cs -109

A COMPUTER SYSTEM FOR WRITING AND TESTING TRANSFORMATIONAL GRAMMARS

FINAL REPORT

JOYCE FRIEDMAN
PRINCIPAL INVESTIGATOR

This research was supported in part by the United States Air Force Electronic Systems Division, under Contract F196828-C-0035.

STANFORD UNIVERSITY COMPUTER SCIENCE DEPARTMENT
COMPUTATIONAL LINGUISTICS PROJECT

30 SEPTEMBER 1968

CS - 109
AF - 38

A COMPUTER SYSTEM FOR WRITING AND TESTING TRANSFORMATIONAL GRAMMARS

Final Report

Joyce Friedman*
principal-investigator

This research was supported in part by the United States Air Force Electronic Systems Division, under Contract F196828-C-0035.

30 September 1968

*present address: Computer and Communication Sciences Department, University of Michigan, Ann Arbor, Michigan.

For the past two years the Computational Linguistics Project in the Computer Science Department at Stanford University has been engaged in research leading to computer programs for accepting and manipulating transformational grammars corresponding to a version of the theory based on Chomsky's *Aspects of the Theory of Syntax,* M.I.T. Press, 1965. These programs have been combined into a computer system for transformational grammar which accepts a transformational grammar in a natural format, and carries out the complete generation of sentences, from phrase structure generation, through lexical insertion and transformation. These programs are the first to handle complete sentence generation. The system has been made available on a limited basis to linguists writing transformational grammars, and has proved valuable.

In order to construct the computer system, it was necessary to do considerable preliminary work in formalizing and making more precise the linguistic notions involved,, Thus, in addition to the value of the programs *per* se, the project has made some interesting contributions to linguistic theory, particularly in the areas of formal definition of grammars, lexical insertion, and traffic rules for transformations. The results obtained are described in reports which have been issued during the-course of the project. To summarize these results the abstracts of the more important papers are included here. The bibliography attached to this report contains a complete list of all current reports -produced by the project, Reports that became obsolete as the project developed have been omitted,

AF - 21
cs - 84

January 1968

A Computer System for Transformational Grammar

by

Joyce Friedman

Abstract

A comprehensive system for transformational grammar has been designed and is being implemented on the IBM 360/67 computer. The system deals with the transformational model of syntax, along the lines of Chomsky's *Aspects of the Theory of Syntax*. The major innovations include a full and formal description of the syntax of a transformational grammar, a directed random phrase structure generator, a lexical insertion algorithm, and a simple problem-oriented programming language in which,the algorithm for application of transformations can be expressed, In this paper we present the system as a whole, first discussing the philosophy underlying the development of the system, then outlining the system and -discussing its more important special features. References are given to papers which consider particular aspects of the system in detail,

AF - 14
CS - 79

October 1967

360 O.S. Fortran IV Free Field Input-Output Package

by

Robert W. Doran

Abstract

Programmers dealing with aspects of natural language processing have a difficult task in choosing a computer language which enables them to program easily, produce efficient code and accept as data freely written sentences with words of arbitrary length., List processing languages such as LISP are reasonably easy to program in but do not execute very quickly. Other, formula oriented, languages like FORTRAN are not provided with free field input,

The Computational Linguistics group at Stanford University Computer Science Department is writing a system for testing transformational grammars. As these grammars are generally large and complicated it is important to make the system as efficient as possible, so we are using FORTRAN IV (G.S. on IBM 360-65) as our language. To enable us to handle free field input we have developed a subroutine package which we describe here in the hope that it will be useful to others embarking on natural language tasks,,

The package consists of two main programs, free field reader, free field writer, with a number of utility routines and constant COMMON blocks.

AF - 24
CS - 95

February 1968
(Revised May 1968)

A Formal Svntax for Transformational Grammar

by

Joyce Friedman and Robert W. Doran

Abstract

A formal definition of a descriptive metasyntax for transformational grammar is given using a modified Backus Naur Form as the metalanguage. Syntax constraints and interpretation are added in English. The underlying model is that presented by Chomsky in Aspects of the Theory of Syntax. Definitions are given for the basic concepts of tree, analysis, restriction, complex symbol, and structural change, as well as for the major components of a transformational grammar, phrase structure, lexicon, and transformations. The syntax was developed as a specification of input formats for the computer system for transformational grammar described in [6]. It includes as a subcase a fairly standard treatment of transformational grammar, but has been generalized in many respects,

AF - 15
cs - 80

October 1967

Directed Random Generation of Sentences

by

Joyce Friedman

Abstract

The problem of producing sentences of a transformational grammar by using a random generator to create phrase structure trees for input to the lexical insertion and transformational phases is discussed. A purely random generator will. produce base trees which will be blocked by the transformations, and which are frequently too long to be of practical interest. A solution is offered in the form of a computer program which allows the user to constrain and direct the generation by the simple but powerful device of restricted subtrees. The program is a directed random generator which accepts as input a subtree with restrictions and produces around it a tree which satisfies the restrictions and is ready for the next phase of the grammar. The underlying linguistic model is that of Noam Chomsky, as presented in Aspects of the Theory of Syntax. The program is written in Fortran IV for the IBM 360/67 and is part of the Stanford Transformational Grammar Testing System. It is currently being used with several partial grammars of English.

Analysis in Transformational Grammar

by

Joyce Friedman and Theodore S. Martner

Abstract

In generating sentences by means of a transformational grammar, it is necessary to analyze intermediate trees, testing for the presence or absence of various structures. This analysis occurs at two stages in the generation process -- during insertion of lexical items (more precisely, in testing contextual features), and during the transformation process, when individual transformations are being tested for applicability.

In this paper we describe a formal system for the definition of tree structure of sentences. The system consists of a formal language for partial or complete definition of the tree structure of a sentence, plus an algorithm for comparison of such a definition with a tree, It represents a significant generalization of Chomsky's notion of "proper analysis," and is flexible enough to be used within any transformational grammar which we have seen.

AF - 25
CS - 103

June 1968

Lexical Insertion in Transformation Grammar

by

Joyce Friedman and Thomas H. Bredt

Abstract

In this paper, we describe the lexical insertion process for generative transformational grammars. We also give detailed descriptions of many of the concepts in transformational theory. These include the notions of complex symbol, syntactic feature (particularly contextual feature), redundancy rule, tests for pairs of complex symbols, and change operations that may be applied to complex symbols., Because of our general interpretation of redundancy rules, we define a new complex symbol test known as compatibility. This test replaces the old notion of nondistinctness. The form of a lexicon suitable for use with a generative grammar is specified.

In lexical insertion, vocabulary words and associated complex symbols are selected from a lexicon and inserted at lexical category nodes-in the tree. Complex- symbols-are lists of syntactic features, The compatibility of a pair of complex symbols and the analysis procedure used for contextual features are basic in determining suitable items for insertion. Contextual features (subcategorization and selectional) have much in common with the structual description for a transformation and

we use the same analysis procedure for both. A problem encountered in the insertion of a complex symbol that contains selectional features is side effects. We define the notion of side effects and describe how these effects are to be treated.

The development of the structure of the lexicon and the lexical insertion algorithm has been aided by a system of computer programs that enable the linguist to study transformational grammar. In the course of this development, a computer program to perform lexical insertion was written. Results obtained using this program with fragments of transformational grammar are presented. The paper concludes with suggestions for extensions of this work and a discussion of interpretations of transformational theory that do not fit immediately into our framework,

A Control Language for Transformational Grammar

by

Joyce Friedman and Bary W. Pollack

Abstract

Various orders of application of transformations have been considered in transformational grammar, ranging from unordered to cyclical orders involving notions of "lowest sentence' and of numerical indices on depth of embedding. The general theory of transformational grammar does not yet offer a uniform set of 'traffic rules" which are accepted by most linguists. Thus, in designing a model of transformational grammar, it seems advisable to allow the specification of the order and point of application of transformations to be a proper part of the grammar.

In this paper we present a simple control language designed to be used by linguists for this specification.

In the control language the user has the ability to:

1. Group transformations into ordered sets and apply transformations either individually or by transformation set.

2. Specify the order in which the transformation sets are to be considered.

3. Specify the subtrees in which a transformation set is to be applied.

4. Allow the order of application to depend on which transformations have previously modified the tree.

5. Apply a transformation set either once or repeatedly.

In addition, since the control language has been implemented as part of a computer system, the behavior of the transformations may be monitored giving additional information on their operation.

In this paper we present the control language and examples of its use, Discussion of the computer implementation will be found in Pollack [1].

AF - 33
CS - 108

September 1968

Computer Experiments in Transformational Grammar

by

Joyce Friedman, Lorraine Klevansky,
Theodore S. Martner, Barbara H. Partee,
and Elizabeth C. Traugott

Abstract

The papers in this volume describe computer runs with six different transformational grammars, in each case using the computer system for transformational grammar described in CS-84 (January 1968). They are collected here as examples which we hope will encourage other linguists to use the system.

The motivation for the first three projects described was primarily to test the system. The remaining papers describe experiments by linguists using the system as a tool in their own research.

In some of the papers there are occasional remarks which indicate a misunderstanding of the system. Editorial notes have been added to clarify these points. Otherwise, the papers are presented without alteration.

i

AF - 36 September 1968

Programmers Manual for a Computer Svstem for Transformational Grammar

by

Joyce Friedman, Thomas H. Bredt,
Robert W. Doran, Theodore S. Martner,
and Bary W. Pollack

Abstract

This Manual is written by and for programmers. Its purpose is to make the code of the computer system for transformational grammar more readily understandable to programmers who wish to maintain and use the system, or to modify and extend it. Section 2 is a short outline of the subroutine structure of the system. It is followed in Section 3 by more detailed descriptions of the subroutines. Sections 4 and 5 are listings of the COMMON blocks and BLOCK DATA statements, respectively, Section 6 discusses possible extensions to the system.

The programs are written in FORTRAN IV for the IBM 360/67 and were compiled under FORTRAN H, OPT=2, under O.S. There are approximately 9000 lines of FORTRAN code; the compiled code, with storage areas, requires approximately 300,000 bytes of storage.

The inputs to the system consist of

1. a grammar (described by the formal syntax of AF-95),
2. a one-line driver for the MAIN program,
3. input trees or skeletons.

Bibliography

Reports of the Computational Linguistics Project

AF - 3 Programming lexical grapho-morphic analysis. Joyce Friedman (Sept 1966).

AF - 4 A new method for storing grammars and its application to checking trees. Alan C. Tucker (Dec 1966).

AF - 8 The Tucker parser. Alan C. Tucker (Apr 1967).

AF - 9 Design of the programmer interface for a transformational grammar programming system. Robert W. Doran (May 1967).

AF - 10 The applicability of computational linguistics. Joyce Friedman (May 1967).

AF - 13 AF test grammar. Olasope O. Oyelaran (Sept 1967).

AF - 14
cs - 79 360 O.S. Fortran IV free field input-output package, Robert W. Doran (Oct 1967).

AF - 15
cs - 80 Directed random generation of sentences. Joyce Friedman (Oct 1967).

AF - 20 Linear representation of tree structure I: basic concepts; isotone notations. William J. Meyers (Nov 1967).

AF - 21
cs - 84 A computer system for transformational grammar. Joyce Friedman (Jan 1968).

AF - 24
cs - 95 A formal syntax for transformational grammar. Joyce Friedman and Robert W. Doran (Mar 1968).

AF - 25
cs - 103 Lexical insertion in transformational grammar., Joyce Friedman and Thomas H. Bredt (June 1968).

AF - 33
cs - 108 Computer experiments in transformational grammar. Joyce Friedman (Ed.) (Sept 1968)
I: Fragment from Aspects (Joyce Friedman)
II: Traugott's grammar of Alfredian prose (Joyce Friedman)
III: The IBM core grammar (Joyce Friedman and Theodore S. Martner)
IV: Old English grammar I (Elizabeth C. Traugott)
v: UCLA AFESP case grammar I (Barbara H. Partee)
VI: UCLA AFESP case grammar II (Barbara H. Partee)
VII: A transformational grammar for Swahili (Lorraine Klevansky).

AF - 34 Analysis in transformational grammar. Joyce Friedman and Theodore S. Martner (Sept 1968).

AF - 35 A control language for transformational grammar. Joyce Friedman and Bary W. Pollack (Sept 1968).

AF - 36 Programmers manual for a computer system for transformational grammar. Joyce Friedman, Thomas H. Bredt, Robert W. Doran, Theodore S. Martner, Bary W. Pollack (Sept 1968).