Development of Syntactic and Semantic Structure in Artificial Agents

Takashi Hashimoto*
Lab. for Information Representation, FRP,
The Institute of Chemical and Physical Research (RIKEN)
2-1, Hirosawa, Wako, Saitama, 351-01, JAPAN
Tel: +81-48-467-9626, Fax: +81-48-462-9881

Abstract

Development of syntax and semantics of language is studied with a constructive approach in a network model of agents. Agents with generative grammar systems evolve through communication in terms of their own grammar in a language game. They form a community sharing usages of symbols and develop to equip recursive structures. After development, agents become to have double articulation mechanism in their grammar. Semantics is expressed by word-space which is a space to represent interrelationship among words. The relationship is formed according to ways of usage of language and calculated by means of mutual dependency between words and sentences. An agent can develop categorization in its word-space from sentences.

1 Introduction

Language is an evolutionary system. At origins of language, it must have smaller number of words, simple syntax, few abstract notions. Our language constructed through word formation, grammaticalization, or diversification of metaphorical expressions. Even now language is evolving, pidgin/creole languages become complex, many new expressions are produced day by day, and so on. Some features in animal communication are inherited in our communication through evolutionary pathway [1]. Therefore it is important to study evolution of language from primitive communication systems.

Evolutionary linguistics is a new candidate to shed light on origins and evolution of language [2]. It is stressed as a key notion that origins and evolution of language is phenomena typically shown in complex systems as emergence, self-organization, adaptation, collective behavior, clustering. To understand complex systems, constructive approaches have great benefit [3]. By adopting this approach to understand evolution of language, we propose a language game in agents with grammar systems.

As a viewpoint to abstractly study communication systems, we have proposed a notion of *code* which is defined as a system of usages and interpretations of symbols in communication and expressions [4]. We are interested in the case of self-organizing code, a code emerging from interactions among elements. For example, dynamics of two characteristic self-organized codes, linguistic code [5] which is developed in this paper and genetic code [6, 7] have been studied.

*toshiwo@puneuma.riken.go.jp,
http://www.bip.riken.go.jp/irl/toshiwo/

Words have some semantic interrelationships. The most controversial problem is from what the interrelationships are constructed, namely *meaning* of words. A lot of discussion has been devoted for the problem. For example, meanings of words are indications to external objects, or representations with normal form [8]. Here we propose that ways of use of language constitute semantic relation among words, this notion has relevance with Wittgenstein's thought [9]. We insist that meaning of a word should be determined in part from usages of the word in expressions spoken in pragmatic situations.

If we imagine to measure distance among words and sentences, the interrelationships can be map onto abstract distance spaces, which we call *word-space*. Word-space is one of representations of language speaker's internal model.

We regard similarity among words as such interrelationships. To calculate similarity of words and sentences from ways of use of them, we adopt Karov and Edelman's algorithm [10]. Its main idea is a mutual dependency between words and sentences. Similar words are used in similar sentences and similar sentences are composed of similar words. Similarity matrices of words and sentences, mutually dependently defined along this idea, are iteratively calculated. Although this algorithm has been proposed for disambiguation of word meanings in machine readable dictionary and corpus, our main interest is dynamics of word-space, not static form as in corpus.

2 Self-organizing Code and Evolution of Grammar

2.1 Evolutionary Language Game

We introduce an evolutionary language game to study dynamics of linguistic code and evolution of grammar systems[1]. An agent is defined as a grammar system, $G_i = (V_{\rm N}, V_{\rm T}, F_i, S)$, where $V_{\rm N}$ is a set of non-terminal symbols, $V_{\rm T}$ is a set of terminal symbols, F_i is a set of rewriting rules, S is a start symbol, and a suffix i is ID of an agent.

At a speaking process, each agent apply its rewriting rules from the start symbol S. When a rewritten word consists of terminal symbols, it is spoken to all agents. In case of existing plural applicable rules, a rule is selected randomly. If there are non-terminal symbols in a rewritten word but there is no applicable rule in F_i, the agent cannot speak.

[1]Here we show model and results briefly. For details, please refer to [5]. Note that in spite of simpler definition of score in a language game (Eq. (1)) than that of [5], results are qualitatively similar.

When a reversely rewritten word from a word w attains to the start symbol S in a recognizing precess of an agent G_i, we say that the agent G_i can recognize the word w.

In our language game, P agents make a communication network. Each agent has chance to speak R times in a time step. Each agent is ranked by three different scores: recognizing – how many words and how quickly an agent recognizes, speaking – how many words an agent speaks, being recognized – how many words which an agent speaks are recognized. Total score is defined by weighted sum of all factors as

$$\begin{aligned} p^{\rm tot} = \quad & \tfrac{1}{R}\{r_{\rm rec}(s^{\rm rec}\sum_{recog}\frac{1}{step} - f^{\rm rec}) \\ & + r_{\rm sp}(s^{\rm sp} - f^{\rm sp}) + r_{\rm br}\frac{s^{\rm br} - f^{\rm br}}{P}\}. \end{aligned} \tag{1}$$

Where $s^{\rm rec}$, $s^{\rm sp}$, $s^{\rm br}$ are the times of speaking, recognizing, and being recognized, respectively. And $f^{\rm rec}$, $f^{\rm sp}$, $f^{\rm br}$ are the times of not speaking, not recognizing, and not being recognized, respectively. *step* is rewriting steps to recognize a word and a summation $\sum_{recog}$ is taken only when an agent can recognize words.

We introduce an evolutionary dynamics. A part of agents is replaced with new agent according to their score. Grammar of new agent is modified by adding (add a new rule to F_i), replacing (replace an old rule with a new rule), and deleting (delete a rule from F_i).

2.2 Self-organization of Code

Amount of information dealt by an agent is quantified by product of average length of words spoken and recognized, which is defined as,

$$f_l = \frac{1}{RP^2}\sum_{c=1}^{R}\sum_{k=1}^{P}|w_{kl}^{\rm rec}(c-1)|\sum_{m=1}^{P}|w_{lm}^{\rm rec}(c)| \tag{2}$$

$$|w_{ij}^{\rm rec}(c)| = \begin{cases} 1 & \text{if } c = 0 \\ |w_{ij}(c)| & \text{if } c > 0 \text{ and } G_i\text{'s word} \\ & \quad \text{is recognized by } G_j \\ 0 & \text{otherwise .} \end{cases} \tag{3}$$

We start simulations from agents with smallest grammars randomly generated, which are in the class of regular grammar. Amount of information flow in a network, measured by average of f_l ($\langle f\rangle = \sum_{l=1}^{P} f_l/P$), increases in course of time. As described in Fig. 1, it shows punctuated equilibrium evolution. Namely, we can see rapidly increase states and equilibrium states in turn.

In the regime of equilibrium state, agents make a community sharing a *code*, ways of use of words. Agents in a community gets higher score than other agents by speaking and recognizing particular words. They spontaneously elaborate particular usage of some words. We can say that a *code*, in a sense of a system of particular usage of symbols, is organized in a network. Since grammar is maintained by a whole network, we call it *net-grammar* [5]. From a community sharing a code, agents who can speak and recognize new words emerge by evolutionary processes which are described in the next subsection. And then a new code is organized. By repeating this process dynamics of code can be seen and amount of information flow in network shows stepwise development as in Fig. 1.

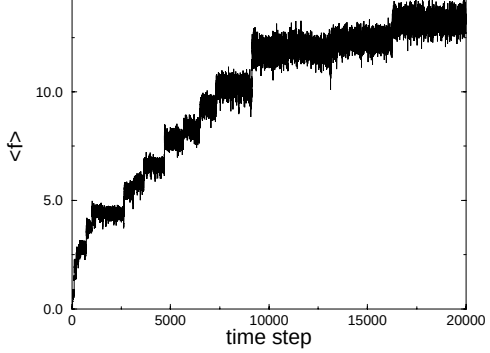

Figure 1: time step v.s. $\langle f\rangle$. $P = 10$ and $R = 10$. Parameters for the total score are $r_{sp} = 3$, $r_{rec} = 1$, and $r_{br} = 1$.

2.3 Evolution of Grammar

In the course of evolution of grammar to higher computational ability, there are two remarkable developmental processes. We mean by a term higher computational ability that an agent can speak and recognize larger set of words.

One is module type evolution. Using a new rule as a module, an agent can speak and recognize many new words. This process corresponds to a word formation process by affixation in natural language.

The other is emergence of loop structure. Equipping loop structures let agents be able to rewrite recursively. It corresponds to making nested sentences to produce expressions with embedded structure. A grammar with loop structure is equivalent to a grammar in the class of context free in the Chomsky hierarchy.

Agents which evolve to higher computational ability by these two processes speak and recognize long and many words, which is reflected on jumping states in Fig. 1.

2.4 Double Articulation As a Structuralization of Grammar

Through organization of code and evolution, grammar of agents is structuralized, which is called *double articulation*. After around 8000 time step, agents have two types of rewriting rules,

$$T \rightarrow \text{sequence of } T\text{s} \tag{4}$$

$$N \quad \rightarrow \text{sequence of } N\text{s} \mid \text{sequence of } N\text{s and } T\text{s}, \tag{5}$$

where N and T are a non-terminal and a terminal symbol, respectively.

Rules of Eq. (4) likely to be used to make frequently spoken words. By using rules of Eq. (5), agents combine several words produced by rules of Eq. (4) to make other rare and long words. In order to speak and quickly recognize words shared in a community, it is advantageous to have their words as single rules (i.e. $S \rightarrow$ sequence of Ts) which is a special type of Eq. (4).

in in natural language, making a sentence from words and making a word from symbols. Double articulation is one of the most important feature of natural language and gives us the ability to create infinite diverse sentences with finite symbols.

3 Development of Word-space

3.1 Words and Sentences

We have summarized our results of development in a syntactic level. Let us consider development in a semantic level. As stated in § 1, semantic relationship should be formed to some extent from ways of use of language. Taking semantic relationship as so, syntax and semantics can not completely be discriminated. So we can discuss developmental semantic structure of inter- and intra-agents as an advance of our model.

We introduce distance spaces of words and sentences and call it *word-/sentence-space*. The distance is similarity among words/sentences, which will be defined in the next subsection based on Karov and Edelman's definition [10]. The important notion to measure similarity among words and sentences is a mutual dependency of words and sentences. That is, similar words appear in similar sentences and similar sentences contain similar words.

We introduce a *word* and a *sentence* based on the double articulation obtained above so as to bring the idea of mutual dependency into our model. A word is successive terminal symbols in Eq. (4) and (5). A sentence is a sequence of terminal symbols. Each agent articulates sentences to sequences of words based on parsing with its own grammar. For example, an agent with a rewriting rule set, $S \rightarrow A0B, A \rightarrow 10, B \rightarrow 11$, parses a word "10011" as $10011 \overset{A \leftarrow 10}{\Rightarrow} A011 \overset{B \leftarrow 11}{\Rightarrow} A0B \overset{S \leftarrow A0B}{\Rightarrow} S$ and articulates it as a sequence of words "10·0·11". From this result of articulation, agents updates its word-space.

3.2 Definition of Similarity and Affinity

In this subsection, we define similarity and affinity among words and sentences according to Karov and Edelman's word sense disambiguation algorithm [10]. At first, affinity of a word for a sentence and that of a sentence for a word are defined as

$$aff_n(w,s) = \sum_{w' \in s} weight(w',s) sim_n(w,w') \ , \qquad (6)$$

$$aff_n(s,w) = \sum_{s' \ni w} weight(s',w) sim_n(s,s') \ , \qquad (7)$$

where a suffix n indicates times to iterate, $w \in s$ means a word included in a sentence s, $s \ni w$ means a sentence including a word w. Normalized factors $weight(w,s)$ and $weight(s,w)$ are defined as

$$weight(w,s) = \frac{factor(w,s)}{\sum_{w' \in s} factor(w',s)} \ , \qquad (8)$$

$$factor(w,s) = \frac{1}{p(w) lg(s)} \ , \qquad (9)$$

$$weight(s,w) = \frac{factor(s,w)}{\sum_{s' \ni w} factor(s',w)} \ , \qquad (10)$$

$$factor(s,w) = \frac{1}{p(s)\#(s,w)} \ . \qquad (11)$$

In $factor(s,w)$ and $factor(w,s)$, $p(w)$ and $p(s)$ is probabilities of a word w and a sentence s, respectively; $lg(s)$ is the length of a sentence s, which is defined by the number of words in the sentence; and $\#(s,w)$ is the number of occurrence of a sentence s including a word w.

Definitions for similarity between sentences and that of between words are

$$sim_{n+1}(s_i,s_j) = \sum_{w \in s_i} weight(w,s_i) aff_n(w,s_j) \ , \qquad (12)$$

$$sim_{n+1}(w_i,w_j) = \sum_{s \ni w_i} weight(s,w_i) aff_n(s,w_j) \ . \qquad (13)$$

Word similarity (Eq. (13)) makes word-space and sentence similarity (Eq. (12)) does sentence-space.

At initial iteration step ($n = 0$), diagonal part of word similarity ($sim_0(w_i,w_i)$) are 1, the others are 0. From this, word-sentence affinity (Eq. (6)) at $n = 0$ is calculated. Then, these four formulae are iteratively calculated (as Eq. (12) → (7) → (13) → (6)) until all elements converge.

3.3 Categorization in Word-space

Randomly generated sentences are given to an agent which is picked from the simulation described in Fig. 1 until the agent recognize 100 sentences. We investigate two ways of calculation. One is batch calculation which is to calculate after recognition of 100 sentences. This is to see convergence of similarity. The other is on-line calculation, updating per each recognition of a sentence, to see dynamics of word-space.

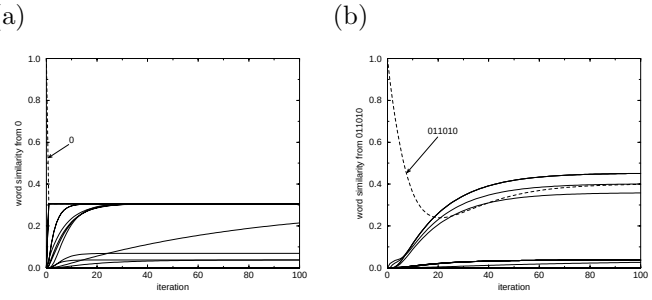

Figure 2: Convergence of similarity from word (a)0, (b)011010. Iteration step v.s. word similarity from the word. Broken lines are similarity with the word itself.

Changes of word similarity per iteration by batch calculation are shown in Fig. 2 for two words '0' and '011010'. The former is often used in sentences with other words. Similarities of almost words from the word '0' quickly converge. The latter is used only one time with a word '00' as "00·011010". Although the word '011010' have no direct relation with other words except for '00', similarities with other words slowly increase. The word '011010' become to have relation

by Karov and Edelman [10], this is an important feature to iterate the algorithm to calculate similarities.

An example of developmental path of word-space by online calculation is shown in Fig. 3. The agent divides 100 sentences into 25 words. Similarity among the words is clustered in 8 groups as shown in Fig. 3(d). We can see increase of the number of clusters and expansion of the number of words in a cluster from Fig. 3(a) to (d).

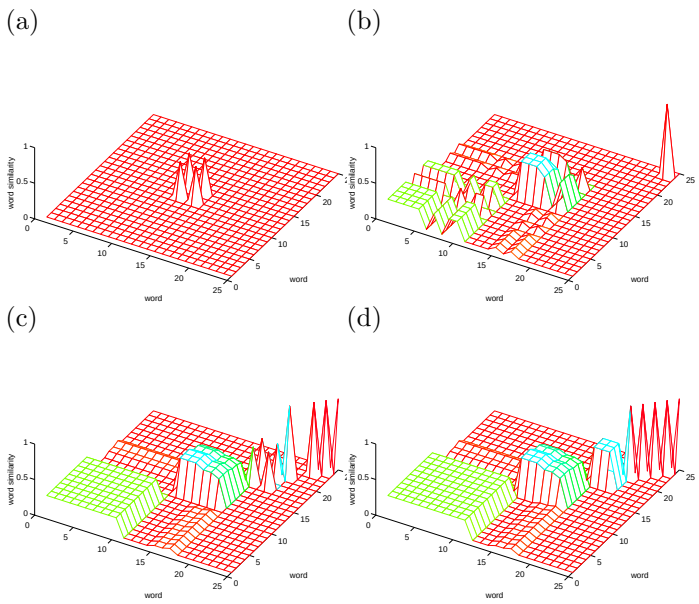

Figure 3: An example of development of word-space. The number of sentences is (a)1, (b)20, (c)50 and (d)100. Z-axis is word similarity. Words are arranged in xy-plane to clearly see clustering of word-space in (d).

This clustering can be regarded as categorization of words by the agent since words in a cluster have stronger relation with each other and less relation with words in the other cluster. Words $15 \sim 17$ in the second largest cluster have similarity with words $1 \sim 11$ forming the largest cluster. These word work to connect these two categories.

Words $20 \sim 25$ have the largest similarity only with itself. These words are one word sentences and are not used in other sentences. Therefore they have no relation with the other categories.

Not all agent categorizes their words. In case of another agent, although the same list of sentences is given to the agent, word-space have only a cluster and similarities of almost all words are same value.

4 Conclusion and Future

We have investigated development of syntax and semantics of language in an network model of agents with grammar systems.

Agents organize community sharing code, a system of usage of symbols. Grammar of them evolve to equip recursive structure. As a result of self-organization of code and evolution of grammar, double articulation mechanism is developed in their grammar.

Of course, results of development of semantic structure is preliminary. The development should be discussed through conversation among agents. If two agents have different rewriting rules, their word-spaces formed through conversation become to have different structure, even if they listen the same sentences. We are interested in dynamics of word-/sentence-space through interaction between different word-spaces. There may exist sentences which does not change one's word-space but change other's word-space. Sharing structure in word-spaces is a new type of linguistic code, in this context it is a system of usage and interpretation of language.

Dynamics of grammar and word-space is separated in this paper. To discuss developmental path of syntactic and semantic structure through communication, we should associate changes of grammar with structure in word-/sentence-space.

References

[1] Hauser, M. D., (1996), *The evolution of communication*, The MIT Press, Cambridge, MA

[2] Steels, L., (1997), Synthesising the origins of language and meaning using co-evolution, self-organisation and level formation, in *Evolution of Human Language*, Hurford, J (ed.), Edinburgh Univ. Press, Edinburgh (in press)

[3] Kaneko, K. and Tsuda, I., (1994), Constructive complexity and artificial reality: an introduction, *Physica*, **D75**, 1–10

[4] Hashimoto, T, (1996), *Evolution of Code and Communication in Dynamical Networks*, PhD. thesis, University of Tokyo

[5] Hashimoto, T. and Ikegami, T., (1996), Emergence of net-grammar in communicating agents, *BioSystems*, **38**,1–14

[6] Ikegami, T. and Hashimoto, T., (1995), Active mutation in self-reproducing networks of machines and tapes, *Artificial Life*, **2**, 305–318

[7] Hashimoto, T. and Ikegami, T., Oscillation of code in machines and tapes networks, (Manuscript in preparation)

[8] Putnum, H., (1975), The meaning of 'meaning', in *Mind, language and reality*, Cambridge University Press, New York, NY

[9] Wittgenstein, L., (1953), *Philosophische Untersuchungen*, Basil Blackwell

[10] Karov, Y. and Edelman, S., (1996), Similarity-based word sense disambiguation, Technical Report of Weizmann Institute, CS-TR 96-06