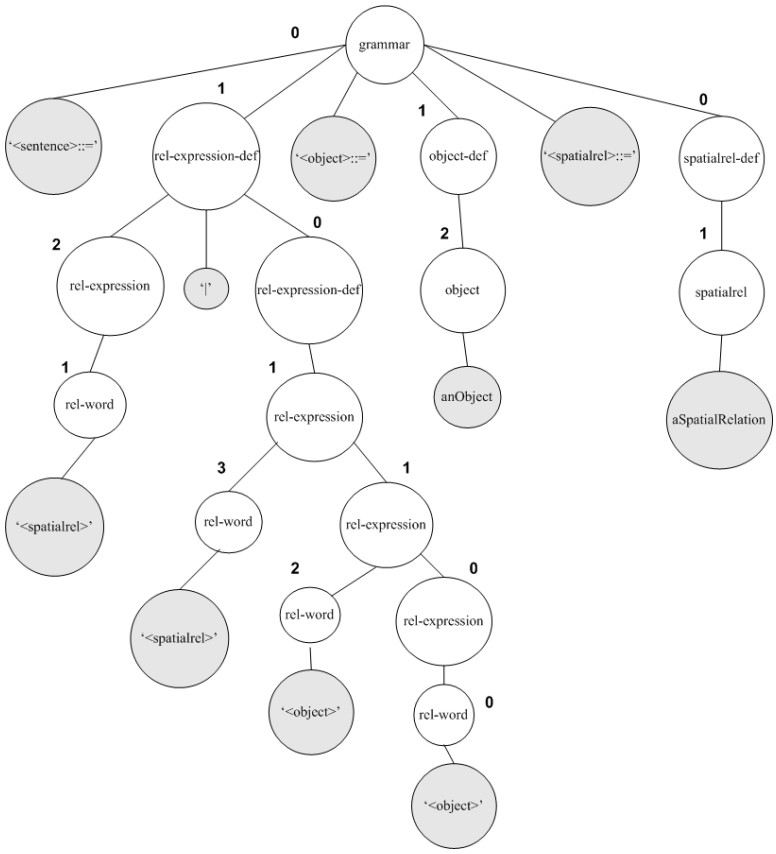

Fig. 3 Derivation tree to generate a solution grammar

The translation process is correct if it is possible to build a well-formed solution grammar. If a solution grammar cannot be generated, the chromosome is considered incorrect and it is discarded. In the example, the solution grammar contains the following rules (in Backus Naur Form or BNF notation):

```
<sentence> ::= <spatialrel>
             | <spatialrel> <object> <object>
<object> ::= anObject
<spatialrel> ::= aSpatialRelation
```

Given that GE^2 evolves grammars, terminal and non-terminal are present in the evolved solution grammars. It is worth noting the role of the terminal symbols *anObject* and *aSpatialRelation*. These symbols play the role of lexical categories such as nouns, adjectives or verbs but they are constrained to a spatial context. Lexical categories are very common in the literature about phrase structure grammars and natural languages. They are an important concept here because they allow the model to generalize the type of phrase structures that a solution grammar can generate. Therefore, a **phrase structure** includes one or several lexical categories which are replaced by the specific symbols when the final sentence is elaborated, as will be shown in the next section. On the other hand, the non-terminal symbols <*sentence*>, <*object*> and <*spatialrel*> stand for syntactic categories and they allow the model to express the grammatical rules that are inherent to most languages. Thanks to a solution grammar, the question of syntax can be easily implemented in an artificial language. The evolved solution grammar in the example defines a language which will allow the agent executing the model to generate two types of sentences:

- A sentence with a single relational word if the first rule is chosen.
- A sentence with three words or symbols when the second rule is chosen: a spatial relational symbol followed by two simple object symbols.

Generating the most suitable sentence for the referential scene implies choosing the right phrase structure and elaborating the final sentence. This is precisely the task corresponding to the second part of the model that we explain in the next section.

5.2 Generating a Sentence from a Solution Grammar

Once a solution grammar has been translated from a chromosome, it has to be evaluated. Evaluation implies three steps:

1. To choose a suitable phrase structure by means of planning and the solution grammar.
2. To replace the lexical categories for the specific symbols in order to build the final sentence.
3. To evaluate the generated sentence in a language game with sentences generated and uttered by other agents.

In order to choose a suitable phrase structure by means of classic planning such as STRIPS [20] which is usually adopted in a *Plan-Based Theory of Speech Act*, we need to define concepts such as final, intermediate, and initial states, and also operators.

- *Initial State*. The starting symbol of the solution grammar is the initial state in the planning process. This symbol is always the non-terminal <*sentence*> (see first rule for the meta-grammar in the appendix section).

- *Final States*. Any expression made of terminal symbols only refers to a final state. A final state is a phrase structure and it contains only lexical categories. Phrase structures are built through a derivation process, as it is usual in formal language theory. During this derivation process the classical tree-like structure is built.
- *Intermediate States*. Intermediate states are intermediate steps in the process of building a derivation tree, i.e, they are associated with partial subtrees where the leftmost non-terminal symbol is the next term to be expanded.
- *Operator*. An operator represents an action to be applied to each state. Operators are the possible production rules to expand the non-terminal symbol.

In short, to apply an operator is similar to implementing a derivation step when the derivation tree is being generated. Essentially, the main problem is how to choose the production rule to apply as operator in an intermediate state. Remember that the term "production rules" in this case stands for the production rules of the solution grammar and not those of the meta-grammar. Solution grammar production rules are applied in the production of sentences while meta-grammar production rules are applied in the previous stage in order to generate each solution grammar. Returning to the initial question, we argue that intentions are the key here. Rules are chosen based on the agent's current informative intentions. Thus, only the solution grammar production rules supporting these intentions are selected to choose the right phrase structure. In terms of classical planning, this process corresponds to a search in the state space with a forward mechanism (see [21]). However, forward search is helped by means of the informative intention, which acts as a heuristic to guide the search. As we mentioned in the introduction section, *Plan-Based Theory of Speech Act* is followed here, so planning is used as the mechanism to choose and generate the final sentences. Other techniques might be used but we believe that the current approach can be easily enhanced with more features from this well known theory, once the initial framework is in place.

This planning process will be clarified by continuing with the previous example. In the solution grammar there are two production rules associated with the start symbol <*sentence*>: (1) <*spatialrel*> and (2) <*spatialrel*> <*object*> <*object*>. To define its current intention the agent needs to look at the referential scene (see Figure 1). After analyzing the scene, the agent perceives each object and spatial relation in the referential scene and maps meanings into symbols associated to simple objects and spatial relations (let us remember that vocabularies for simple objects and relationships are assumed). Nevertheless, the agent needs to combine this set of lexical categories in order to build the right phrase structure. As we know, the solution grammar provides the combination rules. From the agent's intentional point of view, it needs two objects and a spatial relation to describe the linguistic situation of Figure 1. The solution grammar first production rule can be discarded because it cannot

derive a sentence with all these items. According to the concepts usually employed in classical planning (see [21]), **applicable rules** are those production rules which are relevant to the agent's current informative intention. In order to determine why a production rule is applicable the system needs to analyze some details about the solution grammar. Specifically, the solution grammar has to be measured to determine which strings are reachable. A comprehensive and very interesting study about grammar measurement is found in [22] and we adopt some concepts here. In Hemberg's work, grammars are measured by means of a matrix representation of the grammar as it was previously proposed in [23]. Matrix representations help to determine which strings are reachable starting from the grammar start symbol. Two matrices are built in our case from the solution grammar: a *Non-Terminal Expectation Matrix* and a *Terminal Expectation Matrix*. First matrix allows to compute how many non-terminals are expected if a non-terminal is rewritten once. Second matrix is similar but it refers to terminal symbols instead of non-terminals. Details about both matrices and the way they can be computed can be found in [22] or [23]. To our interest, values in both matrices are numbers and they can be combined in order to get a matrix that allows the model to know if a specific terminal symbol can be reached from a non-terminal one. Following with the example, the evolved solution grammar has a matrix with information as Table 2 shows:

Table 2 Matrix for the evolved solution grammar in the example

	aSpatialRelation	anObject
<object>	0.0	1.0
<spatialrel>	1.0	0.0
<sentence>	1.0	1.0

Left column stands for non-terminal symbols and first row contains the terminal symbols in the grammar. Cells have a number greater than zero whether it is possible to reach a terminal symbol from the corresponding non-terminal symbol. In the example, we can reach the terminal symbols *aSpatialRelation* and *anObjet* from the non-terminal symbol <sentence>, which is the axiom of this solution grammar.

This way, applicable rules can be determined starting from this matrix and taking into account the agent's informative intention. In the example, after selecting the second production rule, the first derivation sub-tree is built. Figure 4 shows this situation. We omit < and > symbols in the non terminal nodes because we use the white color to distinguish them.

Next, the planning process applies a new operator to the new sub-tree and the leftmost non-terminal symbol is chosen. In this case, there is a single production rule associated to the <*spatialrel*> non-terminal symbol. This production rule derives the *aSpatialRelation* lexical category. The new sub-tree is shown in Figure 5.

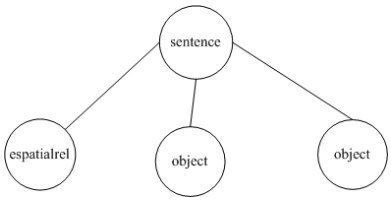

Fig. 4 First derivation step

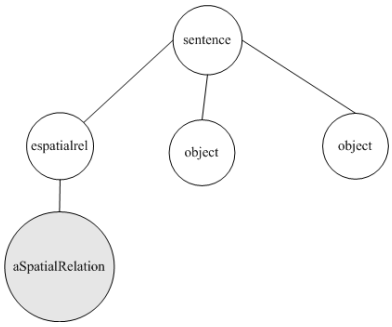

Fig. 5 Second derivation step

This process continues by replacing both <*object*> non-terminal symbols in the second derivation sub-tree. There is also a single production rule associated to this non-terminal according to the solution grammar and this production rule produces the *anObject* lexical category. As Figure 6 shows, this sub-tree has only terminal symbols as leaves.

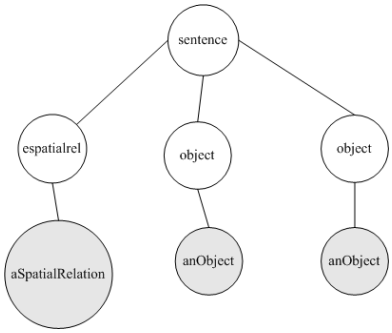

Fig. 6 Last derivation step

The phrase structure *aSpatialRelation anObject anObject* is finally generated from the solution grammar in this example. Then, it must be used in a language game with phrase structures generated by another agent. As the problem described in section three is simple, we can see that this example of phrase structure is valid for describing all the linguistic situations that Figure

1 shows. Language games and fitness assignment will be described in the next section but we need to deal with some issues first. The planning process in this example is simple: as the solution grammar contains only a few production rules, the derivation steps are straightforward. In fact, the concept of applicable rules has been exploited only in the first step, when the start symbol was expanded. The remaining sub-trees did not need to apply it because there was only a production rule in each case. However, the more difficult the solution grammar is, the more important is the concept of applicable rules, because the agent's informative intention helps to discard production rules that are not relevant to that intention. Finally, the planning process is applied in order to search for all the phrase structures starting from the same solution grammar as long as they can describe the current referential scene. If there are no applicable rules in any step the chromosome is discarded because it cannot generate a sentence that describes the linguistic situation.

5.3 Language Games

Language games emphasize language use and the interactions among individuals in communicative acts. The language games hypothesis is based on the supposition that most common expressions will be finally agreed. We need to define previously the concept of equality of sentences:

> *Two sentences are syntactically equal if they have the same words and the words are equally located.*

Language games are carried out by means of the algorithm of Figure 7. It is important to set this algorithm in the context of Figure 2. Specifically, it is executed in the box named *"Language Games"*. If an agent has evolved a solution grammar, it will be able to generate one or several phrase structures. Language games are the social event where the agents utter their sentences but previously they have to transform the phrase structures into a specific sentences. This is where the agent specialized modules replace lexical categories, such as *aSpatialRelation* or *anObject*, by the actual spatial relation or object. If an agent has not evolved a solution grammar or it cannot build a phrase structure, then it will participate with an empty set of sentences in the current language game.

```
For each agent in the team:
    Generate the set of sentences to describe the scene from the evolved solution grammar that is being evaluated
    Compare the agent's set of sentences with other agents' set of sentences and count successes
    Agent Communicative Efficiency <- number of successes (shared sentences / number of linguistic situations)
End for
```

Fig. 7 Language Games Algorithm

Each agent participates in the game with a set of sentences that was generated from the solution grammar that is being evaluated. A solution grammar

defines phase structures as we saw in previous section, and a phrase structure allows to build a sentence. This way, each sentence in the set of sentences tries to describe a particular linguistic situation in the scene. In the simple problem that we have defined in section three, all the linguistic situations can be described by the same phrase structure as we saw in the previous example, so in this case each agent participates only with sentences generated starting from this phrase structure. However, later we will describe a more complicated task where different linguistic situations exist in the scene and a set of different phrase structures are needed. A language game gathers the set of sentences from all agents and counts how many sentences are syntactically equal. Then, it divides the number of shared sentences between the number of linguistic situations in the scene. Finally, the language game output is precisely this numeric value, that we call *Communicative Efficiency*. Remember that each agent is endowed with an evolutionary algorithm with a population of chromosomes (which encode grammars), but each chromosome is evaluated in a different language game. In other words, if an agent has N solution grammars in its population, it can generate M sentences. A group of these M sentences, describing all the linguistic situations in the scene, will participate in the language game. The value N refers to the number of valid solution grammars that can be correctly evolved starting from the individuals in the population. The value M can be different than N because a solution grammar can generate zero, one or more sentences depending on the number of its production rules.

5.4 Fitness Assignment

Evolutionary algorithms require that chromosomes in the population are evaluated by assigning to them a fitness value. In this case, chromosomes encode grammars. But evaluation takes place during language games by counting the number of sentences uttered by all the agents, that are syntactically shared. Thus, solution grammars are indirectly evaluated during language games, by comparing the sentences generated from those grammars. This fitness value is basically the communicative efficiency that we defined in the previous section. Thus, the fitness of the chromosome (and its corresponding grammar) is defined as:

$$Fitness = NSS/NLS \tag{1}$$

Where **NSS** stands for the number of shared sentences among the agents in the language game and **NLS** represents the number of linguistic situations in the referential scene. Remember, that the scene can contain several linguistic situations that are described by means of one or several phrase structures which in turn are defined by the solution grammar that is being evaluated, so we have to divide the number of shared sentences between the number of linguistic situations. The fitness is *maximum* when this value is equal to the number of agents in the team and this case represents the moment when the

team has reached syntactic agreement, because all the agents use the same sentences to describe the scene.

The fitness that Equation 1 defines can be considered as an external measure because even though the population of chromosomes is an internal component within the agent's architecture, the sentences are evaluated in a language game which is external (all team members are involved). Thus, the fitness value depends on the successes in the language games.

5.5 Summary of the Model

Once the individual processes have been explained in detail, it can be useful to have a look at the whole model in order to provide a complete vision of the work. This section is an informal review about the model, formally introduced in Section 5. First, a computational version of the model is included within each agent in the team. The algorithm includes a GE^2 evolutionary algorithm so that a population of binary-strings (chromosomes) is evolved in order to generate solution grammars (phenotype) for each agent. The aim is to evolve the solution grammar that best describes the current spatial situation. Every solution grammar can generate several sentences because it depends on the production rules included in the grammar. In order to select the best sentence to utter, a planning process is used to discard irrelevant sentences, according to the current informative intention. Intentional sentences are uttered in the context of a language game where all the agents in the team participate, and each agent counts how many other agents' sentences are equal to its own. The process goes on until agreement is reached for each spatial situation or a maximum of evolutionary steps and language games is exceeded.

6 Discussion of Results

The experiments have two goals. First, they try to show whether the proposed model is able to find a syntactic agreement or not. Second, they try to analyze how language emerges and how grammars evolve.

As we showed in Figure 1, there are at least six linguistic situations in the referential scene. The first goal is to evolve a suitable solution grammar for describing each linguistic situation and reaching agreement. We ran 30 executions with teams of 5, 10, 15, 20, 25, 30 and 50 agents. Table 3 shows the main evolutionary parameters we use in the experiments. A standard roulette-wheel algorithm is used as selection method.

It is important to differentiate concepts about evolutionary algorithms and concepts about the language development process in artificial agents. In this case, *population size* stands for a typical evolutionary parameter, which represents the number of chromosomes in the population (it is therefore not the number of agents in the team). *Maximum generations, crossover probability* and *mutation probability* are also parameters of the evolutionary algorithm

Table 3 Evolutionary Parameters

Population size	150
Maximum generations	100
Crossover probability	0.8
Mutation probability	0.05

running within each agent. We use standard single point crossover and standard bitwise mutation.

6.1 Results about Syntactic Agreement

First, we will analyze whether syntactic agreement is reached or not and the influence of the team size on this agreement. Figure 8 shows success probability results taking into account the whole set of experiments. Given that evolutionary algorithms are stochastic, and that for each team size there are 30 different executions, the success probability measures the proportion of executions where syntactic agreement was reached.

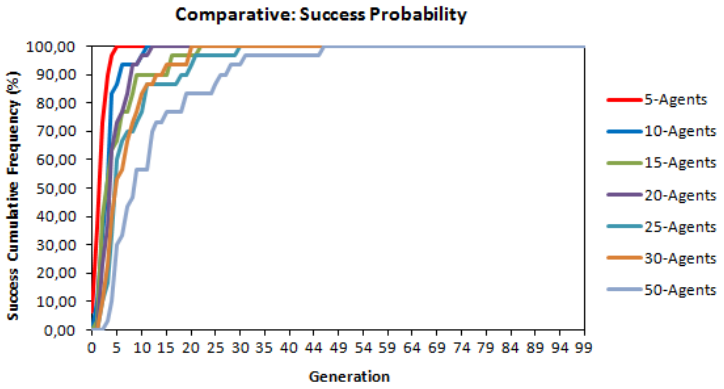

Fig. 8 Success Probability

As we can see in Figure 8, a complete agreement in terms of linguistic expressions is always attained eventually. Results about success probability are very common in evolutionary algorithms in order to show how a model or system performs. Figure 8 shows cumulative probability values which measure the proportion of executions that reached agreement at generation x of the evolutionary algorithm (this is an estimation of the probability of agreement at generation x). Figure 8 shows that the model is able to achieve syntactic agreement in less than 99 generations (of the evolutionary algorithm). In fact, in most tests it needs fewer than 30 generations. 100% success probability was

obtained in all experiments so it can be argued that the model performed well under these circumstances, although time increases as team size increases.

Success probability is an interesting value in order to know whether the model solves the problem and shows how many generations it needs. But it cannot tell us about the solutions themselves. Table 4 shows the syntactical structures that the team finally agreed on in experiments with 5 and 50 agents (we only show results for the first 10 experiments in each case)

Table 4 Evolved syntactical structures with different team sizes

Test	Syntactical structures (5-agents)	Syntactical structures (50-agents)
1	anObject aSpatialRelation anObject	anObject anObject aSpatialRelation
2	anObject aSpatialRelation anObject	aSpatialRelation anObject anObject
3	anObject aSpatialRelation anObject	aSpatialRelation anObject anObject
4	aSpatialRelation anObject anObject	anObject anObject aSpatialRelation
5	aSpatialRelation anObject anObject	anObject anObject aSpatialRelation
6	anObject aSpatialRelation anObject	aSpatialRelation anObject anObject
7	aSpatialRelation anObject anObject	aSpatialRelation anObject anObject
8	anObject anObject aSpatialRelation	aSpatialRelation anObject anObject
9	aSpatialRelation anObject anObject	aSpatialRelation anObject anObject
10	anObject aSpatialRelation anObject	aSpatialRelation anObject anObject

A syntactical structure or phrase structure contains lexical categories which are replaced by the specific objects and spatial relations according to the current linguistic situation. For example, in the fourth experiment with 5 agents the team agreed on the syntactical structure "aSpatialRelation anObject anObject" (see test 4 in Table 4, second column). Table 5 summarizes the solution grammars evolved by agents in this experiment.

Table 5 shows the final solution grammars. It can be seen how this set of grammars can generate a phrase structure such as *"aSpatialRelation anObject anObject"* because each grammar includes a rule which allows the agent to elaborate that kind of expression. We have to remember that syntactic categories such as <*sentence*>, <*object*> and <*spatialrel*> finally derive into lexical categories. The planning process that we described before is responsible for generating the sentences that each agent utters in the language game. As we can see, each agent is able to utter the same linguistic expression. Therefore, syntactic agreement can be reached, and this is what finally happened in this example. Evolved grammars are usually similar in all experiments so this example is a good choice in order to study how the language emerges in the team. At least, all grammars include the production rule $< sentence >::=< spatialrel >< object >< object >$ which can generate the syntactical structure that the agents need to describe the referential scene shown in Figure 1. Agents 0, 1 and 4 include other production rules, that are not relevant to this scene but they could be relevant in other contexts. On the other hand, agents 2 and 3 only include the commented production rule. It is evident that syntactic agreement is possible only when the agents in the

Table 5 Evolved solution grammars in a five-agent experiment

Agent ID	Evolved Grammar

*Agent*_0	`<sentence>::= <object> <object>`
	`  | <spatialrel> <object> <object>`
	`<object>::= anObject`
	`<spatialrel>::= aSpatialRelation`

*Agent*_1	`<sentence>::= <spatialrel> <object> <object>`
	`  | <spatialrel> <object>`
	`<object>::= anObject`
	`<spatialrel>::= aSpatialRelation`

*Agent*_2	`<sentence>::= <spatialrel> <object> <object>`
	`<object>::= anObject`
	`<spatialrel>::= aSpatialRelation`

*Agent*_3	`<sentence>::= <spatialrel> <object> <object>`
	`<object>::= anObject`
	`<spatialrel>::= aSpatialRelation`

*Agent*_4	`<sentence>::= <object> <object>`
	`  | <spatialrel> <object> <object>`
	`<object>::= anObject`
	`<spatialrel>::= aSpatialRelation`

team have evolved similar grammars and these grammars contain at least a production rule which reflects the current informative intention.

The advantage of evolving grammars with a phrase structure is clear because they allow the agents to adapt expressions to specific objects. This way, a expression such as *"aSpatialRelation anObject anObject"* in the studied example can be exploited in order to describe similar linguistic situations such as those that Figure 1 contains. In fact, the final result of the example in terms of syntactic agreement is shown in Table 6.

Table 6 Results of the syntactic agreement in a typical example

Informative intention (meaning level)	Linguistic expression (symbolic level)
meaning(right(meaning(chair),meaning(wardrobe)))	right chair wardrobe
meaning(left(meaning(wardrobe),meaning(chair)))	left wardrobe chair
meaning(front(meaning(chair),meaning(table)))	front chair table
meaning(behind(meaning(table),meaning(chair)))	behind table chair
meaning(right(meaning(table),meaning(wardrobe)))	right table wardrobe
meaning(left(meaning(wardrobe),meaning(table)))	left wardrobe table

Left column in Table 6 stands for the current informative intention and it corresponds to the linguistic situations that we defined in section 4, where six different situations were contained in the same referential scene in Figure 1. The right column represents the linguistic expressions that all the agents agree in order to express each informative intention. It is worth to analyze how two different levels of knowledge are connected here. On the one hand, informative intentions are represented by means of *"meaning"* functions in a semiotic network scheme as we proposed in Table 1. These kinds of functions are a very simplified representations of the agent's internal knowledge but they are essential in order to link the physical world with the perception of the world that the agent builds. We work here with a simplification because we are interested in the symbolic level, which is represented by means of lexical symbols, such as those that right column in Table 6 shows. Expressions in this second column are simply generated by replacing the lexical categories with the specific objects and spatial relationship but they have to be agreed in the corresponding language game. However, we do not need to evolve new solution grammars in each linguistic situation because all of them are essentially a combination of the same items, a spatial relationship as the first word, followed by two objects as the second and third words. Grammaticality is induced by means of the solution grammars that we showed in Table 5.

Returning to the global results, Table 4 displays how different combinations were agreed in the experiments. For example, the linguist expression in a 5-agents team (second column) for the tests 1, 2, 3, 6 and 10 is built in infix notation while the expression for the tests 4, 5, 7 y 9 is build in prefix notation and the expression for the test 8 is build in postfix notation. Infix, prefix, and postfix terms refer to the position where the spatial relationship appears in the expression. This diversity is equally evident in the case of a 50-agents team.

6.2 Grammar-Language Evolution

An interesting measure that we can analyze during the model evolutionary stage is the evolution of the fitness value associated to the best chromosome in the population and the evolution of the population average fitness. Figures 9 and 10 show the average results of the different team sizes taking into account the complete set of 30 executions in each case.

In this context, the concepts of "best chromosome" and "population" stand for the arithmetic mean values, that is, the values computed starting from each individual agent's values and associated to the team as a whole. We prefer to show average values instead of each agent's results in order to save space. Besides, the average values are representative enough. On the other hand, both terms are evolutionary concepts and they are associated with the population of chromosomes that each agent manages. It is important to clarify these different items of the model because we can lose the perspective between the evolutionary stage, which is a private event associated to the agents and the language games, which are a social event in the team. As we have defined the

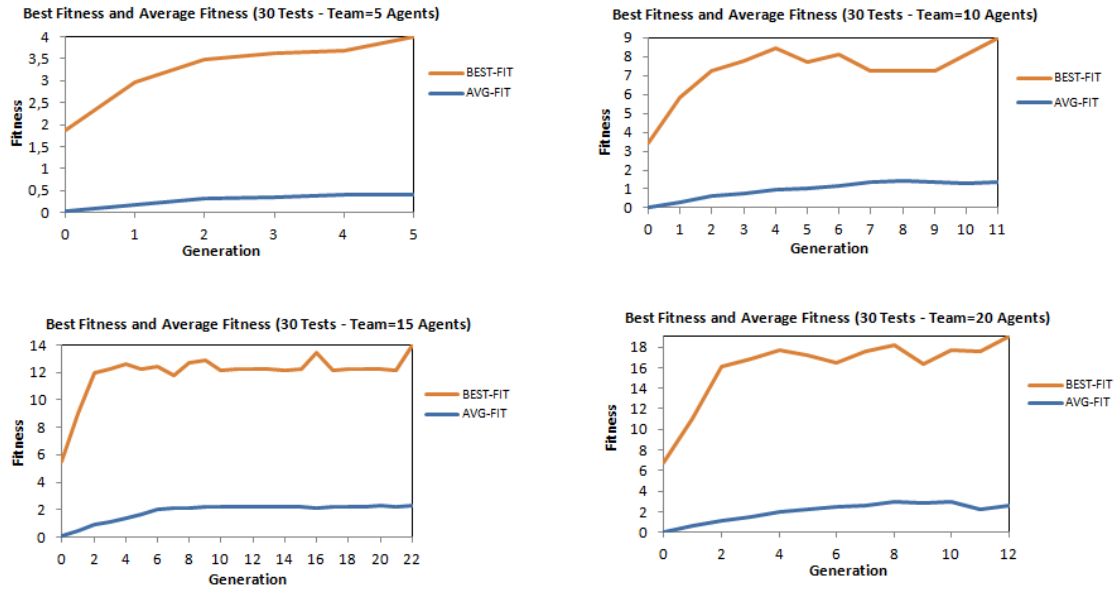

Fig. 9 Fitness Evolution through the Language Games Developing Process: 5, 10, 15 and 20 agents

fitness value in Equation 1, it is an external measure because it is computed on the number of syntactically equal sentences uttered by agents in a language game. Basically, from the point of view of an agent, an agreement is attained when the number of syntactically equal sentences is equal to the number of agents in the team, minus one (the agent is left out because it obviously agrees with itself). This means that the team agrees to use the same expressions in order to describe the current referential scene. Obviously, we take into account the number of linguistic situations in he scene but in this simple problem we only need a syntactic structure or phrase structure to describe all the linguistic situations in the scene. Therefore, we can see in Figures 9 and 10 how the fitness associated to the best chromosome arises as the number of generations increases, until finally the maximum fitness and syntactic agreement is reached. As all the experiments were successful, the final fitness is always the maximum possible fitness. For example, in a team with 5 agents the maximum fitness is 4, in a team with 10 agents the maximum fitness is 9 and so on. Evolution of the average fitness is very different and results in this case show how most chromosomes in the population are less suitable to generate valid sentences according to the current informative intention. This result merits a reflection. The main reason why the population of solution grammars that each agent manages is so poor can be found in the applicable rules concept. As we commented, a rule is applicable if it reinforces the agent's current informative intention. They help the evolutionary process to discard production rules that are not consistent with this intention, but they are a strong bias

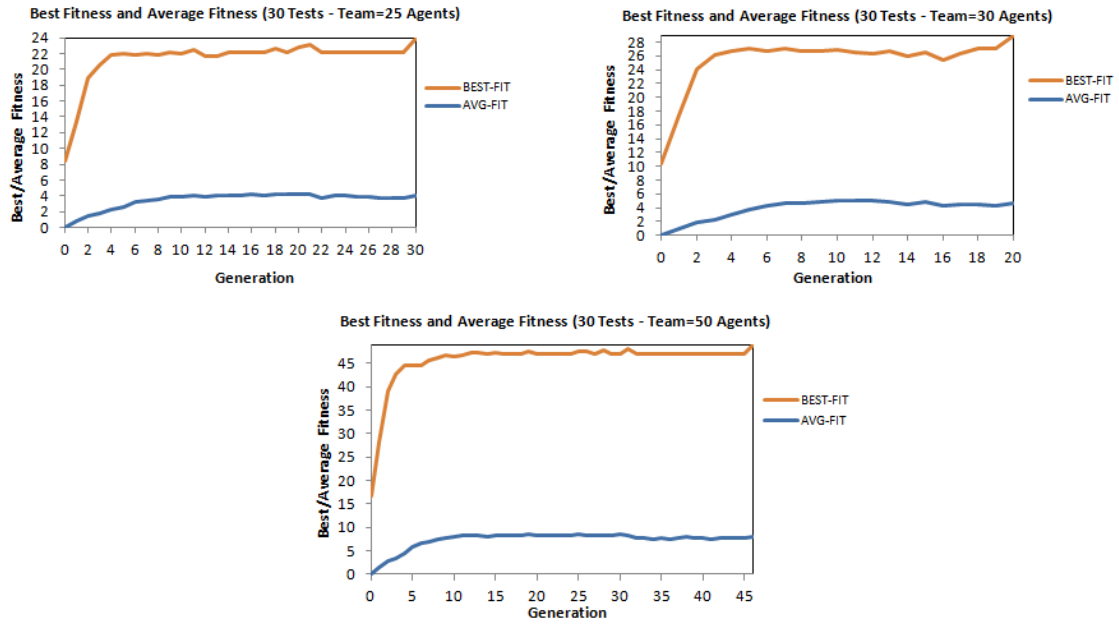

Fig. 10 Fitness Evolution through the Language Games Developing Process: 25, 30 and 50 agents

in the process as well, as many grammars are discarded and not evaluated in language games. Consequently, the evolutionary process is guided only by the best chromosome in the population. We tested some experiments where this concept was totally or partially relaxed but the results showed that agents would agree non intentional sentences because agents in this model have no other knowledge but the meta-grammar rules and their own intentions. In any case, practically all the proposed models in the literature that we reviewed in the introduction section include some constraint, either in the grammatical rules (nativist approaches) or in the procedure that they use in order to acquire the language (cultural models). Maybe it can be argued that the evolutionary process could be replaced by another bottom-up mechanism, but results show that it performs in all the experiments and it opens the door to future improvements.

Figures 9 and 10 reveal how the language and grammar evolution process is very similar in all cases, whatever the team size is, and they show that the model works equally on each agent in the team and the global process is quite steady. Initially the best chromosome's fitness is low as it is usual in evolutionary algorithms. Then, it improves substantially for a few generations. Again, it must be remembered that the best chromosome's fitness in the figures is an average value, which is computed from the best chromosome's fitness for each agent. An average value allows us to measure the concept in global terms, that is, from the team's viewpoint instead of each agent's viewpoint. According

to the results, the best chromosome's fitness is close to the optimum value, so we can argue that the team comes near syntactic agreement in a short time. In most experiments only one or two agents in the team found difficulties at evolving their best solution grammars and thus final consensus is not achieved.

We finish this section with a brief reflection about the fitness. Previously, we argued that the fitness value is an external value because it is measured during a language game which is considered an external action (to the agent). On the other hand, in a language game only the sentences uttered by each agent are evaluated. This means that the grammar is indirectly evaluated. We think that this explanation can justify the reason why we study the fitness as a measure about the grammar and language evolution process. As we studied in the previous section, when syntactic agreement is achieved, all grammars are equivalent in the sense that all of them contain at least an equivalent production rule. Grammar and language are obviously connected in this model and best chromosome's fitness evolution shows indirectly how the best solution grammar also improves. To summarize, fitness results account for a steady progress associated to the grammar and language evolution. Each agent in the team evolves a population of chromosomes, which are transformed into solution grammars, which finally are the device where each sentence participating in a language game is generated from.

7 A New Level of Linguistic Description

Previous sections have described and analyzed the model in a basic problem, where only simple sentences with the same syntactic structure can be evolved and the main difficulty was the word order. In this section we will study whether the model can be scaled to more complicated problems. We define a new task where we suppose that objects can appear in singular or plural, that is, in the scene the agents can watch several objects of the same type. For example, based on Figure 1, we will try now to evolve a language to describe spatial relationships such as:

1. *the chairs are on the right of the wardrobe*,
2. *the wardrobe is on the left of the chairs*,
3. *the chairs are in front of the table*,
4. *the table is behind of the chairs*,
5. *the table is on the right of the wardrobe*,
6. *the wardrobe is on the left of the table*,
7. *the stools are on the left of the chairs*,
8. *the chairs are on the right of the stools*

To simplify, we have considered plurality in the group of object, so we suppose that *"all the chairs are on the right of the wardrobe"* instead of having some chairs on the right and other chairs on the left. As we can see, we have a new object *stools* in the vocabulary about objects while the vocabulary for the spatial relations remains the same.

In this problem, different syntactic structures are involved in the same referential scene, that is, we have linguistic situations that need different phrase structures to describe them. Besides, a new category is needed in order to represent the numerical property. Therefore, the model must include this knowledge in some way but we only need some minor changes in order to deal with this problem. Thanks to the concept of meta-grammar, we only need to modify the rules in the meta-grammar to allow the model to develop new structures. Appendix B shows the new meta-grammar's rules. Essentially, the meta-grammar keeps the production rules to evolve the previous syntactic structures but it adds new production rules to allow the model to generate new structures. Now, it is possible to evolve two syntactic levels instead of one as the previous meta-grammar did. We clarify this with an example. In the basic problem the meta-grammar could only evolve solution grammars which, in turn could generated a single valid structure (i.e. a structure with three words: two objects and a spatial relation). Of course, a solution grammar could contain other production rules to generate other kind of syntactic structures, but they will not reflect the linguistic situation that the basic problem required. For example, a meta-grammar could generate sentences with only two objects or four objects and a spatial relation but those structures would be wrong. In fact, the intentional consideration in the model constrained these possibilities and the evolutionary process usually discarded them.

If we look at the new meta-grammar, we can see how the description of an object is now more sophisticated. An object can appear in singular or in plural. If the object is in singular we do not use any mark, but if the object appears in plural we add the category *aDeterminer* to the object. Comparing old and new meta-grammars, the production rule for the non-terminal <*object-def*> is different. In the basic meta-grammar this symbol was defined as:

```
<object-def> ::= <object>
```

While in the new meta-grammar is defined as:

```
<object-def> ::= <object>
             | '<determiner>' <object>
```

This way, the evolutionary algorithm can evolve different types of syntactic structures which allow to describe situations with objects in singular or plural. For example, in the new linguistic situation, expressions one and two include *chairs* in plural and a *wardrobe* in singular. A similar situation is considered in expressions three and four. Expressions five and six include both a *table* and a *wardrobe* in singular. Finally, expressions seven and eight refer to a *chairs* and a *stools* both in plural. On the other hand, the evolutionary process is affected because the number of alternatives associated to the production rule for the non terminal <object-def> is duplicated. This implies more combinations to analyse during the translation process in the evolutionary stage.

7.1 Experimental Work and Results

In this new task there are eight different linguistic situations in the referential scene and different phrase structures are needed in order to describe all the possibilities. We work with teams of 5, 10, 20 and 30 agents and we ran 30 executions in each case. Evolutionary parameters are essentially equal those of Table 3 but we limit the number of individuals in the population to 100 in order to reduce execution time. Intentions must reflect the new communicative goals. We do not reproduce here figures about the evolution of fitness because they are essentially similar to the basic problem. We show in Figure 11 the success probability measures grouped by team size. We use average values as we did in the basic problem.

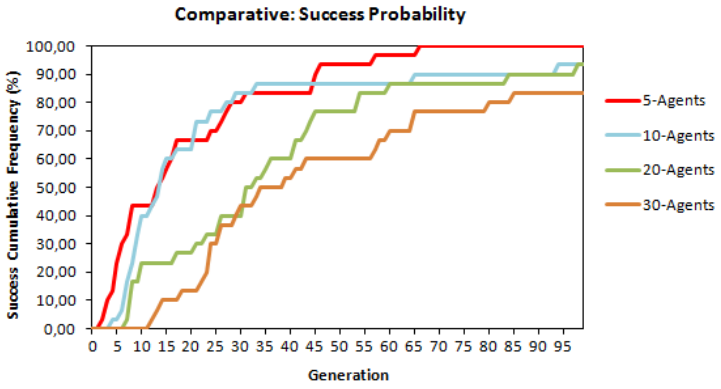

Fig. 11 Success Probability in a Task with Different Syntactic Structures

As we can see in Figure 11 all the experiments with 5 agents were successful and the syntactic agreement was reached. Bigger teams show the difficulty of the task. With 10 and 20 agents the model achieves 93.3% of success probability and it failed in two experiments on each configuration. Finally, in a team of 30 agents the success probability is 83.3%. Five experiments failed in this case. All these experiments failed because a single agent in the team evolved a different grammar to the rest of the team, so the syntactic agreement was not possible. In two experiments with 10 agents, two experiments with 20 agents and five experiments with 30 agents, there was an agent who disagreed and the syntactic agreement failed. We will analyze what really happened in one of these failed experiments. For example, in the fifth experiment with 10 agents, nine of them evolved a solution grammar like the following:

```
<sentence>::= <spatialrel> <object> <object> | .......
<object>::= <determiner> anObject | anObject | .......
<spatialrel>::= aSpatialRelation
<determiner>::= aDeterminer
```

But the sixth agent evolved this solution grammar:

```
<sentence>::= <object> <object> <spatialrel> | <object>
<object>::= anObject | <determiner> anObject
<spatialrel>::= aSpatialRelation
<determiner>::= aDeterminer
```

Let us take a look to the first production rule for the solution grammar's axiom. Nine agents had a rule that allowed them to generate a sentence with the spatial relation as first word and the objects following the spatial relation. However, the sixth agent had a rule that allowed the agent to build sentences with the spatial relation at the end. Both kinds of solution grammars are suitable to the linguistic descriptions in the referential scene but obviously the whole syntactical agreement is not possible in this case. Agents evolved a grammar but nine of them were not able to understand the sixth agent.

We can mention two possible solutions to this problem: 1) the evolutionary parameters can be tuned (more generations or a bigger population size among others) and 2) the concept of syntactic agreement can be slightly relaxed, by considering a majority of agents, instead of all of them. In any case, we think that model performs reasonable well in this more complicated task.

Now, we will analyze successful experiments by studying the type of evolved solution grammar and the syntactic descriptions that the agents finally agreed on. To do this we review an experiment in a group of 5 agents. Table 7 shows results in the third experiment of this group.

All the agents evolved a production rule to generate sentences with the spatial relation as first word and the objects following the relation. Of course, other production rules can be evolved and some of them are duplicated or they are wrong for this communicative intention. Again, the remarkable issue here is how the model allows the agents to evolve a grammar that they eventually can share. In this new task, we can see more complicated production rules than we saw in the basic problem. These grammars can evolve different syntactic structures. Specifically, objects can appear in singular or plural in this case.

We finish this section showing the final result of the example in terms of syntactic agreement in Table 8.

Table 8 is equivalent to Table 6 for the basic problem. Nevertheless, there is an important difference. While expressions for the basic problem always contained three words, the expressions in this new task depend on the number of the objects in the referential scene, so the agents use the word *these* when the objects appear in plural and they do not use any mark when the object is single. Communicative intention helps to decide between both concepts. The word *these* belongs to the *aDeterminer* category and this category is associated to the *meaning number* as we can see in the left column in Table 8. As we defined in Table 1, we use simplified functions to deal with the concept of meaning.

Results in other experiments are equivalent to this case, so we can argue that model is able to develop different kind of syntactic structures inside the same referential scene and what is more important in our opinion, syntactic

structure shows the agent's communicative intention that is internally represented as a meaning which is derived from the watched scene. As we have commented, we have simplified this internal representation and we have focused in the higher layer, that is in the symbolic processing.

8 Conclusions

In this work, we propose a model so that a team of artificial agents can autonomously evolve a shared language with syntactic properties, in order to describe scenes with spatial relations between objects. The proposed model combines an evolutionary algorithm that is able to evolve grammars (constrained by a meta-grammar) and a planning process that can generate sentences from a grammar, where the agent's informative intention is used to avoid generating irrelevant sentences. Agents are encouraged to progressively agree on a shared language by playing language games, where sentences are uttered, and agreements are used as reward. We think that the combination of intentionality by planning with the use of the grammatical formalism, are the major contributions of our model. It is important to remark that the grammar rules themselves are not fixed in advance, but they emerge from the interactions between the agents.

The proposed model has been first applied to describe several simple spatial situations in a basic referential scene. Experiments have been carried out with different team sizes (from 5 to 50 agents). Results show that syntactic agreement between the agents is reached in all experiments in a reasonable time (although it takes loner for larger teams). Second, a more complicated problem where objects can appear in singular or plural has been tested with teams of 5, 10, 20, and 30 agents. Different syntactic structures appear in the referential scene but the model is able to achieve syntactic agreement in most cases, although it becomes more difficult as team size increases. But even in those cases, agreement is almost reached, except for one of the agents.

In future work, we will study the important problem of grounding and how to link effectively the semantic with the syntactical framework that the model proposes. We will also consider other stages in the language acquisition and evolution process, such as the invention or creation of new syntactical structures and the recognition of other agents sentences.

A Meta-grammar Definition

We use the usual BNF notation to specify the grammatical production rules and we adopt here the formalism proposed in [18]. In short, according to this convention, terminal symbols between quotation marks stand for non-terminal symbols in the solution grammar level which will be created starting from the following meta-grammar rules.

```
<grammar> ::= '<sentence>::=' <rel-expression-def> /
              '<object>::=' <object-def> /
              '<spatialrel>::=' <spatialrel-def>
```

```
<rel-expression-def> ::= <rel-expression>
                  | <rel-expression> '|' <rel-expression-def>
<object-def> ::= <object>
<spatialrel-def> ::= <spatialrel>
<rel-expression> ::= <rel-word>
               | <rel-word> <rel-expression>
<rel-word> ::= '<object>'
         | '<spatialrel>'
<object> ::= anObject
<spatialrel> ::= aSpatialRelation
```

where terminals $'<sentence>::='$, $'<object>::='$, $'<spatialrel>::='$, $'<object>'$ and $'<spatialrel>'$ will represent non-terminals in a solution grammar as we showed in 5.1 and the symbol $'|'$ stands for the alternative symbol as it is usual in BNF notation. Finally, terminals *anObject* and *aSpatialRelation* represents lexical categories such as nouns, adjectives or verbs but they are constrained to a spatial context.

B Extended Meta-grammar Definition

This meta-grammar is an extension of the previous one. It allows to develop more complicated languages.

```
<grammar> ::= '<sentence>::=' <rel-expression-def> /
              '<object>::=' <obj-expression-def> /
              '<spatialrel>::=' <spatialrel-def> /
              '<determiner>::=' <determiner-def>
<rel-expression-def> ::= <rel-expression>
                  | <rel-expression> '|' <rel-expression-def>
<obj-expression-def> ::= <object-def>
                  | <object-def> '|' <obj-expression-def>
<object-def> ::= <object>
                  | '<determiner>' <object>
<spatialrel-def> ::= <spatialrel>
<determiner-def> ::= <determiner>
<rel-expression> ::= <rel-word>
               | <rel-word> <rel-expression>
<rel-word> ::= '<object>'
         | '<spatialrel>'
<object> ::= anObject
<spatialrel> ::= aSpatialRelation
<determiner> ::= aDeterminer
```

References

1. L. Steels, Loetzsch, M. Perspective Alignment in Spatial Language. In K. R. Coventry, T. Tenbrink and J. Bateman. Spatial Language and Dialogue. Oxford University Press, 2009.
2. M. F. Schober. Spatial Dialogue between Partners with Mismatched Abilities. In K. R. Coventry, T. Tenbrink and J. Bateman. Spatial Language and Dialogue. Oxford University Press, 2009.
3. L. Steels. A Self-Organizing Spatial Vocabulary. Artificial Life, 2:319-332, 1996.

4. J. de Lara, M. Alfonseca. Some Strategies for the Simulation of Vocabulary Agreement in Multi-Agent Communication. Journal of Artificial Sociesties and Social Simulation, volume 3, number 4, 2000.
5. D. Maravall, J. de Lope, R. Domínguez. Coordination of Communication in Robot Teams by Reinforcement Learning. Robotics and Autonomous Systems, 61(7), 661-666, July 2013.
6. D. Maravall, J. de Lope, R. Domínguez. Self-emergence of a Common Lexicon by Evolution in Teams of Autonomous Agents. Neurocomputing, 75, 106-114, 2012.
7. L. Steels. The origins of Syntax in Visually Grounded Robotic Agents. Artificial Intelligence, Volume 103, Issues 1-2,133-156, 1998.
8. J. de Beule, J. V. Looeveren, W. H. Zuidema. Grounding Formal Syntax in an Almost Real World. Presented at the Belgium-Netherlands Artificial Intelligence Conference, 2002.
9. M. Spranger, L. Steels. Emergent Functional Grammar for Space. In Luc Steels (Ed.), Experiments in Cultural Language Evolution, 207-232. Amsterdam: John Benjamins. 2012.
10. L. Steels, J. de Beule. A (very) Brief Introduction to Fluid Construction Grammar. Third International Workshop on Scalable Natural Language Understanding, June 8, 2006.
11. R. Schulz, A. Glover, M. J. Milford, G. Wyeth, J. Wiles. Lingodroids: Studies in Spatial Cognition and Language. IEEE International Conference on Robotics and Automation, May 9-13, Shanghai, China, 2011.
12. J. L. Austin. How to do things with words. J. O. Urmson (Eds.), Oxford University Press, 1962.
13. J. Searle. Speech Acts: An essay in the philosophy of language. Cambridge: Cambridge University Press, 1969.
14. J. A. Allen. A Plan-Based Approach to Speech Act Recognition. Ph.D. Thesis, Technical Report No 131/79, Dept. of Computer Science, University of Toronto, January, 1979.
15. P. R. Cohen, C. R. Perrault. Elements of a Plan-Based Theory of Speech Acts. Cognitive Science 3, 177-212, 1979.
16. L. Wittgenstein. Philosophical Investigations. New York. Macmillan. 1953.
17. J. J. Collins, C. Ryan, M. O'Neill. Grammatical Evolution: Evolving Programs for an Arbitrary Language. Lecture Notes in Computer Science 1391, Proceedings of the First European Workshop on Genetic Programming. Springer-Verlag, 83-95, 1998.
18. M. O'Neill and C. Ryan. Grammatical Evolution by Grammatical Evolution: The Evolution of Grammar and Genetic Code. In Maarten Keijzer, Una-May O'Reilly, Simon M. Lucas, Ernesto Costa, and Terence Soule, editors, Genetic Programming 7th European Conference, EuroGP 2004, Proceedings, volume 3003 of LNCS, pages 138-149, Coimbra, Portugal, 5-7 April 2004. Springer-Verlag. ISBN 3-540-21346-5.
19. S. C. Levinson, D. P. Wilson. The Background to the Study of the Language of Space. In S. C. Levinson and D. P. Wilson editors. Grammars of Space. Explorations in Cognitive Diversity. Cambridge University Press, 2006.
20. R. Fikes, N. Nilsson. STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving. Artificial Intelligence, 2(3/4):189-208, 1971.
21. Russell, S., Norvig, P., Artificial Intelligence: A Modern Approach. Third Edition, Prentice Hall, 2010.
22. E. A. P. Hemberg. An Exploration of Grammars in Grammatical Evolution. PhD Thesis, University College Dublin. September, 2010.
23. C. S. Wetherell. Probabilistic Language: A Review and Some Open Questions. ACM Computing Surveys (CSUR), 12(4):361-379, 1980.

Table 7 Evolved solution grammars in a Task with Different Syntactic Structures (5-agents)

Agent ID	Evolved Grammar
Agent_0	<sentence>::= <object> <object> | <spatialrel> <object> <object> <object>::= anObject | <determiner> anObject | <determiner> anObject <spatialrel>::= aSpatialRelation <determiner>::= aDeterminer
Agent_1	<sentence>::= <spatialrel> <object> <object> | <spatialrel> <object> <object>::= anObject | anObject | <determiner> anObject | anObject <spatialrel>::= aSpatialRelation <determiner>::= aDeterminer
Agent_2	<sentence>::= <spatialrel> <object> <object> <object>::= anObject | <determiner> anObject | anObject <spatialrel>::= aSpatialRelation <determiner>::= aDeterminer
Agent_3	<sentence>::= <spatialrel> <object> <object> <object>::= anObject | <determiner> anObject | anObject <spatialrel>::= aSpatialRelation <determiner>::= aDeterminer
Agent_4	<sentence>::= <object> | <spatialrel> | <object> <object> <spatialrel> <object> <object> <spatialrel> | <spatialrel> <object> <object> <object>::= anObject | <determiner> anObject | <determiner> anObject | anObject <spatialrel>::= aSpatialRelation <determiner>::= aDeterminer

Table 8 Results of the syntactic agreement in a Task with Different Syntactic Structures (5-agents)

Informative intention	**Linguistic expression**
meaning(right(meaning(number(meaning(chair)));meaning(wardrobe))))	right wardrobe these chair
meaning(left(meaning(wardrobe);meaning(number(meaning(chair))))	left wardrobe these chair
meaning(front(meaning(number(meaning(chair)));meaning(table))))	front table these chair
meaning(behind(meaning(table);meaning(number(meaning(chair))))	behind table these chair
meaning(right(meaning(table);meaning(wardrobe)))	right table wardrobe
meaning(left(meaning(wardrobe);meaning(table)))	left wardrobe table
meaning(left(meaning(number(meaning(stool)));meaning(number(meaning(chair))))	left these stool these chair
meaning(right(meaning(number(meaning(chair)));meaning(number(meaning(stool))))	right these chair these stool