CURRENT RESEARCH IN THE COMPUTER SIMULATION OF HISTORICAL CHANGES IN LANGUAGE

by

Sheldon Klein
University of Wisconsin
Madison, Wisconsin 53706
U.S.A.

Computer Sciences Technical Report #6

August 1967

Presented at:

Xth International Congress of Linguists
Bucharest, Rumania
August 28 - September 2, 1967

(Conference Papers limited to
6 pages of 2000 characters.)

CURRENT RESEARCH IN THE COMPUTER SIMULATION OF HISTORICAL CHANGE IN LANGUAGE*

by

Sheldon Klein
University of Wisconsin
Madison, Wisconsin
U.S.A.

1.0 Introduction

The primary purpose of this research is to establish the usefulness of a research methodology that is new to the field of linguistics. It is anticipated that the methodology will permit computer testing of models and hypotheses pertaining to language change and group language behavior that are not subject to direct empirical testing nor to static, deductive analysis because of the complexity of the phenomena. The hypotheses and models within the domain of the research include psychological, sociocultural, and demographic factors as well as those of a purely linguistic nature.

The simulation system is relatively independent of any models or hypotheses it may be used to test. What is inherent is that each member of a speech community be associated with a generation grammar and a recognition grammar, and that members of the community converse with each other (generate and parse terminal strings). The conversational interactions are determined by some extralinguistic model made up of stochastic rules that may, as indicated earlier, refer to psychological, socio-cultural, and demographic factors. Accordingly, each individual is also associated with various parameter values that are pertinent to the extralinguistic model. The grammars of individuals are not necessarily identical. Language change for an individual is a result of interaction with other individuals with dissimilar grammar rules.

A complete description of an earlier version of the simulation system is contained in (1). Also described therein is an experiment in the simulation of twenty-five years in a hypothetical speech community. To insure control of free variables before undertaking experiments with factors causing change, an initial experiment was performed to obtain a condition of linguistic stability, and essentially identical results for the population as a whole from several computer runs differing only in the choice of random numbers referred to in decision making processes.

---

*This research was sponsored by the National Science Foundation and the Wisconsin Alumni Research Foundation.

The learning mechanisms described in the next section are not all operational. Most are working in some form but are not yet integrated into the simulation system. The system described in (1) is being translated from the JOVIAL programming language into ALGOL for the Burroughs 5500 and 8500 computers. The heuristics discussed in the next section are already written in that version of ALGOL.

2.0 Learning Mechanisms

Language learning in the previously mentioned experiment took place in a trivial manner: if an auditor in the model encountered an unfamiliar construction, he simply took the appropriate rules from the speaker's grammar and added them to his own.

Current work on the system has been directed toward the construction of programs which synthesize rules. Accordingly, programs have been designed for learning context free and context sensitive phrase structure rules, and bilingual as well as monolingual transformations. Work is also being directed to the construction of a program for learning languages within the framework of a stratificational model of grammar.

2.1 A Program for Learning Phrase Structure Rules

This program, as well as the one for learning transformations, was written first for testing in the AUTOLING, an automated linguistic fieldworker (2). In this system, the computer interacts via a teletype with a live informant. One of the tests of the validity of the rules derived by the program is the informant's acceptance of productions derived from them. All the learning mechanisms are heuristic rather than algorithmic: they do not guarantee acquisition of a complete grammar (3).

In the simulation system, the validity checking role of the informant is to be supplanted by other members of the speech community. For example, if the grammar of a child in the system yields an ungrammatical form in a conversation with an adult, the grammar of the adult may reject the string, causing the child speaker to modify its rules.

The rules derived are unordered and fit the more general definition of context free language that is prevalent in the description of computer programming languages rather than the more restricted definition customarily used by transformational linguists.

Although programs are being developed to perform such analyses, the heuristics given here do not cover morphology and phonology. Any rule formulated by any of the following heuristics may be discarded if it yields a production unacceptable to other speakers in the system. Children will of course be most sensitive to rejection of rules by adults. On the other hand, adult rule discrepancy can be made less subject to modification.

Heuristic 1 If a terminal string cannot be completely parsed, coin a rule yielding the nodes of the top of the partially analyzed tree. Where several partial parses are possible, the tree with the least number of top nodes is used. Where there are several parses with equal numbers of top nodes, the first derived is selected. If a given rule coined by these criteria is rejected, the top of the next preferred parse is used to coin a rule. Where there are no rules applicable to a string, this heuristic yields the string itself as the right half of a production.

Heuristic 2 Given two rules of the form $S_i \rightarrow X\ A\ Y$ and $S_j \rightarrow X\ B\ Y$, where X, A, Y, and B are strings, and where either X or Y may be empty but not both, the system coins two new rules, $S_k \rightarrow A$ and $S_k \rightarrow B$ and rewrites the two initial rules as $S_i \rightarrow X\ S_k\ Y$. If the more general rules yield productions that are subsequently rejected, the more specific rules are restored.

Heuristic 3 The current program makes a special test for recursion whenever the right half of a rule contains two or more identical symbols in sequence. Given rules of the type $S_i \rightarrow A\ X\ B$ and $S_j \rightarrow A\ X\ X\ B$, the program coins a rule of the type $X \rightarrow X\ X$, because of heuristic 2 and also to avoid making an arbitrary and possibly incorrect assignment of immediate constituents. The proper immediate constituency may be assigned later as a function of a higher level rule via the earlier heuristics.

Heuristic 4 Another heuristic (not currently implemented) is capable of learning recursion and embedding as a single case (in combination with heuristics 1 and 2). If the right and left halves of a rule R_1 each contain only one symbol, the left half of R_1 may be used to replace the left half of some rule R_2 if the left half of R_2 is identical to the right half of R_1. For example, given:

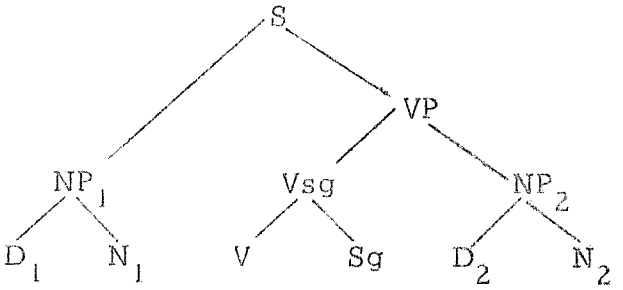
a. $S_1 \rightarrow X\ Y$, b. $S_2 \rightarrow X\ S_1\ Y$, and c. $S_3 \rightarrow X\ S_2\ Y$,

Heuristic 2 permits the coining of d. $S_4 \rightarrow S_1$ and e. $S_4 \rightarrow S_2$, and the replacement of rules b and c by the rule $S_2 \rightarrow X\ S_4\ Y$. Heuristic 4 then permits the coining of the rule $S_4 \rightarrow X\ S_4\ Y$.

Heuristic 5 If a general rule coined by one of the other heuristics is subsequently rejected, the rule may be converted to one which makes it applicable only in the context in which it was first coined. While this heuristic is not yet fully implemented, the data base of the system has been given the capability of representing context sensitive phrase structure rules.

2.2 Heuristics for Learning Transformational Rules

Two separate heuristic learning programs have been defined: bottom-to-top and top-to-bottom. The top-to-bottom method is the version actually implemented in a working program, and is attributable to William Fabens (1). Both methods yield learning of bilingual as well as monolingual transformations. Also, the inputs of both approaches are identical: a strong that is to undergo transformation plus its phrase structure analysis, and the string into which it is to be transformed (without any phrase structure analysis). The existence of a lexicon, a parser and phrase structure grammar is assumed. The following might be an input:



a man bite s the dog : the dog is bitt en by the man

2.2.1 Bottom-to-Top Learning

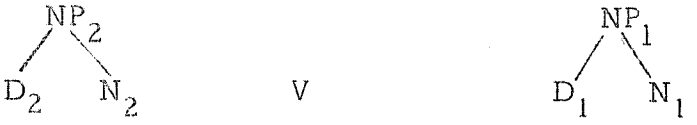
The first transformation posited is the most specific:

A man bite s the dog → the dog is bitt en by the man

The next level of generality is obtained by assigning like parts of speech to items common to both strings (the lexicon is assumed to contain sufficient information to recognize 'bit-' as a variant of 'bite'): The next transformation posited is:

$D_1\ N_1\ V\ Sg\ D_2\ N_2 \rightarrow D_2\ N_2$ is V en by $D_1\ N_1$

The next transformation posited is derived by assigning to the transform string the next level of tree structure common to both strings:



the dog is bitt en by the man

yielding:

$NP_1\ V\ Sg\ NP_2 \rightarrow NP_2$ is V en by NP_1

Since no other higher level structure is common to both strings, the program determines this to be the most general transformation obtainable. Should a transformation yield an invalid production (rejected by another speaker), the level of generalization is cyclycly lowered one phrase unit at a time until an acceptable maximum level of generality is obtained.

2.2.2 Top-to-Bottom Learning

This method (too lengthy to describe in detail for this paper) defines a group of ordered transformations which may be optional as a package, but all obligatory once the first in that package is applied. The top-to-bottom learning program yielded the following transformations for the preceding input:

1. $S(NP_1 \quad VP) \rightarrow S(VP \quad en \quad by \quad NP_1)$
2. $VP(NP_2 \quad V) \rightarrow VP(NP_2 \quad is \quad V)$
3. $V(VS \quad S) \rightarrow V(VS)$

where X(Y Z) indicates that Y and Z are immediately dominated by X. The top-to-bottom method, although somewhat more elegant than the other, permits no choice of levels of generality. Additional heuristics, as yet unspecified, are needed to salvage an over-generalization.

2.3 Bilingual Transformations

Both methods yield learning of bilingual transformations from the same inputs. Additional information required is a one-to-one correspondence of lexical items common to the input and the desired output. These correspondences in no way limit the system to learning word for word translation, but rather serve as indices for matching phrase units. Nor one-to-one correspondences are handled by deletion and adjunction.

3.0 Learning Model for Stratificational Grammar

Work on this system is at a less advanced stage then for the transformational model. Many of the programs are common to both systems. In constructing the specific portions the author will modify some of his already existing systems which are approximations to a stratificational generative grammar (4). These include a device for learning semotactic rules via dependency analysis. The implemented model will not use the logic network notation of Sydney Lamb, but rather a more abstract logical equivalent.

Notes and References

1. Klein, S. Historical Change in Language Using Monte Carlo Techniques, Mechanical Translation, Vol. 9, Nos. 3 & 4, Sept. & Dec., 1966, (published, May 1967).

2. Klein, S., Davis, B. C., Fabens, W., Herriot, R. G., Katke, W. J., Kuppin, M. A., and Towster, A. E. "AUTOLING: An Automated Linguistic Fieldworker," presented at the 1967 International Conference on Computational Linguistics, August 23-25, Grenoble.

3. E. Shamir has written a paper proving the impossibility of obtaining a discovery algorithm for grammars. A Remark on Discovery Algorithms for Grammars, Information and Control, Vol. 5, 1962.

R. Solomonoff, in "A New Method for Discovering the Grammars of Phrase Structure Languages," Information Processing, Proceedings of the International Conference on Information Processing, Unesco, 1959, UNESCO, Paris, presents discovery procedures already well known in the field of linguistics, and described by such researchers as Leonard Bloomfield, Zellig Harris, and Rulon Wells.

At a RAND Linguistics Colloquim meeting in December 1964, Santa Monica, California, I suggested to Paul Garvin, the invited speaker, that he might use the concept of heuristic problem solving in his work on automated language learning. He adopted the suggestion and presented this approach at the 1965 Meeting of the Linguistic Society of America, in the paper, "Automation of Discovery Procedure in Linguistics," which is also to appear in a forthcoming issue of Language.

4. Klein, S., Automatic Paraphrasing in Essay Format, Mechanical Translation, Vol. 8, Nos. 3 & 4, June and Oct. 1965.

Klein, S., Leiman, S. L., and Lindstrom, G. E., DISEMINER: A Distributional-Semantics Inference Maker, (in press) Computer Studies in the Humanities and Verbal Behavior, also presented at the 1966 meeting of the Association for Machine Translation and Computational Linguistics, Los Angeles.

Lamb, Sydney, Outline of Stratificational Grammar, Georgetown University Press, Washington, D. C., 1966.