The Fifteenth International Symposium on Artificial Life and Robotics 2010 (AROB 15th '10),
B-Con Plaza, Beppu,Oita, Japan, February 4-6, 2010

Cultural Evolution of Compositional Language under Multiple Cognition of Meanings

R. Matoba[*1], S. Sakamoto[*1] and T. Hashimoto[*2]

**1 School of Information Science,*
**2 School of Knowledge Science,*
Japan Advanced Institute of Science and Technology,
1-1, Asahidai, Nomi, Ishikawa, 923-1292, Japan
(Tel : +81-0761-51-1372; Fax :+ 81-0761-51-1149)
{r-matoba, s0710202, hash}@jaist.ac.jp

Abstract: In the actual world, we often happen to meet a state which contains several meanings. In this paper, we call this phenomenon *multiple cognition of meanings*. Under an environment of multiple cognition, language needs to distinguish ``A do B'' from ``B is done by A'' to represent speaker's intension, even a state change which is pointed these two sentences are the same state change. Solving this matter, most present existing languages have functions such as case marking, agreement of person and number, or word order, etc. Here, one question is popped up, which is where these functions come from. Thinking of this question, we employed Iterated Learning Model which represents evolution of compositional language, and built state change and multiple cognition into this model. Simulating with our model, we speculated an influence of multiple cognition on evolution of compositional language.

Keywords: Cultural Evolution, Compositionality, Multiple Cognition, Iterated Learning Model

1. Introduction

In the actual world, we encounter a situation which contains several meanings frequently. For example, if there exist two objects, object A and object B separately on a table, and somebody puts object A on top of object B, then we can describe this action in different expressions such as ``object A is put on object B", or ``object B is put under object A", even the way the state has changed is the same. In this paper, we use a term *multiple cognition* as one uttered situation which contains several possible meanings, like above example. To represent speaker's intention precisely, existing languages have grammatical features such as agreement of number, person, and gender, case marking, or word order, and so on. Taking advantages of these features, language users are able to discriminate subject and object of a sentence, i.e., preventing influence of multiple cognition. As long as a person communicates with a person, there are possibilities to happen multiple cognition, but it is hard to assume that grammatical features for preventing influence of multiple cognition had existed from origin of language. Hence, these features might have been generated in the course of language evolution.

In this paper, we employed the Iterated Learning Model (ILM, hereafter) by Kirby[1] which is one of models of cultural evolution represents evolution of compositional language from holistic language, and built state change and multiple cognition into ILM. Through the computer simulation using our model, we speculated an influence of multiple cognition on evolution of compositional language.

2. ILM with State Change

2.1. Briefing Kirby's Model

Our study is based on the ILM by Kirby[1], who introduced the notions of *compositionality* and *recursion* as fundamental features of grammar, and showed that they made it possible for a human to acquire compositional language. Also, he adopted the idea of two different domains of language[2,3,4], namely, I-language and E-language; I-language is the internal language corresponding to speaker's intention or meaning, while E-language is the external language, that is, utterances. In his model, a parent is a speaker agent and her infant is a listener agent. The speaker agent gives the listener agent a pair of a string of symbols as an utterance (E-language), and a predicate-argument structure (PAS) as its meaning (I-language). A number of utterances would form compositional grammar rules in listener's mind, through learning process. This process is iterated generation by generation, and finally, a certain generation would acquire a compact, limited number of grammar rules. We include a meaning inference using a state change into this process. We implement agents with state change in a virtual world, and make them learn a grammar by computer simulation.

In our model, we changed two points of Kirby's model, which are (i) taking away the transmittance of meanings between the parent and the infant, and (ii) giving a state change which contains more than one meaning.

2.1.1 Utterance Rule of Kirby's Model

According to Kirby's model, we show a pair of I-language and E-language as follows.

S/eat(john, apple) → `eatjohnapple` (1)

Where a speaker's intention is a PAS *eat(john, apple)* and its utterance becomes \``eatjohnapple`'; the symbol \`*S*' stands for that they belong to the category Sentence. Thus, as far as a listener is given an utterance paired with its meaning (PAS), the listener can understand the speaker's intention precisely at all times. However, compared to the actual situation, it seems a very strong assumption. In our model, we loosen this assumption and build state change instead of meaning share into this model. This means that the listener agent receives utterances without meanings, to show the influence of multiple cognition of meanings.

2.1.2 Rule Subsumption

chunk This operation takes pairs of rules and looks for the least-general generalization. For example, if there are two rules below,

S/read(john, book) → `ivnre` (2)
S/read(mary, book) → `ivnho` (3)

then, after operation chunk, the two rules above become

S/read(x, book) → `inv N/x` (4)
N/john → `re` (5)
N/mary → `ho` (6)

merge If two rules have the same meanings and strings, replace their nonterminal symbols for one common symbol. For example, if given rules below,

S/read(x, book) → `ivn A/x` (7)
A/john → `re` (8)
A/mary → `ho` (9)
S/eat(x, apple) → `apr B/x` (10)
B/john → `re` (11)

then, after operation merge, the rules above become

S/read(x, book) → `ivn A/x` (12)
A/john → `re` (13)
A/mary → `ho` (14)
S/eat(x, apple) → `apr A/x` (15)

replace If a rule is embeddable in another rule, replace the latter for a compositional rule with variables. For example, if given two rules below,

S/read(pete, book) → `ivnwqi` (16)
B/pete → `wqi` (17)

then, after operation replace, the rules above become

S/read(x, book) → `ivn B/x` (18)
B/pete → `wqi` (19)

2.2. State, State Change, Action of Predicates

The difference point between Kirby's model and our model is that the listener cannot always get a meaning of an utterance precisely, i.e., in Kirby's model, the listener always gets a pair of the utterance and its meaning (PAS). Instead of getting the meaning of the utterance, the listener gets the utterance and a state change which corresponds to the utterance, and uses them for his learning.

In our representation, the state which agents pay attention to is constructed by five connected boxes, and two or three boxes out of five are filled with numbers, from one to five. See figure 1.

	1		5	

Fig.1. State representation

The state change is represented as a change of numbers and a change of places which numbers are filled with. The agent's language of our model is constructed five kinds of predicates and five kinds of arguments like Kirby's model. The arguments are numbers from one to five, and the predicates are \`step', \`gather', \`swap', \`put', and \`sub'. In our model, each predicate has an operation to change a state. The followings are the definitions of each predicate. Hereafter, we call a state which is not operated yet \`before state', and a state which is operated by the parent \`after state'.

step(x, y)
Condition: Before state contains x and y, and there is at least one box between x and y.
Operation: move x to y one box. If the box next to x is filled with a number, swap the number and x.

gather(x, y)
Condition: Before state contains x and y, and there is at

least one box next to y.
Operation: move x next to y.
swap(x, y)
Condition: Before state contains x.
Operation: If before state contains y, swap a position of x and y. If before state does not contain y, swap x and y, and x disappears from after state.
put(x, y)
Condition: Before state contains x, and does not contain y, also there is at least one box next to x.
Operation: add y next to x.
sub(x, y)
Condition: Before state contains x, and x > y, also does not contain x-y.
Operation: Subtract y from x (rename x to x-y).

3. What is Multiple Cognition of Our Model?

To represent multiple cognition, i.e., several meanings for one situation, the operations performed by the parent are designed to overlap its action deliberately. Figure 2 indicates multiple cognition of `step(2, 4)', `step(2, 5)', and `gather(2, 5)'.

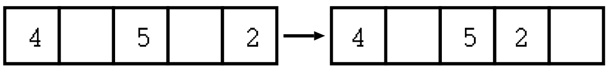

Fig.2. Example of multiple cognition
step(2, 4) – step(2, 5) – gather(2, 5)

III. Experimented Procedure

The simulation implements these processes:
1. The speaker tries to produce utterances which will form input to the listener. This process repeats number of times (depends on the experimenter: in our experiment, it is limited to 50 times, which is the same as Kirby's experiment).
(a) The speaker is given a before state and an operation (meaning) chosen randomly from a meaning space.
(b) The speaker executes the operation, and makes an after state.
(c) If the speaker is able to generate strings as an utterance for that operation using her grammar, she makes it. Otherwise, she invents strings randomly, and if the need arises, she uses this invented rule again.
(d)The listener infers the operation (meaning) of the strings (utterance). If he can figure out the operation of strings, uses a pair of the operation (meaning) and the strings (utterance) to his learning. If there exists several candidates of operation, choose them randomly. If he cannot figure out the operation of the strings, then he adapts one operation randomely.
2. The speaker's grammar is logged, and she is deleted from the simulation
3. The listener becomes the new speaker, and a new listener with an empty grammar knowledge is added to the simulation.

IV. Experiment and Result

In this section, we show the procedure and the result of our experiment. The purpose of the experiment is to demonstrate acquisition of compositional language under environment of multiple cognition. To evaluate the accomplishment of the learning, we investigate expressivity in the following definition, as well as the number of grammar rules.

Definition (Expressivity)
Expressivity is the ratio of the utterable meanings against the whole meaning space.

The experiment was carried out until the 100th generation. In fact, we have carried out until the 1000th generation; however, both of expressivity and the number of rules converged until the 100th generation, and thus we discuss the result derived by the 100th generation.

2. Experiment 2: Learning without Meaning of Utterance

In this experiment, the listener does not get a meaning of an utterance, and infers the meaning from the utterance and a state change. Different from Kirby's model, incomprehensible utterances are given at the tail of the 50 utterances, e.g., in case of 20 %, the first 40 sentences are given paired with PAS as training data while the rest 10 utterances are meaningless. This is because inference of the listener must be evoked after a certain accumulation of grammar knowledge. Namely, the listener could consider that the intention of the speaker was not clear but referring back to the previous knowledge the listener could partially guess what the speaker had said. Figure 3 and Figure 4, the results of the average of 100 trials, show that the expressivity and the number of rules at the rate of the inference rate are set from 0% to 100 %.

The results of this experiment show that while the inference rate is low, i.e., the listener can get many training data from the speaker, the expressivity of the grammar indicates high value, and the number of rules is small. Namely, the listener acquires compositional language. On the other hand, the more inference rate

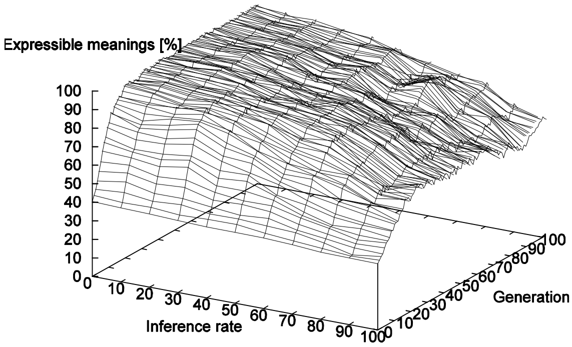

Fig.3. The movement of the expressivity per generation with inference.

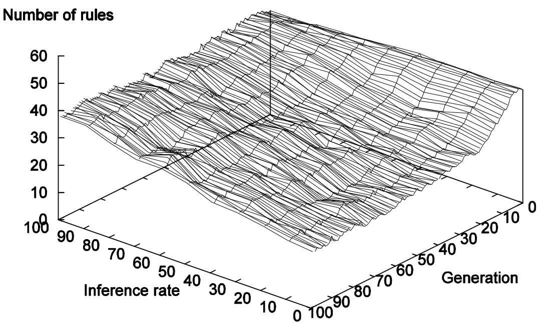

Fig.4. The movement of the number of rules per generation with inference.

increases, the more the expressivity decreases and the number of rules increases. The possible reason is that the complement of meanings by inference is incomplete, the listener acquires rules which were originally not included in the speaker's grammars, so the listener cannot generalize such rules well. Also, the mechanism to avoid multiple cognition of our model is only the training data from the speaker, so increase of inference rate leads low expressivity and large number of rules. Table 1 shows the acquired grammar of the listener in the convergence generation while inference rate is 8 %, and table 2 shows the result of the inference rate is 50 %. Obviously, we can observe many polysemous words in table 2.

V. Conclusion and Future Work

In this paper, we investigated the influence of multiple cognition to the development of compositional language with Iterated Learning Model. As the result of our experiment, without training data from the speaker, the listener could not avoid the influence of multiple cognition. The result of ILM strongly responds to learning algorithm, so instead of just following Kirby's model, we need to improve learning algorithm of our model. In our present model, if a need to choose one meaning from several meanings arises, the listener chooses randomly. In the near future, using weighted selection, we will improve this random selecting method to a method with directional characteristics for selecting one meaning. Also, we are planning to employ prefix or suffix to the arguments of predicate of language in our model to avoid multiple cognition.

Table 1. Sample grammar of convergence generation. Inference rate is 8 %.

S/p(x, y) → `j A/x g A/y D/p inrgh`	
A/1 → `h`	*D/gather* → `k`
A/2 → `i`	*D/put* → `la`
A/3 → `m`	*D/step* → `gnnag`
A/4 → `ji`	*D/sub* → `nke`
A/5 → `f`	*D/swap* → `kr`

Table 2. Sample grammar of convergence generation. Inference rate is 50 %.

S/p(x, y) → `A/x e C/p oc A/y`	
S/p(x, y) → `A/y e C/p poc A/x`	
A/1 → `q`	*D/gather* → `v`
A/2 → `q`	*D/put* → `czc`
A/3 → `q`	*D/step* → `v`
A/4 → `q`	*D/sub* → `v`
A/5 → `cr`	*D/swap* → `v`

REFERENCES

[1] Kirby, S. (2002), Learning, bottlenecks and the evolution of recursive syntax. In: Linguistic Evolution through Language Acquisition. Cambridge University Press.
[2] Kirby, S. (1999), Function, Selection and Innateness: The Emergence of Language Universals. Oxford: Oxford University Press.
[3] Chomsky, N. (1986), Knowledge of Language. Praeger, New York.
[4] Hurford, J. (1987), Language and Number: The Emergence of a Cognitive System. Blackwell, Oxford.