

Emergence of Hierarchy in Fluid Construction Grammar

Luc Steels^{1,2} and Joris Bleys²

¹ SONY Computer Science Laboratory - Paris

² Artificial Intelligence Laboratory, Vrije Universiteit Brussel
Pleinlaan 2 - B-1050 Brussel

Abstract. One of the key properties of (natural) languages is that they are hierarchical. Phrases combine into larger phrases eventually covering complete sentences. The semantics of each phrase combine to form the complex meaning of the whole. A key question in explaining the origins and evolution of language is therefore how such hierarchical structures may originate and spread in a population. This paper reports on breakthrough experiments in which unbounded hierarchical structure can emerge as a side-effect of repair strategies in a population of agents.

1 Introduction

The research reported in this paper is part of a growing body of research that tries to show through careful computational and robotic experiments how communication systems with properties similar to those of human natural languages may emerge in populations of agents (see recent overviews in [6], [12], a.o.). Many aspects of language are being studied, ranging from the origins of sound systems, the origins of lexicons, the co-evolution of lexicons with ontologies usable for categorisation [13], etc. In this paper we focus on issues related to grammar, and in particular on the issue of hierarchy. Prior work exists in this area (see e.g. [5], [9], [1], [8]) but it is fair to say that so far no convincing demonstrations have been produced in which *natural language like* constituent structure arises in co-evolution with semantic compositionality using a world model grounded in (visual) perception and action. This paper reports recent experiments focusing on this question.

We adopt the general framework of the embodied language game research also used in earlier research of our team [10], which can be summarised as follows: (1) We look at the language problem in its entirety, comprising perception and action, conceptualisation, and verbalisation as part of production; parsing, interpretation, and world modelling through perception and action as part of language understanding. (2) Speakers and hearers align their inventories at all levels (phonetic, lexical, morphological, grammatical, conceptual). This leads to self-organised convergence of a shared inventory. (3) New linguistic structure is invented by speakers and adopted/imposed by hearers based on their communicative needs and on the hope to gain in cognitive economy (the least

effort principle). In this paper the above principles are applied to the question of the emergence of hierarchy. We focus in particular on how speakers can invent new hierarchical structure and how hearers can infer them. The hierarchy is semantically motivated, instead of being motivated by more efficient coding or overcoming a transmission bottleneck. Agents evolve not only the syntactic structures but also the semantic composition rules for hierarchical units.

The experiments in this paper rest on highly sophisticated technical tools contributed by many other members of our team. Lexicon and grammar use the Fluid Construction Grammar (FCG) framework, which is a new HPSG-style implementation of construction grammar, [15]. The technical part of this paper assumes some familiarity with FCG, and in particular the way that hierarchy is handled using the J-operator (see [4]). Semantic aspects are handled through grounded procedural semantics based on a constraint language called the Incremental Recruitment Language (IRL) (see [14]). Both systems are combined in a powerful experimental framework called Babel2¹.

2 Background Assumptions

Grounded Semantics We assume that in situated dialogue, agents maintain a non-symbolic world model by the operation of their sensori-motor apparatus. This world model is analog to and based on direct output of sensors and actuators. Often it is assumed that there is a simple straightforward way to transform a non-symbolic world model into a categorial situation model, which is a representation of the world in the form of facts in some variant of predicate calculus. Conceptualisation of what to say by the speaker is then equal to selecting some part of this situation model and interpretation as matching the meaning reconstructed by the parser. This assumption has also been made in earlier work on the emergence of grammar, such as [1] or [8].

We instead adopt an alternative ‘grounded semantics’ view, which resonates with earlier work on procedural semantics in AI [17]. The meaning of a phrase is not an expression to be matched against a situation model, but a program to perform the necessary categorisations and conceptualisations in order to achieve specific communicative goals like reference. Conceptualising what to say now becomes a planning process and interpretation becomes equal to running programs reconstructed from parsing a sentence. We call these meanings ‘semantic programs’. The situation model then arises by storing (caching) the results of semantic programs but this is not even necessary. We have opted for a constraint propagation language and use our own implementation IRL (Incremental Recruitment Language) based on a LISP substrate.

Construction Grammar The key question then is how these IRL constraint networks are mapped in a systematic and incremental fashion onto language utterances. We adopt the perspective of construction grammar [3], [2] and use our

¹ An implementation of Babel2 based on a LISP substrate has been released for free download through <http://www.emergent-languages.org>.

Fluid Construction Grammar (FCG) framework for the implementation. Construction grammar assumes that every rule in the grammar has both a semantic and a syntactic pole. This contrasts with a (generative) constituent structure grammar which specifies only syntax. Semantics is supposed to be defined separately by translation rules. The semantic pole of a construction specifies how meaning has to be build up in parsing or decomposed in production, and the syntactic pole how the form has to be analysed in parsing or built in production. A unique feature of FCG is that rules are truly bi-directional. The same rule can be used both for parsing and production, even if it involves hierarchy [4]. The syntactic and semantic structure being built during parsing and production takes the form of feature structures, and unify and merge are the basic operations that are used for expanding these feature structures through the application of rules, similar to the widely used HPSG frameworks [7].

3 Starting from Scratch

Assuming now an open-ended system for planning a constraint network to satisfy specific communicative goals, how can lexical and grammatical constructions arise that express these constraint networks? In other words, what repair strategies should speakers and hearers use so that they end up with an effective and co-ordinated lexicon-grammar. Repair strategies of speaker and hearer are almost identical. The speaker repairs either during production when there are uncovered items, or during re-entrance to fix variable equalities. The hearer reconstructs first how he would express something himself (as much constrained as possible by what he has understood from the speaker) and then adds the necessary rules to absorb the missing elements of the network and the word-order used by the speaker. In this section, we look at the direct expression of constraints *de novo*, and later look at re-use.

The main ideas behind the repair strategies will be illustrated with English-like examples, but these should be taken as examples only. The artificial languages that the robotic agents develop are not like English as they use a self-invented lexicon and a self-invented morpho-syntax. This section uses the following example phrase: “ball”. Its grounded procedural semantics is to filter the set of objects in the current context and retain those that are similar to the prototype of a ball. There should normally be only one element remaining, and this is the one which is the referent of this expression. In IRL notation the constraint network achieving this is as follows (note that there is no control ordering implied):

```
[1] (external-context s1)           ; set s1 to current context
[2] (filter-set-prototype s2 s1 p1) ; retain elements of s2 of p1
[3] (prototype p1 [BALL])          ; declare prototype to be used
[4] (unique-element o1 s2)          ; pick out unique element from s2
```

Suppose this is the first challenge given to the language system and there are no prior rules yet. There are three repair strategies (R-strategies) becoming active, each possibly producing rules:

[R-strategy 1.1] *Handle a new semantic entity* A semantic entity is an entity from the ontology used by a (primitive) constraint, such as a category, a prototype, a relation, etc. They have a name (local to a constraint program as above) which may be bound to a variable, as well as an identifier which directly refers to the item in the (agent’s local) ontology. These identifiers are constructed when the entity itself is constructed. When a new semantic entity has never been encountered before, a new word is invented (by the speaker) or adopted (by the hearer). This is the case in this example for [3] which introduces a new prototype. This semantic entity is captured in a lex-stem rule which relates it to this word. Additionally, this rule also assigns a (possibly entirely new) lexical category to the word (in this case “Noun”). The lexical rule for “ball” as well as all other rules we need are shown in figure 1.

[R-strategy 1.2] *Handle function of semantic entity* The function of a semantic entity is a constraint that introduces this entity to the network by taking it as a direct argument. This is the case for [2] in the above example. It uses the prototype for filtering s1 to yield s2. A new construction is made that has a sub-unit for a semantic entity of a specific type (for example prototype) and of the lexical category associated with this type (for example Noun). The parent node contains the constraint (for example **filter-set-prototype**) that uses this entity. The slots in the constraint that are not delivered by sub-units form part of the link of the parent so that when the constituent is used as a building block in more complex structure, the necessary elements can be passed (in the example this is the case for ?s1 and ?s2).

[R-strategy 1.3] *Handle surrounding context* The uncovered constraints that do not involve semantic entities are now grouped together to form the remaining overarching construction (this is the case here for [1] and [4]). This construction contains sub-units for each of the constraints that were handling semantic entities (in this case only one with the Noun constituent). It introduces a new constituent (here the NounPhrase constituent), whose link includes variables for the items that will be supplied externally (in this case ?o1). The variables that are delivered by the sub-units are excluded from this link (in this case ?s1 and ?s2). Most importantly, this rule also specifies the variable equalities between the different sub-units and the constraints in the rule itself using the links of the sub-units (for example the first argument of **external-context** should be equal to the second argument of **filter-set-prototype**). Figure 2 shows the structure that is built by all these rules.

It is entirely possible of course that there are already rules for each of these cases so that the corresponding repair strategy is not necessary and the existing rules provide information for the others. Thus when the NounPhrase construction already exists, it will specify the type of constituent to be used by [R-strategy 1.2]. Or if there is already a Noun construction, it will specify what the lexical category is of the newly to be invented word for use by [R-strategy 1.1].

A similar analysis can be made for a slightly more challenging example for phrases like “a ball”. For these phrases, the communicative goal of the speaker

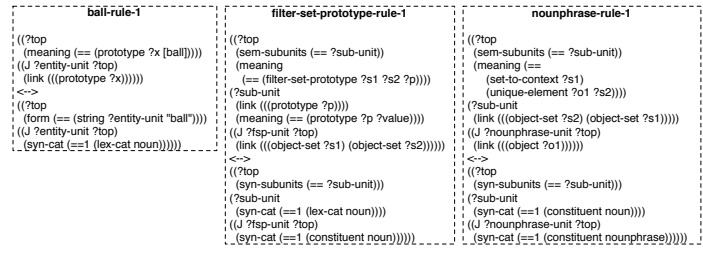


Fig. 1. Left: lexical rule for “ball”. Middle: functional construction rule for nouns. Right: contextual construction for simple noun-phrases.

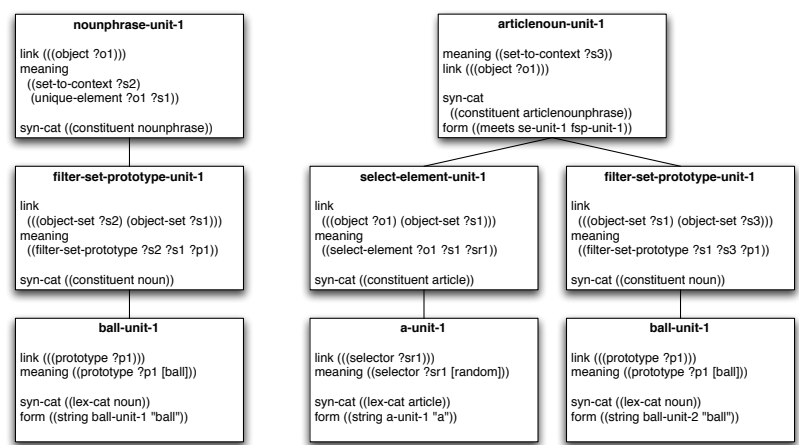


Fig. 2. Left: The hierarchical structure for “ball”. Right: Hierarchical structure for “a ball”.

is satisfied when the hearer is able to identify one of several objects which all are similar to a certain prototype. The corresponding semantic constraint network is similar to the one discussed above, except for the phase in which the referent is selected from set s2 ([4] and [5]). Instead of expecting a unique element that satisfies this test, the semantic network is satisfied when the referent is a member of the intermediate set s2.

- [1] (external-context s1) ; set s1 to current context
- [2] (filter-set-prototype s2 s1 p1) ; retain elements of s2 of p1
- [3] (prototype p1 [BALL]) ; declare prototype to be used
- [4] (select-element o1 s2 sel1) ; select element from s2
- [5] (selector sel1 [RANDOM]) ; declare selector to be used

A new lexical rule is created which associates a new stem “a” with the selector [RANDOM] by [R-strategy 1.1] and a new lexical category Article. [R-strategy 1.2] invents a new utilisation construction rule which introduces the semantic entity to the semantic network. All remaining constraints are grouped together

by [R-strategy 1.3] which invents a contextual ArticleNounPhrase construction. This construction covers all these constraints (in this case only [1]) and specifies variable equalities between the different constraints in the semantic network using the links of the sub-units. As described above, the link of the new constituent in this rule is calculated by propagating the unknown variables upwards (in this case only o1), ready to be used as part of a bigger structure. Additionally, this rule also contains word-ordering constraints on the syntactic structure: the forms specified below the Article unit should precede the forms below the Noun-unit and no other forms should be in-between. The rules for this example are depicted in figure 3 and the corresponding structure is shown in figure 2.

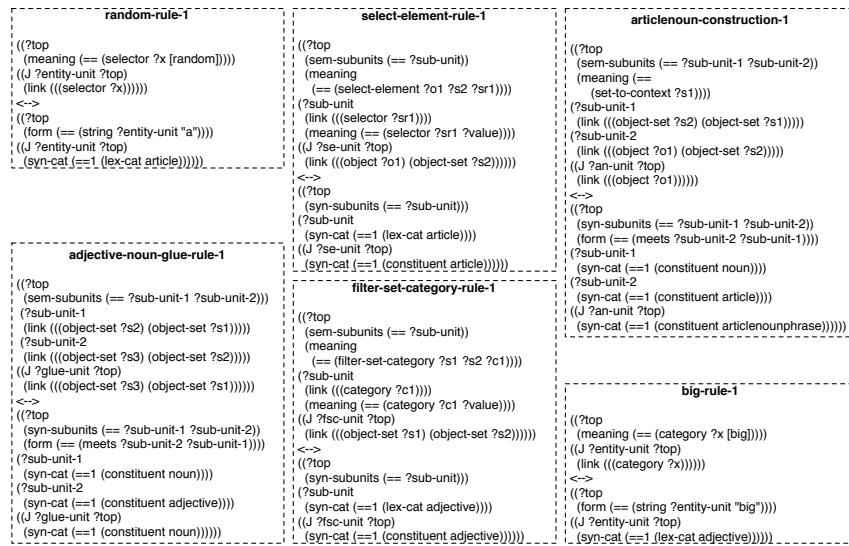


Fig. 3. Top (left to right): lexical rule for “a”; functional construction rule for articles; contextual construction for article-nounphrases. Bottom (left to right): article-noun construction; functional construction rule for adjectives; lexical rule for “big”.

4 Reusing existing constructions

Re-use occurs when there is a construction which covers part of the meaning to be conveyed. This might be the case for example, when the following constraint program is offered by the conceptualisation component for verbalisation, which reflects the grounded procedural meaning of “a big ball”.

- [1] (external-context s1) ; set s1 to the current context
- [2] (filter-set-prototype s2 s1 p1) ; filter s1 for matches in s2
- [3] (prototype p1 [BALL]) ; set the prototype

```

[4] (filter-set-category s3 s2 c1) ; filter s2 with c1
[5] (category c1 [BIG])           ; set the category
[6] (select-element o1 s3 sel1)   ; select element from s3
[7] (selector sel1 [RANDOM])       ; set the selector

```

[1], [2], [3], [6] and [7] are already covered by rules produced by the strategies discussed in the previous section, so it is highly desirable to re-use all the lexical and grammatical rules already built up as much as possible. [4] and [5] will be handled with the repair strategies [R-strategy 1.1] and [R-strategy 1.2]. They will yield a new word string “big” with lexical category Adjective and an Adjective constituent.

[R-strategy 2.1] The strategy that handles re-use starts from the best matching construction (in this case the ArticleNounPhrase construction) and tries to detect what prevented it from being applicable (in this case an extra unit for the adjective which also corrupts the expected variable equalities). Next it tries to construct a rule which removes these obstructions. This could be achieved by adding an extra unit which combines two units (for example the Adjective and the Noun) into one and also taking care of the variable equalities between the units that are glued together. The link of this additional unit is calculated as before, but as the links of the units might share some variables, they should also be excluded from the link. This rule also contains word-order constraints, which are either invented (in case of the speaker) or deduced from the utterance (in case of the hearer). The syntactical category (Noun) is derived from the best matching construction. A possible resulting construction for the above example is shown in figure 3. If no such rule could be found the problem is handed to [R-strategy 1.3] which will create a completely new rule without re-use. All other rules remain as is, but we now get hierarchical structure as shown in figure 4.

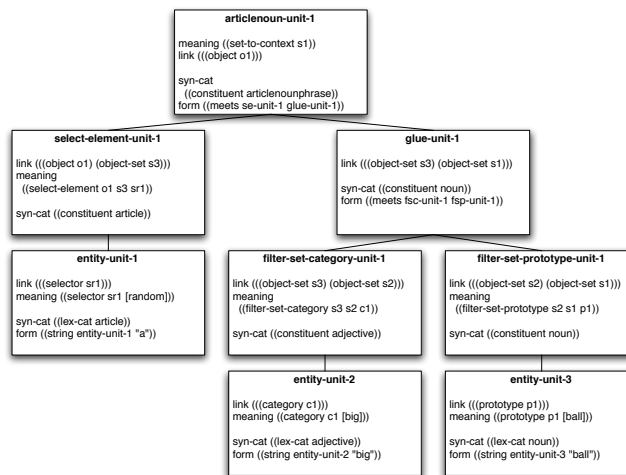


Fig. 4. The hierarchical structure for “a big ball” .

This repair-strategy automatically gives rise to unbounded hierarchical structure, including recursion (in the strict sense of a tree where a node of a simpler type reoccurs as daughter). An easy way to show this is to consider an example in which two filtering operations are used sequentially in order to restrict the referents even further. The only difference in comparison to the previous example is the fact that the AdjectiveNoun rule has to be applied twice before the ArticleNounPhrase construction becomes applicable.

5 Experiments

The repair strategies discussed in this paper have been implemented and explored in multi-agent simulations. The experiment is scaffolded in the sense that the world and the conceptualisation component are by-passed. This way, we are able to focus on the development of the repair strategies in an ideal setting.

In the experiment, the complexity of the semantic programs offered to the language component is controlled by the experimenter² and is divided in learning stages. Each learning stage extends the previous learning stage by a new challenging constraint network. The first four learning stages are similar to the different examples discussed above. The fifth one incorporates a different filtering operation which modifies the category it uses (similar to an Adverb as in “a very big ball”) and the sixth one extends this by another filtering operation. The next stage introduces another primitive constraint which constrains objects based on a relation between them (like for instance “a ball in a box”). The final learning stage extends this by incorporating another filtering operation.

The graph depicting the dynamics of the experiment is shown in figure 5. The size of the population is 10. Each time the learning stage is increased, the language of the agents experience a period of turbulence after which the communicative success of the agents stabilizes on perfect communicative success. At each stage in which new semantic entities are used by the semantic constraint networks, the lexicon sizes of the agents overshoot before settling down on a one-on-one mapping between semantic entities and lexical entries. On the level of grammatical rules one can observe a similar pattern but this should be interpreted as the agents trying to reach a consensus on which word-order they will use to express a certain combination of syntactical categories (for example putting the Article before or after the Noun). One can also observe the agents are not inventing new grammatical rules for each level of complexity which points to the fact that the agents are truly re-using their grammatical knowledge.

6 Conclusions

The paper briefly discussed recent experiments to explore how hierarchy can emerge in a self-organised language by language games among agents. The pro-

² Ideally, this parameter should be under control of the agents themselves but as this has already been convincingly investigated in another experiment [16], it is not of our immediate interest.

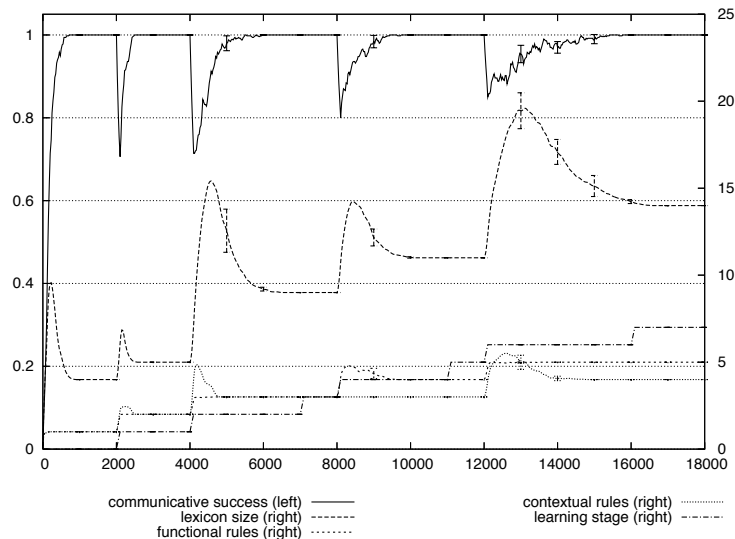


Fig. 5. Resulting dynamics in a multi-agent experiment. Increasing the learning stage puts the emergent languages under stress which dissolves when the agents reach both a lexical and a grammatical consensus.

cess is entirely driven by communicative need and the least effort principle. The meaning of an utterance is not seen as an expression to be matched against other ‘symbolic’ expressions in a situation model, but as a semantic program in the form of a constraint network. The semantic program operates directly over the world model coupled to the agent’s perception and action systems. Agents invent new constructions when part of the meaning is uncovered and they reuse existing constructions as much as possible in order to keep the size of the inventory as small as possible. We do not claim that these forces are the only ones that play a role in the emergence of grammar with hierarchical structure but they clearly play a role and are highly effective. We are currently engaged in more extensive experiments to scale up the repair strategies introduced here and to ground the agents fully, and study in particular the generality of the repair strategies and the integration of sub-goals in the semantic constraint networks.

7 Acknowledgments

The research reported here has been conducted at the Artificial Intelligence Laboratory of the Vrije Universiteit Brussel (VUB) and at the Language Group of the Sony Computer Science Laboratory in Paris. The authors are indebted to all members of both laboratories for their insightful comments and suggestions. Additional funding for the Sony CSL activities has come from the EU FET-ECAgents project 1170. Joris Bleys is financed through a fellowship of the

Institute of the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen).

References

1. Batali, J.: The negotiation and acquisition of recursive grammars as a result of competition among exemplars. In Ted Briscoe, editor, *Linguistic Evolution through Language Acquisition: Formal and Computational Models*. Cambridge University Press, Cambridge (2002)
2. Bergen, B.K. and Chang, N.C.: Embodied Construction Grammar in Simulation-Based Language Understanding. In: Ostman, J.O. and Fried, M. (eds): *Construction Grammar(s): Cognitive and Cross-Language Dimensions*. John Benjamin Publ Cy., Amsterdam (2003)
3. Croft, W. A.: *Radical Construction Grammar; Syntactic Theory in Typological Perspective*. Oxford University Press, Oxford (2001)
4. De Beule, J. and Steels, L.: Hierarchy in Fluid Construction Grammar. In Furbach U., editor, *Proceedings of KI-2005*, pages 1–15. Springer-Verlag, Berlin (2005)
5. Hashimoto, T. and Ikegami, T. (1996) Emergence of net-grammar in communicating agents. *Biosystems*, 38(1):1–14.
6. Minett, J. W. and Wang, W. S-Y.: *Language Acquisition, Change and Emergence: Essays in Evolutionary Linguistics*. City University of Hong Kong Press, Hong Kong (2005)
7. Pollard, C. and Sag, I.: *Head-driven phrase structure grammar*. University of Chicago Press (1994)
8. Smith, K., Kirby, S., and Brighton, H.: Iterated Learning: a framework for the emergence of language. *Artificial Life*, 9(4):371–386 (2003)
9. Steels, L.: The origins of syntax in visually grounded robotic agents. *Artificial Intelligence*, 103(1-2):133–156 (1998)
10. Steels, L. Evolving grounded communication for robots. *Trends in Cognitive Science*, 7(7):308-312 July 2003 (2003)
11. Steels, L.: Constructivist Development of Grounded Construction Grammars Scott, D., Daelemans, W. and Walker M. (eds) *Proceedings Annual Meeting Association for Computational Linguistic Conference*, p. 9–19. Barcelona (2004)
12. Steels, L.: The emergence and evolution of linguistic structure: from lexical to grammatical communication systems. *Connection Science*, 17(3-4):213–230 (2005)
13. Steels, L. and Belpaeme, T.: Coordinating Perceptually Grounded Categories through Language: A Case Study for Colour. *Behavioral and Brain Sciences*, 28(4):469–89 (2005)
14. Steels, L. and Bleys, J.: Planning What To Say: Second Order Semantics for Fluid Construction Grammars. In: *Proceedings of CAEPIA 2005*, Santiago. LNAI Springer-Verlag, Berlin (2005)
15. Steels, L., De Beule, J., Neubauer, N.: Linking in Fluid Construction Grammar. In: *Transactions Royal Flemish Academy for Science and Art. Proceedings of BNAIC-05*, p. 11–18 (2005)
16. Steels, L., Wellens, P.: Scaffolding Language Emergence Using the Autotelic Principle. *Proceedings of IEEE Alife 07* (in press) (2007)
17. Winograd, T.: *Understanding Natural Language*, Academic Press (1972)