R E P O R T   R E S U M E S

ED 010 873 AL 000 026

COMPUTER-AIDED RESEARCH IN MACHINE TRANSLATION D199, A PARSING PROCEDURE FOR A VECTOR-SYMBOL PHRASE GRAMMAR OF RUSSIAN.

BY- MARTINS, GARY R. SMITH, STEVEN B.
BUNKER-RAMO CORP., CANOGA PARK, CALIF.

REPORT NUMBER BRC-PR-12 PUB DATE DEC 65
EDRS PRICE MF-$0.18 HC-$3.96 99P.

DESCRIPTORS- *RUSSIAN, *COMPUTATIONAL LINGUISTICS, *STRUCTURAL ANALYSIS, SYNTAX, CONTEXT FREE GRAMMAR, DATA PROCESSING, LINGUISTIC THEORY, *COMPUTER PROGRAMS, VECTOR SYMBOL PHRASE GRAMMAR, CANOGA PARK

A COMPUTER PROCEDURE IS DESCRIBED FOR PARSING RUSSIAN SENTENCES WITH A CONTEXT-FREE RECOGNITION GRAMMAR. THIS IS THE FIRST PROJECT UNDER A PROGRAM FOR THE INVESTIGATION OF SEVERAL ASPECTS OF NATURAL LANGUAGE DATA PROCESSING BY FORMALIZED METHODS TO DETERMINE THE USEFULNESS OF FORMALIZED LINGUISTIC TECHNIQUES IN PRACTICAL LANGUAGE DATA PROCESSING APPLICATIONS. EVERY HYPOTHESIS WAS TESTED AS A RUNNING COMPUTATIONAL PROCEDURE BEFORE ACCEPTANCE AS A WORKING PRINCIPLE. THE HARDWARE USED INCLUDED A BUNKER-RAMO MODEL 130 (AN/VYK-1) COMPUTER. THE PROGRAMING LANGUAGE WAS A VERSION OF FORTRAN IV. AN IMPORTANT CHANGE IN THE GRAMMAR AND ALGORITHM OF THE SYSTEM HAS BEEN THE INTRODUCTION OF "GRAMMATICAL VARIABLES" AS COMPONENTS OF GRAMMATICAL LABELS. THE TERM "VECTOR-SYMBOL PHRASE GRAMMAR" IS USED TO DISTINGUISH IT FROM THE MORE USUAL PHRASE "STRUCTURE GRAMMAR." THE BASIC ALGORITHM AND THE FORM AND FUNCTION OF THE GRAMMATICAL VARIABLES ARE DISCUSSED IN DETAIL. THE RESULTS INDICATE THAT THE MODIFICATION OF THE "VECTOR-SYMBOL PHRASE GRAMMAR," WITH NODE SUPPRESSION, MAKES IT POSSIBLE TO UNDERTAKE WRITING A PHRASE STRUCTURE GRAMMAR FOR WRITTEN RUSSIAN SUITABLE FOR DATA PROCESSING APPLICATIONS. (KL)

PROGRESS REPORT NO. 12
Under Contract NSF-C372
With the National Science Foundation
COMPUTER-AIDED RESEARCH IN
MACHINE TRANSLATION
D199

A PARSING PROCEDURE FOR A VECTOR-SYMBOL PHRASE GRAMMAR OF RUSSIAN

U.S. DEPARTMENT OF HEALTH, EDUCATION & WELFARE
OFFICE OF EDUCATION

THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL OFFICE OF EDUCATION POSITION OR POLICY.

PROGRESS REPORT NO. 12
Under Contract NSF-C372
With the National Science Foundation
COMPUTER-AIDED RESEARCH IN
MACHINE TRANSLATION
D199

A PARSING PROCEDURE FOR A VECTOR-SYMBOL PHRASE GRAMMAR OF RUSSIAN

by
Gary R. Martins and Steven B. Smith

U. S. DEPARTMENT OF HEALTH, EDUCATION AND WELFARE
Office of Education
This document has been reproduced exactly as received from the person or organization originating it. Points of view or opinions stated do not necessarily represent official Office of Education position or policy.

December 1965

THE BUNKER-RAMO CORPORATION
8433 FALLBROOK AVENUE
CANOGA PARK, CALIFORNIA 91304

ERIC

ACKNOWLEDGMENT

The work described in this report was supported partly by funds from the National Science Foundation under Contract NSF-C372, monitored for NSF by Eugene Pronko and Richard See of the Information Systems Program, and managed for Bunker-Ramo by Gerhard Reitz. Additional support from corporate research and development funds is also acknowledged.

The authors wish to thank Miss Christine Montgomery for her aid in the earliest days of this research program. Mrs. Marguerite Mazur drafted the sentence diagrams in Appendix C, working from line printer outputs. All secretarial work was performed by Mrs. Joan Arias.

ERIC

TABLE OF CONTENTS

ABSTRACT

A computer procedure is described which performs bottom-to-top, direct-substitution parsing from a vector-symbol phrase grammar. The characteristics of the particular vector-symbol phrase grammar developed under this research program are described. A grammar of this type for Russian is presented and the results of the application of the parsing procedure to Russian sentences are discussed and illustrated. The procedure embodies a generalized method for the recognition and elimination of "trivially ambiguous" structures. Despite its implementation on a relatively small computer, the procedure operates very efficiently.

1. INTRODUCTION

In January of 1965, a program was organized within the Language Analysis and Translation Section of the Research Laboratory of The Bunker-Ramo Corporation for the investigation of several aspects of natural language data processing by formalized methods. As originally conceived, this program was to embrace a number of separate but related projects, each devoted to a different aspect of the overall problem. Eventually, the program was to involve sentence analysis and generation by various methods based upon context-free, context-sensitive, free-rewrite, and transformational linguistic systems.

The first project under this program was dedicated to the development of a context-free recognition grammar for Russian. Work on this project, which has already resulted in a grammar of impressive scope and power, is still in progress; and since our interest in this project has to date precluded the development of other projects under this program, the remainder of this report will be concerned almost exclusively with the technique that has been developed for parsing Russian sentences with a context-free grammar, and with the grammar itself.

1.1 RESEARCH POLICY

From the very inception of this program, it has been our policy to subject ourselves to the rigorous discipline of operational demonstrability in all phases of the work in progress. That is to say, every hypothesis concerning the organization of the processing algorithms and the structure and context of the grammar has been tested as a running computational procedure before gaining full acceptance as a working principle. This general policy has been observed with respect to all levels of detail of the work, from the broadest to the most specific—involving at the lowest level the testing of individual grammar rules.

The adoption of this policy required that a very short research cycle (design, test, evaluate) be maintained, in which the temporal distance separating the linguistic research group from the computer center is minimized. The latter goal has been achieved by carrying out the greater part of the project work in the computer center, the linguistic research personnel performing all of the required programming and equipment operation tasks themselves (generally during second and third shift periods). The uncommon luxury of such "hands-on" working conditions has been made possibly by the in-house availability of a corporate-utility research computer facility. Many of the features of the linguistic processing system developed under this project have been designed specifically to exploit this advantage to the fullest; in a batch-processing or production environment much of the input/output and peripheral utility programming would have taken on a markedly different form.

1.2 THE RESEARCH COMPUTER CENTER

1.2.1 Hardware

Another source of program design constraints has been, quite naturally, the hardware available in the computer facility. This consists of the following:

- one Bunker-Ramo Model 130 (AN/UYK-1) computer; this is an 8K, 15-bit, parallel binary, stored micrologic machine with a basic read-generate cycle time of 6 usec.

- 8K cells of directly-addressable extended memory

- one 120-character line printer (BR-282)

- one magnetic tape controller (BR-192) and two transports (BR-170)

- one input/output controller (BR-143), with associated typewriter, paper-tape reader and punch, and card reader

Many of the details of organization of the operating programs—and even of the grammar—have been chosen with this hardware system's limited memory and intermediate running speed (as a result of the micrologic programming feature) in mind.

1.2.2 Software

A third influence on the present form of the algorithm and utility programs has been the availability of a version of the FORTRAN IV programming language in the computer center's software library. The experimental nature of the projects to be undertaken made it advisable to sacrifice the efficiency of machine-language programming, at the outset, in favor of the ease of programming and debugging with a procedure-oriented language; while a symbol or string processing language (such as COMIT) might have been preferable for this purpose, the ready availability of FORTRAN, and the familiarity of the research group with its use, combined to bring about its adoption for the initial development of the language data processing algorithms and utility programs. As a consequence, some of the characteristics of these algorithms derive ultimately from their initial expression in an arithmetic-oriented programming language.

The reader will find many of the details of the exposition that follows easier to understand if these three sources of design limitations are taken into account.

2. THE CONTEXT-FREE PARSING PROJECT

The program for the study of formalized language data processing systems began with the project for the parsing of Russian sentences with a context-free algorithm and grammar. Several motives contributed to the decision to initiate such a project. One of the strongest of these was the academic interest, on the part of some of the members of this department, in formal linguistic systems. Another, related to this but of a more utilitarian turn, was a desire to determine the usefulness of formalized techniques in practical language data processing applications such as machine translation, information retrieval, etc. Moreover, at the time of its inception, this project was relevant to departmental contractual obligations.

The second of these motives deserves further elaboration. There are several characteristics of formalized linguistic procedures which seemed, at least superficially, to offer substantial advantages in practical language data processing operations. The much discussed separation of grammar and algorithm is one such characteristic which promised particular benefits for applications systems subject to continuing modification through research. Another apparent advantage of formalized linguistic procedures which was of interest to the research group is the requirement of formal "homogeneity" in the grammatical specifications (or rules). A third such advantage is the relative simplicity of programs embodying the basic processing algorithms. The fundamental justification for our inception and continuation of this research program has been the testing of these (and other) apparent advantages for practical language data processing applications; some conclusions concerning these topics, derived from the context-free parsing project, are presented later in this report.

2.1 THE CONTEXT-FREE PARSING SYSTEM

From the beginning of this project to the time of this writing, the overall context-free parsing system has been undergoing a continuous evolutionary development. But for a need to save time, space, and the reader's patience, a historical description of the system would be the most natural kind of treatment. It will be more practical, however, to write only a few words about the system's humble beginnings and then to describe in limited detail its present state.

Originally, it was decided that the relatively simple and efficient bottom-to-top direct-substitution procedure, working with binary phrase-structure rules, would serve our needs very well. The first task performed under this project was, accordingly, the programming, in FORTRAN, of such an algorithm and associated input/output subroutines. This first system was, in all respects, rather primitive. It provided for a grammar of quite modest proportions (250 rules, each of an ordered triple of integers ranging from 000 to 999) on which a crude sequential search was performed during look-up operations. A very limited kind of output was provided. Input capacity was restricted to short strings with a few grammar codes per item. Program flow was relatively inefficient, and it tended to run slowly. These deficiencies have since been remedied, one by one; and an efficient and quite powerful sentence analysis system has evolved.

In the course of this development, significant changes have been made in the grammar and algorithm of the system. The most important of these changes has been the introduction of grammatical variables as components of grammatical labels in order to enhance the discriminatory power of the system. In his latest publication on linguistic theory[1], Noam Chomsky argues,

[1] See Aspects of the Theory of Syntax, M.I.T. Press, Cambridge, 1965, pp. 67, 89, 90, 210, 211, and 215. For purposes of consistency, our use of the term "phrase-structure" in this report has been limited to accord with Chomsky's interpretation of it.

in effect, that this modification removes the system from the class of phrase-structure systems to the class of transformational systems. Chomsky's argument on this point is not entirely convincing, however, particularly in view of the radical distinctions between the formal properties of this system (and others like it) and those of transformational systems as discussed in the rather ample literature on the subject. Nevertheless, to avoid the possible accusation of abusing the term "phrase-structure", the present system will be referred to as a vector-symbol phrase grammar[1] system; this name is suggested by the modified form of the system's grammatical labels, which consist of ordered triples of symbols.

To simplify the presentation of the system's details, a full discussion of the form and function of the grammatical variables will be given after the operations of the basic (phrase-structure) parsing algorithm have been presented. It will then be shown in precisely what fashion the grammatical variables restrict these operations. This order of presentation is intended not only to simplify matters for the reader but also to reflect the evolutionary history of the system itself.

2.1.1 The Algorithm

2.1.1.1 Description

The structure of the basic parsing algorithm[2] is very simple, yet difficult to describe clearly in plain English. For this reason, the following brief verbal description is supplemented with a gross

[1] abbreviated VSPG.

[2] Of the kind known widely as the "Cocke algorithm" after John Cocke of IBM. We are indebted to Martin Kay of the RAND Corporation for most helpful discussions and advice at the outset of this project.

schematic flowchart on pages 10 and 11.

This algorithm performs elementary list-processing operations on a group of four interrelated linear arrays representing the developing tree structure(s) assigned to the input string. If these arrays are regarded as a single two-dimensional array, then we may say that each row of the array represents a single node of one or more of the trees; the columns of the array specify, for each row, (1) the position in the input string, counting from left, of the leftmost terminal dominated by the node corresponding to the row, (2) the grammatical label corresponding to the node, (3) the number of the row corresponding to the left-hand daughter of the node, (4) the number of the row corresponding to the right-hand daughter of the node. A set of counters are maintained as pointers during operations on the rows, while the columns are addressed by name. A compact table is kept of the row numbers of the earliest and latest nodes dominating substrings of a given length.

As the algorithm runs, rows corresponding to progressively longer well-formed substrings are added to the initial array. The procedure begins with attempts to combine, as left-hand and right-hand daughters of a single node, rows dominating substrings of length 1 (i.e., single input items). Such a combination can be made if and only if the substrings dominated by the daughter-candidate rows are adjacent <u>and</u> the grammar contains one or more rules providing for the combining of the daughter-candidate rows' grammatical labels. If both conditions are met in a given instance, then a new row is added to the array for each applicable grammar rule. When all the possibilities for forming substrings of length 2 (by combining substrings of length 1) are exhausted, a new series of attempts are made to form substrings of length 3 (combining, in both orders, substrings of length 1 with substrings of length 2). Next, the formation of substrings of length 4 is attempted—requiring exhaustive testing of substrings of length 1 and 3, 2 and 2, and 3 and 1. And so on, until no further combinations are possible. When this condition is met, the latest row(s) entered in the basic array will represent the longest well-formed substrings in the input string; if all has gone well, the latest row(s) will correspond to nodes dominating the entire input string and will carry the grammatical label "Sentence". The flowcharts and explanatory notes, on the next three pages, show the structure of the basic algorithm in some detail.

This algorithm has recently been reprogrammed in machine language for the Bunker-Ramo 130 computer, resulting in a very substantial gain in operating speed over that obtained with the FORTRAN version. All input/output subroutines are still expressed in FORTRAN, however, simply for the sake of convenience of modification if and when the need arises.

2.1.1.2 Flowcharts of the Basic Parsing Algorithm

2.1.1.2.1 Explanation of Symbols Used in Flowcharts on Following Pages. In the following basic system flowcharts, the four parsing list arrays are referred to as follows:

INW_i	position in the input string, counting from the left, of the leftmost terminal dominated by the node corresponding to row i
GL_i	grammatical label of the node corresponding to row i
LC_i	row number of the left daughter of the node corresponding to row i
RC_i	row number of the right daughter of the node corresponding to row i

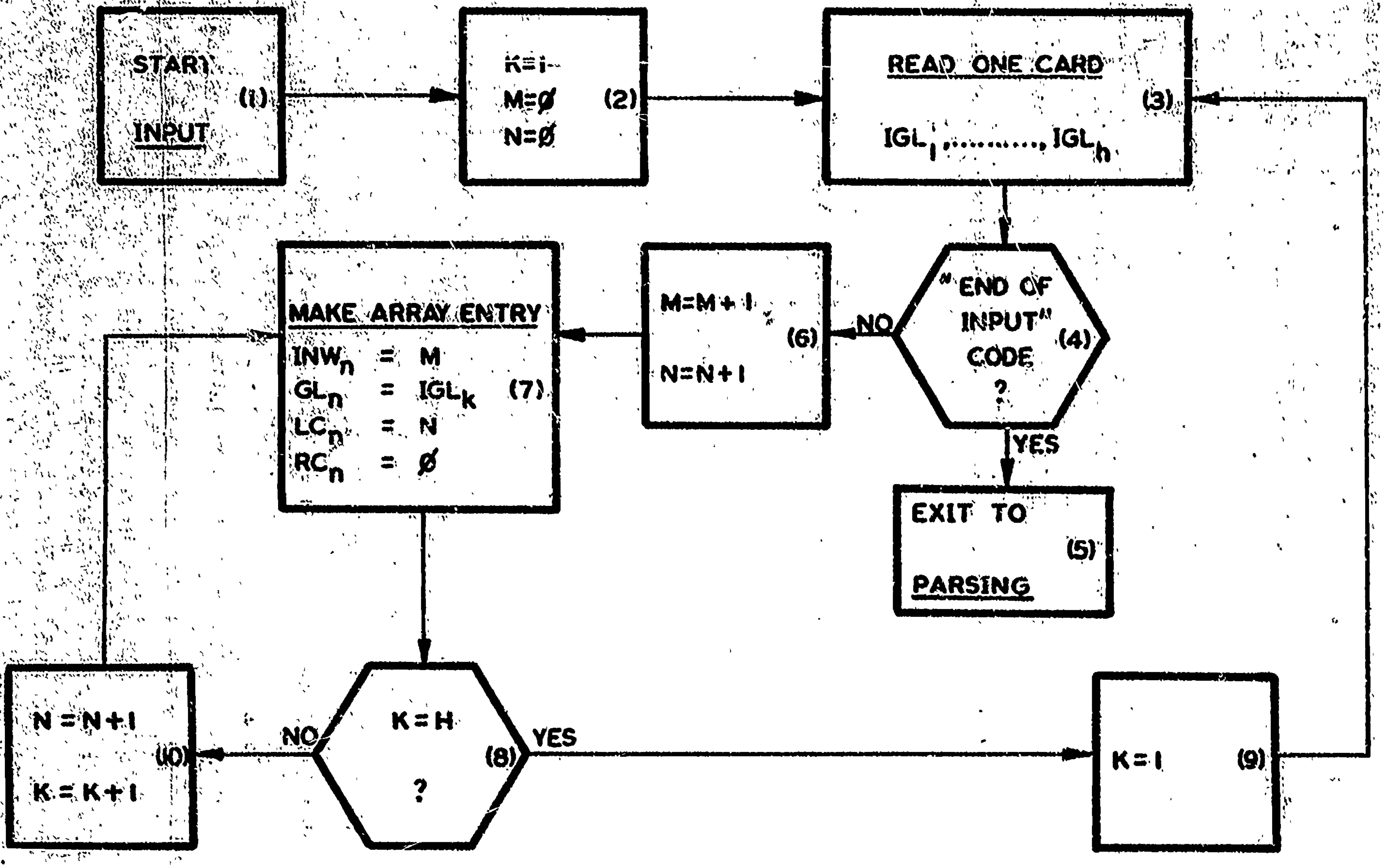
A number of counters and pointers (lower-case when subscripts) appear:

M	counter for items in the input string
N, n	counter for rows of the parsing list arrays (or nodes)
H, h	counter for input grammatical labels in INPUT subprogram
L, l	pointer for earliest right-hand daughter-candidate
K, k	pointer for input grammatical labels
I, i	pointer for left-hand daughter-candidates
J, j	pointer for right-hand daughter-candidates

Other symbols are:

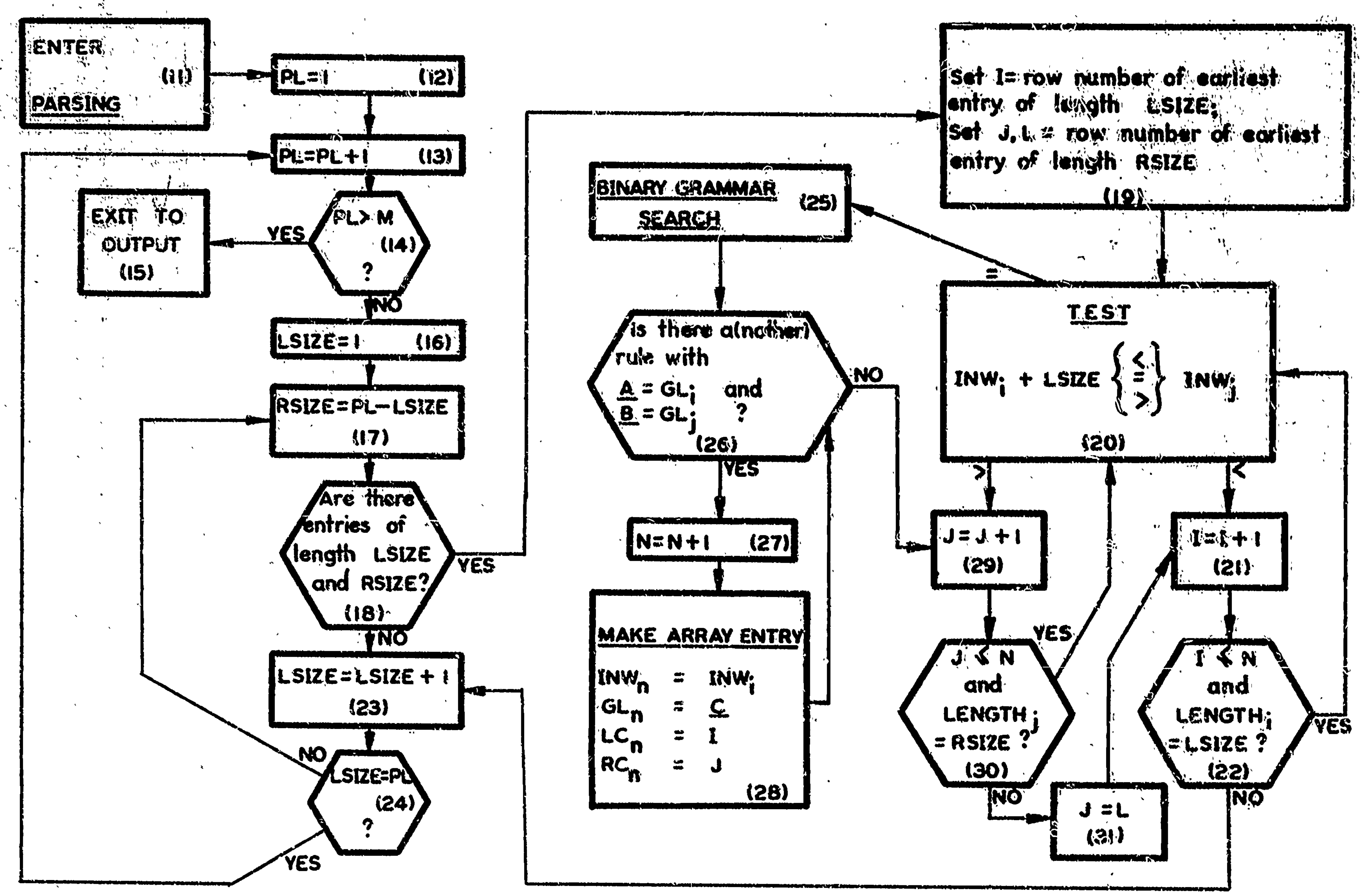
IGL_i	the i-th input grammatical label for a given input item
PL	phrase length; the length of substrings currently being formed
LSIZE	the length of the left-hand component of the substring under test
RSIZE	the length of the right-hand component of the substring under test
$LENGTH_i$	the length of the substring dominated by the node corresponding to row i

Components of grammar rules are referred to as A, B, C; it is assumed that the rules themselves are of the form $\underline{A} + \underline{B} \rightarrow \underline{C}$. Two flow diagrams are presented; the first shows how the input data are initially arranged in the parsing array. Box (3) of the INPUT diagram involves the reading of a single punched card, representing a single input word, having h distinct grammatical labels. The second diagram describes the entire basic parsing strategy which commences when the input operations are terminated.



INPUT SUBPROGRAM FLOWCHART

2.1.1.2.2



PARSING SUBPROGRAM FLOWCHART

2.1.1.2.3

2.1.1.3 Modification of the Algorithm

As has been mentioned, substantial modifications have been introduced into the system. The two most important of these are described in this section. The first, which restricts the strong generative capacity of the system, was motivated by the observation that a context-free phrase-structure system whose grammar contains labels that are both left and right recursive will produce practically irrelevant ambiguous structural descriptions. The second, and more fundamental, of these modifications introduced vector-symbols into the grammar as phrase markers and required a corresponding supplementation of the algorithm to manipulate them. The first modification affected only the algorithm; the second, since it has affected both grammar and algorithm, is discussed below with respect to its effects on the parsing strategy and in a later section (2.1.2) with respect to its effects on the grammar itself.

2.1.1.3.1 The Node Suppression Option. The first of these special features, the node suppression option, has been adopted to deal with the tendency of context-free parsing systems of the present kind to derive trivially different structural descriptions for even relatively simple input strings under certain conditions. This may occur only when a particular grammatical label is both left and right recursive in the grammar.[1] Let us consider four ways in which this condition may occur:

1. A single recursive rule of the form $\underline{A} + \underline{A} \longrightarrow \underline{A}$ makes the grammatical label A both left and right recursive. Strings of the form $\underline{A} + \underline{A} + \underline{A} + \ldots + \underline{A}$ will be analyzed ambiguously with this rule.

---

[1] This discussion is limited in application to systems with binary grammars; space does not permit the development of the fully generalized case. Specifically excluded from this discussion is the case of the ambiguous grammar (i.e., a grammar containing more than one rule for rewriting a given pair of candidate labels).

2. The two recursive rules (1) <u>A</u> + <u>B</u> → <u>A</u>, and (2) <u>C</u> + <u>A</u> → <u>A</u> make the label A left recursive and right recursive respectively. Strings of the form <u>C</u> + <u>C</u> + <u>C</u> + ... + <u>C</u> + <u>A</u> + <u>B</u> + <u>B</u> + <u>B</u> + ... + <u>B</u> will be analyzed ambiguously with these two rules.

3. The left and right recursive condition may derive from the combination of a recursive rule and a cycle of rules (of indefinite length). For example, the three rules (3) <u>A</u> + <u>B</u> → <u>A</u>, (4) <u>C</u> + <u>A</u> → <u>B</u>, and (5) <u>X</u> + <u>Y</u> → <u>C</u> will provide ambiguous analyses for strings of the form <u>A</u> + <u>X</u> + <u>Y</u> + <u>A</u> + <u>X</u> + <u>Y</u> + ... + <u>A</u>. In this case, the label <u>A</u> is left recursive by virtue of the cycle consisting of rules (3) and (4).

4. A set of non-recursive rules may fulfill the left and right recursive condition through cycles. For example, the non-recursive rules (6) <u>B</u> + <u>C</u> → <u>A</u>, (7) <u>A</u> + <u>X</u> → <u>B</u>, (8) <u>Y</u> + <u>A</u> → <u>C</u> provide the label <u>A</u> with both left and right recursiveness through two cycles consisting of the rules (6) and (7), and (6) and (8). In this case, strings of the form <u>A</u> + <u>X</u> + <u>Y</u> + <u>A</u> + <u>X</u> + <u>Y</u> + ... + <u>A</u> will be ambiguously analyzed by the given rules.

Other sets of conditions are possible, of course, but these four examples are sufficiently varied to facilitate a brief discussion which can easily be generalized. In our present operations, ambiguous structural descriptions resulting from conditions such as those in 1., 2., and 3., above are considered <u>trivially different.</u> More specifically, alternative structures resulting from the application to a given string of the same set of rules, but in a different order, are considered trivially different only when the label dominating the string is both left and right recursive in the applied rule set and is recursive in a recursive rule in the applied rule set.

In 4. above, it will be seen that these conditions are not met inasmuch as the rules given are all non-recursive.[1] It is an empirically based judgment of ours that alternative structural descriptions occurring under these conditions are potentially more interesting for purposes of grammatical research than those covered by the formulation in the preceding paragraph. There is a formal correlative to this judgment which is illustrated by the contrast between situations 3. and 4. above. Note that, in both cases, the sample string for which alternative descriptions are provided by the rules is the same, namely $\underline{A} + \underline{X} + \underline{Y} + \ldots + A$, etc. This string type may be described as a repetition of the recursive label with a fixed substring interposed between the repetitions. Now, in 3., the fixed interposed substring (i.e., $\underline{X} + \underline{Y}$) is reducible by itself to a single structure (i.e., $\underline{C}$). This same substring in 4., however, is not reducible to a single structure and consequently its relationship with the recursive label will be more complicated and, quite possibly, more interesting.

Practical considerations have forced the adoption of some means to keep these trivially different alternative structures out of the parsing lists. On the one hand, their explosive combinatorial growth as a function of string length is disastrous for both available memory space and program running time; on the other, they result in the production of reams of uninformative printout and would constitute a serious obstacle to grammatical research.

Several solutions to this kind of problem are available. It could be solved by eliminating recursive labels from the grammar altogether, or by eliminating all labels which are both right and left recursive from the grammar. We regard these solutions as

---

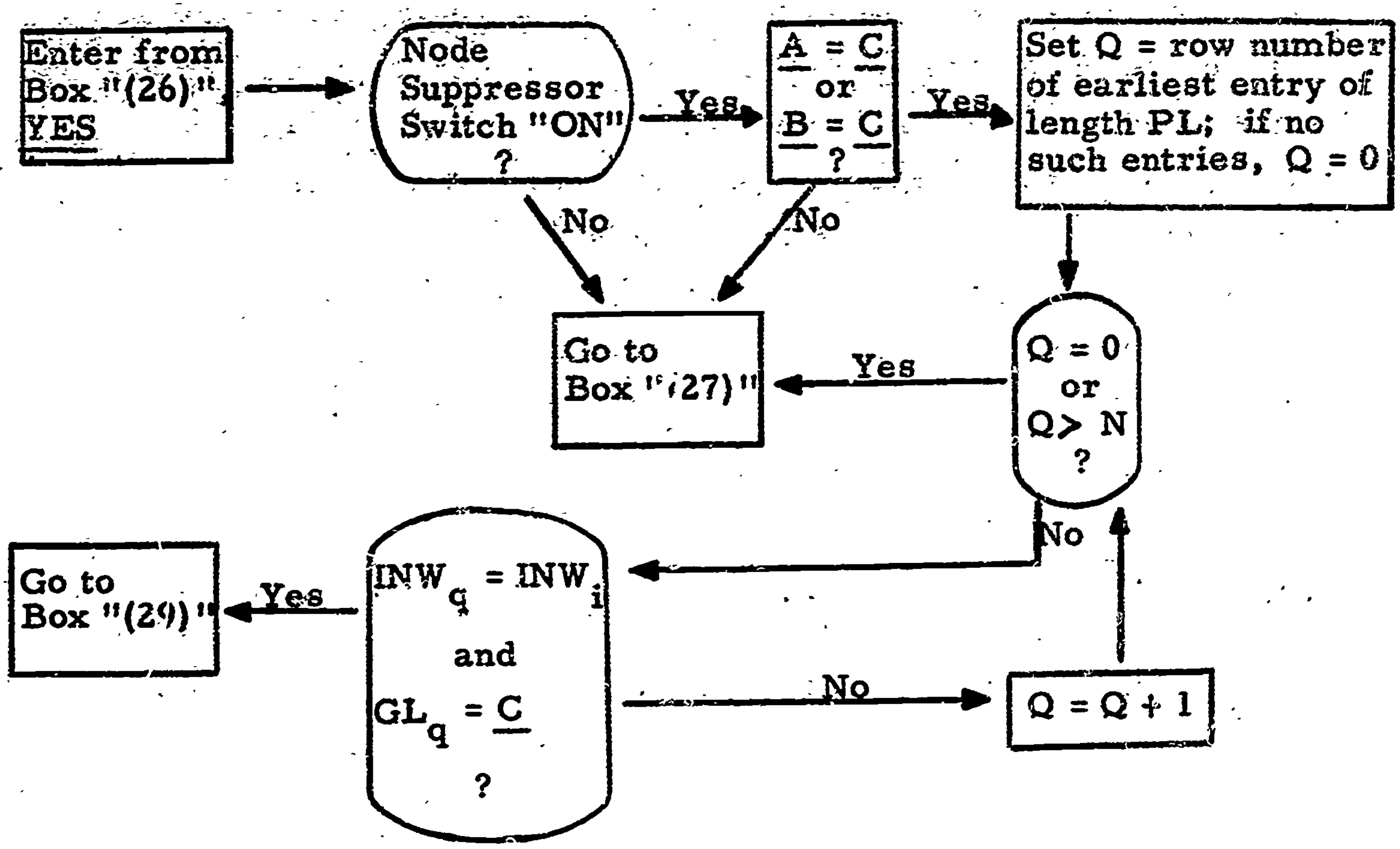
[1] In the present grammar, there are no labels which are both left and right recursive solely by virtue of cycles of non-recursive rules.

both practically and theoretically undesirable, because the former solution, by eliminating recursiveness altogether, leaves a grammar of finite generative capacity, while the latter would require the addition to the grammar of a great many rules to maintain its weak generative capacity. Another kind of solution would involve the imposition of some order on the applicability of the rules in the grammar; we have avoided this in order to maintain the independence of grammar rules from one another as an aid to experimentation. Finally, because of the unusually stringent constraints on memory utilization, all solutions requiring the maintenance of supplementary lists during parsing operations were ruled out.

Instead, a toggle-switched optional subroutine was added to the basic algorithm which recognizes trivial ambiguities in the making and discards them. This optional node suppression subroutine fits into the flowchart on page 11 between boxes (26) and (27). It ascertains, first of all, whether the grammar rule to be applied is of the recursive type (i.e., in the terminology of the flowchart, whether <u>A</u> or <u>B</u> is identical with <u>C</u>); if it is <u>not</u>, processing continues (to box (27)) as usual. If the rule is recursive, then a check of all parsing list entries of length PL is made to determine whether any one of these has the same entries in the INW and GL columns as would the potential new entry. If this condition is met, the potential new entry is discarded, and an exit is made to box (29); if this condition is not met, processing continues as usual. This subroutine may be represented thus:[1]

---

[1]It may be of interest to note that this technique achieves precisely the same economies in parsing list reduction as does that proposed by Jane Robinson of the RAND Corporation (in <u>Endocentric Constructions and The Cocke Parsing Logic, presented to the International Conference on Linguistics</u>, New York, May 1965), but without the overt marking and arrangement of recursive rules.



2.1.1.3.2. Vector-Symbol Phrase Markers. A second, and more interesting, modification of the basic algorithm was adopted to deal with a problem which is particularly severe when dealing with highly inflected languages such as Russian. The problem is essentially that a context-free phrase structure grammar for such a language will contain many subsets of rules, dealing mainly with agreement and government situations, which closely resemble the paradigms of case, number, and gender presented in classical grammars. For instance, to link up an adjective with a following noun in a sentence from such a language, an imposing array of rules, having the following general form, are required:

$A_{masc., sg., nom.}$ $+ N_{masc., sg., nom.}$ — $NP_{masc., sg., nom.}$
$A_{masc., sg., gen.}$ $+ N_{masc., sg., gen.}$ — $NP_{masc., sg., gen.}$

.
.
.

$A_{masc., pl., nom.}$ $+ N_{masc., pl., nom.}$ — $NP_{masc., pl., nom.}$

.
.
.

$A_{fem., sg., nom.}$ $+ N_{fem., sg., nom.}$ — $NP_{fem., sg., nom.}$

.
.

and so on.

There is something inelegant about such collections of rules and, more importantly from a practical point of view, they consume a tremendous amount of storage space. Moreover, these sets of rules constitute an impediment to the kind of research in which we have been engaged—elementary modifications to the grammar would often involve the rewriting of entire sets of such rules. The solution finally adopted was suggested by the label + subscript notation employed in the illustration above; it consists of splitting each component of a grammar rule into two parts—a fixed part, denoting a major grammatical category, and a variable part, denoting sublcasses within the major category.

2.1.1.3.2.1 Numerical Format. Some preliminary explanation is necessary before the details of this vector-symbol (VS) arrangement can be presented. It has been mentioned that the word length of the BR-130 computer is 15 bits, and that the basic algorithm was initially programmed in FORTRAN, using integer arithmetic. There was therefore available a range of non-negative numbers from zero through $16,383)_{10}$ [1] for the representation of grammatical labels. In deciding how to implement a more compact arrangement of grammar rules of the kind mentioned, the question of how to accomplish the task within these numerical constraints arose. For a variety of reasons, we finally decided to employ the units and tens positions of the numerical grammatical labels for the expression of "paradigmatic variations", and to employ the three most significant digits as fixed major category tags. To permit full utilization of the variable digits, it was, of course, necessary then to restrict ourselves to the range 00000 through 16299 for the expression of grammatical labels. The fixed major category tags would then vary from 000 through 162, while the variable or "suffix" tags could range from 00 through 99.

Finally, in order to achieve maximum flexibility in the overall arrangement, the two variable or "suffix" digits were effectively split off from one another, so that they might be manipulated independently. This gave the result that each component of a grammar rule consisted of three segments: (1) a major grammatical category tag, represented by a non-negative integer from 000 through 162, (2) a suffix variable, ranging from 0 through 9, (3) a second suffix variable, ranging from 0 through 9. The grammatical label 12374 could thus be

[1] Unless explicit mention is made to the contrary, all numbers in the remainder of this report will be in decimal notation.

thought of as consisting of the segments 123-7-4. The convenience of such a convention is that it permits one of the suffix variables, when attached (for example) to the major tag for Noun, to represent gender and number, while the other suffix tag might represent case.

2.1.1.3.2.2 A Formalism Governing the Application of Grammatical Variables. In order to take advantage of this schema to render more compact the list of grammar rules, a formalism was required which would permit the replacement of a paradigmatic subset of rules by a single "cover" rule. To illustrate the full set of conventions finally adopted, the following notation will be used:

A. Grammar rules will be represented as

$$\underline{A}V_{a_1}\ V_{a_2} + \underline{B}V_{b_1}\ V_{b_2} \longrightarrow \underline{C}V_{c_1}\ V_{c_2},$$

where $\underline{A}$, $\underline{B}$, and $\underline{C}$ represent major category tags, and the multiply-subscripted V's represent the suffix variables appended to them.

B. To represent grammatical labels of neighboring nodes (candidates for rewriting), the same arrangement will be used, except that "A", "B", and "C" replace their underlined counterparts, and the symbol "T", with subscripts as above, will be used instead of "V"; thus, $AT_{a_1}\ T_{a_2} + BT_{b_1}\ T_{b_2}$ will represent a pair of adjacent nodes for which an appropriate grammar rule is being sought; $CT_{c_1}\ T_{c_2}$ will represent the grammatical label derived from the application of a rule to the foregoing pair.

C. The symbol "x" ranges over a, b ; "i" and "j" range over 1, 2 . The formalism has two parts: the first governs the testing of the applicability of a given rule to a given candidate-pair:

I.	If $A = \underline{A}$ and $B = \underline{B}$, the rule applies, except that,	
I.a.	$0 < V_{x_i} < 9$	requires that $T_{x_i} = V_{x_i}$
I.b.	$V_{a_i} = V_{b_i} = 0$	requires that $T_{a_i} = T_{b_i}$

The second part of the formalism specifies the composition of the grammatical label derived from the application of a rule under the above conditions:

II.a.	$C = \underline{C}$	
II.b.1.	$V_{c_i} \neq 0$	requires that $T_{c_i} = V_{c_i}$
II.b.2a.	$V_{c_i} = V_{x_i} = 0$	requires that $T_{c_i} = T_{x_i}$, otherwise
II.b.2b.	$V_{c_i} = V_{x_j} = 0$	requires that $T_{c_i} = T_{x_j}$ $(i \neq j)$

2.1.1.3.2.3 Explanation of the Formalism. The net result of these conventions can be roughly summarized in plain English as follows:

1) When any digit other than "0" or "9" appears in a suffix-variable position of a grammar rule component, on the left of the rewrite sign, a match is required between that digit and the corresponding digit of the appropriate candidate node label.

2) When the digit "9" is used in a suffix-variable position of a grammar rule component, on the left of the rewrite sign, then no matching at all is required in that position.

3) When the digit "0" is used in corresponding suffix variable locations in BOTH grammar rule components on the left of the rewrite sign, then the corresponding two digits in the candidate nodes must be identical: otherwise the effect of a variable suffix "0" is the same as that of a "9".

4) When the digit "0" is used as a suffix variable in the output component of a grammar rule (i.e., on the right of the rewrite sign), then it indicates that suffix information is to be carried upward from either or both of the candidate nodes to the new parent node; the location(s) of one or more zeros, in suffix positions to the left of the rewrite sign, indicate where the information carried upward is to be taken from. The existence of "0" in a suffix position of the output component of a rule always requires that there be at least one "0" in a suffix position on the left of the rewrite sign of the same rule.

2.1.1.3.2.4 Examples. To illustrate the usefulness and clarify the mechanics of these conventions, let us consider two hypothetical cases.

Suppose we need to write a rule covering the government of a noun (001-) by a preposition (025-). It will be necessary to determine that the noun is of the proper case. This could be accomplished by writing a series of rules, one for each case, gender, and number situation—this was, in fact, required before the algorithm was modified. But, if we allow the first suffix variable with nouns to represent gender and number, and the second to represent case, and allow the second suffix variable for prepositions to represent case also, then a single rule: 02590 + 00100 ⟶ 03700, suffices for all instances. Assuming that a neighboring pair of nodes with labels 02563 and 00173 are encountered, we can test the applicability of the above rule and, if applicable, determine the composition of the new parent node by referring to the formalism given on page 19. To make this easier, the components of the rule and of the node labels can be segmented and juxtaposed:

GRAMMAR RULE:	025-9-0	+	001-0-0	⟶	037-0-0
NODE LABELS:	025-6-3		001-7-3		037-7-3

Referring to rule I. of the formalism, we test whether the major grammatical tags in the rule are identical with those of the node labels; since they are, we go on to rule I.a., which is seen to be inapplicable. Rule I.b. applies for i = 2, requiring that the second suffix variables of the node labels be identical; this is seen to be the case (both are "3"). This last test amounts to a test for "agreement" between the case specification of the preposition and that of the noun; at this point, since no more test conditions are listed, the rule is found to be applicable.

To determine the composition of the resultant parent node, we refer first to rule II.a. of the formalism; this enables us to obtain 037- as the major grammatical tag of the parent node. Rule II.b.1. is seen to be inapplicable. Rule II.b.2a. applies for x = b and i = 1, enabling us to fill out the resultant label to 0377-. Rule II.b.2a. also applies for i = 2 and x =(a or b), enabling us to determine the final digit of the resultant label, giving 03773 as the final result. Rule II.b.2b. is not consulted, since the task is already complete.[1]

[1] It may be of interest to the reader to note that it has not been found necessary or desirable as yet to include, in the C-F grammar for Russian, rules of a form that would require the application of rule II.b.2b. of the formalism; this rule has nonetheless been embodied in the algorithm for reasons of programming symmetry and because it seemed inadvisable to preclude the use of such rules if needed.

It will be noted that, were the noun in the above example not of the proper case (e.g., if it carried the label 00176), then the rule could not have been applied because of the condition expressed by rule I.b. of the formalism. Observe also that the first suffix variable in the preposition's label (i.e., "6") is ignored altogether because of the "9" in the corresponding position in the grammar rule. Similarly, the gender-number suffix variable in the noun's label (i.e., "7") is not considered in testing the applicability of the rule—the reason for this is given in 3) of the notes on page20.

Let us consider one more illustrative example of the mechanics of the modified algorithm. Suppose we have a grammar rule: 01205 + 11604 $\longrightarrow$ 06313, and a neighboring pair of nodes in the developing tree structure with labels: 01265, 11664. Rule I. of the formalism is satisfied. Rule I.a. holds for x=a, i=2 and also for x=b, i=2; in both cases, it is satisfied. Rule I.b. holds for i=1, and it, too, is satisfied. The rule is therefore said to be applicable to the candidate-pair. Rule II.a. tells us that the resultant parent node label will begin 063-. Rule II.b.1 applies for i=1 and also for i=2, yielding a final resulting node label 06313. In this instance, "agreement" was required between the candidate node labels in the first suffix position only; no suffix information is carried upward to the parent node label.

2.1.1.3.2.5 Remarks. As complex as the formalism, the informal explanation, and the examples may make this modification to the algorithm appear, in practice it becomes simple in the extreme; and its adoption has resulted in a grammar that is both greatly reduced in size and considerably more elegant than it could otherwise be. A beneficial side effect of this technique has been to greatly enhance the mnemonic value of individual grammar rules, which eases the interpretation

ERIC

of experimental results for the research linguists. This is due to the similarity between the VS format of the grammatical labels and the familiar stem-suffix structure of inflected Russian forms, a similarity that has been heavily exploited in the assignment of grammatical labels.

In relating this modification of the algorithm to the flowchart on page 11, the subroutines corresponding to rule I of the formalism should be included in box (26), while those corresponding to rule II belong to box (28).

One rather different approach to dealing with the problem of inefficiency inherent in context-free phrase-structure grammar rules is known to the authors.[1] Briefly, it involves the use of grammatical variables whose status is tested against stored tables of conditions which are chained together in a highly flexible manner. We have not adopted this interesting technique, because its implementation would tax the memory capacity of the hardware system at our disposal, and also because it seemed preferable for our purposes to express the entire significance of each grammar rule in a single formula—this enhances the mnemonic value of the rules and imposes a somewhat tighter structure on the grammar itself.

2.1.1.4 Input/Output Conventions

2.1.1.4.1 Grammar Rules, Input and Output. The rules of the grammar are punched, one rule per card, on standard 80-column data cards. Each component of a rule is punched as a 5-digit decimal number, with preceding zeros where required. The resulting 15 digits occupy columns 7 through 21 of the card, the remainder of the card being irrelevant (i.e., it is ignored by the input subprogram). The grammar rule 03165 + 12102 ⟶ 00321 would be punched 013651210200321, beginning in column 7 of the grammar rule card.

[1] This is due to Martin Kay of the RAND Corp. Economical schemes have been developed, also at RAND, for categorial and dependency grammars, but these deal with a rather distinct set of problems and are not relevant to this discussion. The system proposed by G. H. Harman (in "Generative Grammars Without Transformational Rules: A Defense of Phrase Structure", Language, 39, pp. 597-616) is very similar to a VSPG as described here except that (1) it employs discontinuous structures, and (2) the grammatical variables are not ordered.

The cards containing the grammar rules are assembled manually into a deck which is in numerical ascending order. In the ordering, only the major grammatical category tags of the first two components are considered; these are treated as though they formed a single continous number from the left. For purposes of ordering the rule deck, therefore, the card containing the sample rule above would have a value of 013121. Consequently, it would appear nearer the top of the deck than a card bearing the rule 02513 + 16299 → 16100 (whose "ordering value" is 025162).

A special card is placed on the bottom of the deck as a signal to the input subroutine that it has finished its task. When this card is read, input operations terminate and a special subroutine is called which performs two functions: (1) it ascertains that there are no errors in ordering of the rules, and (2) it marks the boundaries of "families" of grammar rules by converting the leftmost component of the last-received rule of each family from a positive to the corresponding negative integer (see examples in Appendix D.

It is well known that, for lists longer than a fairly small threshold size, search operations can be most efficiently accomplished by the binary search technique. This technique requires that the list to be searched be numerically ordered— it is for this reason that the grammar rule deck is ordered in the fashion described above. (Box (25) of the flow diagram on page 10 indicates the place of the binary grammar search in the parsing algorithm.) Because of the method of ordering the grammar rules, it is possible that several rules will have the same "ordering value"; such a group of rules constitutes a "family" of rules (within which the ordering is entirely arbitrary). It will be noted that a group of rules constituting a family share the property that if, in a given instance, any one of them satisfies rule I. of the formalism (page 20), all of them will satisfy it.

For this reason, rules are checked for applicability in family groups (see box (26) of the parsing flow diagram, page 11). The purpose of the binary search, then, is to locate appropriate families of rules.

Since the ordering of rules within a family is arbitrary, a simple procedure is required which will guarantee that none of its members is overlooked in testing. In other words, it is necessary to be able to identify the boundaries of family groups in the rule list. By marking these boundaries in the manner described above, this is accomplished with an irreducibly minimal expenditure of running time.

The grammar rule list is output via the line printer in the form shown in Appendix D.

2.1.1.4.2 Sentence Input and Output. Sentences to be parsed are keypunched, one "word" per card, on standard 80-column data cards. Columns 1 through 24 of the card contain an alphanumeric transliteration of the Russian "word" (see Progress Report No. 6, page A-14). Columns 25 and 26 contain a two-digit decimal number, with preceding zero where necessary, specifying the number of distinct grammatical labels assigned the input item; this number may range from 01 through 20 (it corresponds to the subscript h in box (3) of the input flow diagram). Column 27 is left blank. The remaining columns, from 28 through 72, in fields of 5 columns, contain up to nine distinct grammatical labels. If the "word" has been assigned more than 9 such labels, the remainder are punched on a second card, in fields of 5 columns, from column 7 through column 61. A typical input "word" card appears as follows:

FORMAT: transliteration; number of labels; 1st label; 2nd label; 5th label

CONTENTS: PARTII 05 0012200123001260014100144

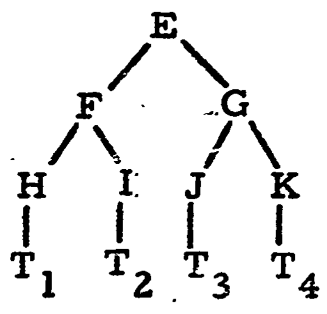
COLUMN: 1 25 28 33 38 43 48 52

An input "word" need not necessarily correspond, it must be pointed out, to an ordinary Russian text word, as in the above example. We have defined the notion "input word" to include anything which (1) can be determined by purely mechanical means from the Russian text, and (2) is most conveniently handled in the grammar as a separate entity. This definition has, in the course of our research, included such things as ordinary text words, punctuation marks (comma, period, dash), initials and names ("USA", "N", "S", Khrushchev, etc.), and artificial elements ("sentence-begin", "comma-follower", etc.) such as are discussed in section 2.1.2 of this report.

The individual input cards for a sentence are assembled into a sentence input deck in natural text order. A special "end-of-input" card is added to this deck to trigger the transfer from the input subroutine to the parsing algorithm proper (see box (4) of the input flow diagram). As each input card is read into the computer, the grammatical labels it contains are arranged (as shown in diagram 2.1.1.2.2) in the initial parsing array. At the same time, the input items are numbered and printed via the line printer in the form shown in Appendix B. The alphanumeric transliterations are also stored where they can be later recalled for use in printing out the results of the parsing operations. Needless to say, these transliterations play no part whatsoever in the parsing procedures—they merely represent data upon which a dictionary-search subprogram might act in an operational environment, and they serve to render the parsing system's output more readable.

2.1.1.4.3 Output of Parsing Results. Several kinds of displays of parsing results can be selected by means of toggle-switch settings. One of these is a line printer display of the initial parsing array as prepared by the input subprogram (diagram 2.1.1.2.2). Appendix B, pages B-2 and B-3, shows a line printer display of the final parsing array after the parsing algorithm has terminated. It is from this final parsing array, along with the stored alphanumeric transcriptions of the input text, that the remaining displays are derived.

Appendix B shows a sample of a line printer display which depicts a tree structure derived from the input. For each non-terminal node in the tree, the parsing array row number and the grammatical label are printed at the left; at the right, the row numbers, labels, and corresponding substrings of the daughter nodes are given. To illustrate, the following tree, labeled "A", would be represented as in diagram "B" below:



(A)

E	F	T_1
		T_2
	G	T_3
		T_4
F	H	T_1
	I	T_2
G	J	T_3
	K	T_4

(B)

It is also possible to obtain a line printer display of well-formed substrings derived from the input. Normally, the subroutine which prepares this display prints out in succession all well-formed substrings of progressively increasing length, beginning with two-word phrases and exhausting all entires in the parsing array. By appropriate typewriter-input commands, however, selective displays can be generated on the basis of phrase length or phrase label or both.

2.1.2 The Vector-Symbol Phrase Grammar

The grammatical labels are given in Appendix A. They represent both the basic grammar codes (the results of the pseudo-dictionary lookup), as well as the higher-level labels assigned as the result of applying the rules to the basic grammar codes (e.g., participial phrase, sentence, etc.). The same labels in many cases may refer either to basic grammar codes or to higher level constructs (e.g., a noun by itself may constitute a noun phrase).

The variables, A and B, as shown on the first page of Appendix A, are used to indicate number/gender/person or case. The values for each of the digits which may be used in the variable positions are given (recalling that 9 and 0 are special symbols used in the rules, but never assigned as grammatical labels). The subscripts "a" and "g" indicate whether the variable illustrates a government or agreement relationship.

The method of dealing with so-called "homographs" within this system cannot be stressed enough. Homographs are marked by more than one grammar code, the maximum number of codes being twenty. From the codes provided, it can be seen that some morphological types which very regularly take on a number of different syntactic functions are no less homographs for the purposes of this grammar than are many "accidental" homographs. Thus, for example, "сел" (third singular past of "sit" and genitive plural of "village") is not more a homograph (indeed less so) than "новой" (feminine genitive, dative, instrumental and prepositional of "new"). These alternatives can either be incorporated into viable structures accounting for the entire sentence or they cannot; if more than one of the grammar codes can be incorporated (or the same code incorporated in more than one significantly different way), the sentence is grammatically ambiguous; if only one of the grammar codes of a given word may be employed in a construct leading to a sentence derivation, its homography may be said to have been resolved. Homographs are thus not to be looked upon as "special cases", but simply as the normal material upon which the parsing algorithm operates.

Of course, it is always possible to reduce the number of alternate grammar codes at the cost of increasing the number of rules. Thus, for example, each unique combination of grammar codes assignable to some class of Russian words (including classes of only one member) could be given a unique grammar code. This would, however, greatly reduce the generality of the grammar and make grammar writing a prodigious task. No attempt has therefore been made to "telescope" the entries having multiple grammar codes except in a few cases.

The grammar codes given herein are not complete nor even necessarily definitive insofar as they go. They do not, for example, account for some cases of dual government (e.g., a short form passive participle taking two instrumental objects) nor are nouns taking nominal complements other than genitive considered. The grammar codes thus far created are adequate for the coding of the vast majority of the Russian sentences we encounter. One of the major advantages of such a system is the ability to create new grammar codes pro re nata and immediately incorporate them in rules in the grammar.

The Rules

Appendix D gives the 318 rules currently in use. They are, generally, quite readily interpretable though some patience may be required.

A few matters, however, deserve further attention. In general, a sentence is recognized as: sentence begin symbol, legal sentence tree, sentence terminating symbol. Of course, since the algorithm is binary, the recognition of these components must be done in two steps. The reason for incorporating a symbol to indicate the beginning of a sentence is that some Russian structures occur only at the beginning of sentences or after commas (e.g., gerund clauses). If the beginning of the sentence were not indicated, the rules would have to be formulated in such a way as to accept such structures whether or not they were preceded by a symbol indicating their left boundary (since they might occur sentence initial) which could lead to undesirable results.

In connection with this, another problem arises—that of the dual functioning of Russian punctuation. There are many Russian constructions which are obligatorily marked as to their beginning and ending (e.g., gerund and relative clauses). Unfortunately, one mark of punctuation may serve to both end one construction and initiate another or to end two such constructions simultaneously. Consider, for example, a relative clause occurring at the end of a sentence and obligatorily surrounded by punctuation indicating its beginning and end. In this case, however, the period that marks the end of the relative clause also marks the end of the sentence.

Several alternatives come to mind for the solution of such difficulties; three will be discussed below.

First, the rules may be so formulated as to accept such structures whether they are preceded and/or followed by overt boundary markers or not. Thus, boundary markers serving to delimit two or more structures simultaneously could unambiguously be assigned to one or the other in the course of a sentence derivation (caution must, of course, be exercised in framing the rules to insure that multiple syntactic interpretations hinging solely on which structures the multi-functioning boundary markers are assigned to, are not derived).

An immediate objection to such expedient is that there is no way of knowing in advance whether, under certain specified circumstances, ignoring a boundary marker (which is the effect of framing the rules in this manner) will result in multiple analyses, some of which could have been avoided by taking the boundary indicator in question into consideration. In an early version of the grammar, this, in fact, occurred when an incorrect derivation was assigned to one of the test sentences as a result of ignoring a comma ending a relative clause.

In general, it is undesirable for a parsing grammar to rely on the non-occurrence of certain (presumably) ill-formed strings. Aside from weakening the grammar's ability to recognize ill-formed input, presumptions about ill-formedness are often ill-conceived, since rarely are all the ramifications of such a decision apparent from the outset.

Alternative 2 requires the inputting of a "dummy" word following each potential boundary marker. This dummy element receives the same grammatical classification as the potential marker (and thus, of course, is a potential boundary marker itself). Rules are included for combining dummy elements and boundary markers to produce a single boundary indicator to handle those cases in which such elements (e.g., comma, period) do not serve multiple functions.

This option was, in fact, exercised for a while with generally gratifying results. It is easy enough to see, however, that when the number of structures obligatorily initiated or concluded by a single mark of punctuation reaches three, the technique is inadequate and would have to be augmented by the method to be described subsequently (i.e., that of "carrying up" grammatical information to higher level structures). While few sentences having punctuation serving the functions described have been encountered, they must nevertheless be taken into account. The technique was therefore abandoned for the one described below.

The third alternative is to formulate the rules in such a way that information about the termination of lower level structures is carried upward to be used in determining whether the same boundary marker may also indicate the termination of a higher level structure.

Consider the parsing for the sentence illustrated on pages C-22 and C-23 in Appendix C. The sentences ends with a participial phrase which requires a mark of punctuation to terminate it. In this case, the period terminates both the participial phrase and the sentence. As can be seen from the tree structure, the sentence end information was carried up to the noun phrase object, the verb phrase, and ultimately the sentence. This solution, though altogether adequate and now functioning reasonably reliably, is not so elegant as one might wish, since it requires the creation of nodes for both sentence terminating and non-sentence terminating noun phrases, verb phrases, etc. The problem can be solved in a general and somewhat simpler way by increasing the number of variables or by allowing context sensitive rules. In is anticipated that in some of our future experiments either or both of these possibilities will be explored.

3. CONCLUSIONS

Our primary concern has been the creation of a formalized sentence analysis system capable of describing Russian sentences adequately for machine translation, information retrieval, and related data processing operations. Our initial experiments were based upon a binary, context-free, phrase-structure grammar, chosen because of the simplicity of the companion algorithm and of the form of the grammar itself. We did not, of course, believe that such a system would be adequate to our main task; rather, we felt it would be useful to explore its weaknesses in some detail as a preliminary to the design of a more suitable system. These weaknesses are much discussed in the literature from a general linguistic point of view. Our exploratory study was undertaken to develop empirical results that would help to sharpen the relevance of such general discussions for our own rather limited data processing applications.

The two most serious deficiencies we encountered were, predictably, the multiplicity of grammar rules required to account for agreement and government situations, and the proliferation of trivially different structural descriptions assigned to even simple input strings. Both of these problems might be dealt with up to a point—at least for our purposes—by allowing the grammar to grow to the very limits of practicality. But a solution of this kind is not only uninteresting, it is quite hopeless as well, in view of the burden it places on the research grammarian and the computer's internal storage capacity.

To overcome these difficulties, the vector-symbol phrase grammar, with node suppression, was designed and incorporated in the parsing system. The "abbreviative" conventions for collapsing large sets of PS rules into small sets of VS rules have made it possible to undertake seriously the writing of a phrase grammar for written Russian suitable for our data processing applications. In our opinion, the practical potential of this type of grammar—whose appeal lies in its basic simplicity and in the straightforwardness of its implementation—has not been sufficiently explored or exploited. The rapidity with which it has been possible to develop the grammar described in this report is felt to be one justification of this position. The grammar is far from "complete", except as measured against most other operating formal grammars for Russian text; it does, however, deal in a general fashion with a broad range of Russian sentence types, and no major intrinsic impediments to its continued evolution have been encountered.

Appendix B gives computer output of parsings for two sample sentences. The English has been typed on the examples to assist those unfamiliar with Russian in following the parsing. The parsing list is provided as well as the equivalent of a tree diagram for each sentence.

Appendix C gives sample sentences and their parsings, illustrating a variety of Russian sentence types. For convenience, the trees have been redrawn from the computer output. Interpretations for the labels on the nodes may be found in Appendix A. Interested readers can refer to the rules in Appendix D to see the way in which they were applied to the sample sentences.

APPENDIX A

GRAMMATICAL LABELS[1]

Significance of Variables

A. Number/gender or person

1. Masculine
2. Feminine
3. Neuter
4. Plural
5. First person singular
6. Second person singular
7. First person plural
8. Second person plural

B. Case

1. Nominative
2. Genitive
3. Dative
4. Accusative
5. Instrumental
6. Prepositional
7. Accusative animate

Subscripts

a. Agreement
g. Government

[1] We are indebted to Warren Joseph Plath's Harvard doctoral thesis for several of our syntactic classes. (Mathematic Linguistics and Automatic Translation, Report No. NSF-12 to the National Science Foundation, Anthony G. Oettinger, Principal Investigator, Cambridge, Massachusetts, June, 1963.)

PREDICATIVES

1)	$002A_aB_g$	Finite, transitive, personal
2)	$002A_a8$	Verb phrase with object
3)	$012A_a8$	Finite, intransitive, personal
4)	$022A_aB_g$	Impersonal
5)	$032A_aB_g$	Short form adjective
6)	$042A_aB_g$	Future form of BYT* (to be)
7)	$052A_aB_g$	Past form of BYT* (to be)
8)	$062A_aB_g$	Short form comparative adjective
9)	$072A_aB_g$	Short form participle
10)	$082A_aB_g$	Short form, infinitive government
11)	$092A_aB_g$	Verb phrase with a period
12)	$102A_aB_g$	Past participle with nominal clause subject
13)	$112A_aB_g$	Impersonal verb with infinitive subject
14)	$122A_aB_g$	Personal verb with nominal clause direct object
15)	$132A_aB_g$	Personal verb with infinitive direct object
16)	$142A_aB_g$	Personal verb with dative object and object in another case (as specified by the variable)
17)	$152A_aB_g$	Model auxiliary

NOMINALS

Values for variables

4th octal digit Number/gender or person

5th octal digit Case

1)	$001A_aB_a$	Noun taking adjectives and a genitive nominal complement
2)	$021A_aB_a$	Surname
3)	$031A_aB_a$	Title (e.g., GENERAL)
4)	$051A_aB_a$	Coordinated noun phrase
5)	$061A_aB_a$	"SHTO" (as nominal)
6)	$0068B_a$	Coordinated conjunction plus nominal
7)	00676	Name of month in genitive singular

7)	$071A_aB_a$	Unmodifiable nominal (e.g., pronouns)
8)	$091A_aB_a$	Noun phrase with a period

INFINITIVES

1)	$0048B_g$	Infinitive taking one object
2)	00488	Infinitive phrase with object(s) (or intransitive)
3)	$004B_gB_g$	Infinitive taking two objects

GERUND

1)	00588	Gerund phrase with object(s) (or intransitive)
2)	$0058B_g$	Gerund taking one object
3)	$004B_gB_g$	Gerund taking two objects
4)	00578	Gerund phrase plus comma

ADVERBS AND SPECIAL WORDS, PHRASES

1)	00711	Adverb of manner
2)	00712	Adverb of time
3)	$0063A_aB_a$	Relative pronoun
4)	00777	Subjunctive particle "BY"
5)	01711	Single capitalized Cyrillic character
6)	01811	Capitalized character plus period
7)	00733	Negative "NET"

PREPOSITIONAL PHRASES

1)	$0088B_g$	Preposition
2)	$0188B_a$	Prepositional phrase
3)	01888	+ prepositional phrase
4)	$0288B_a$	Prepositional phrase + period
5)	$0788B_a$	Prepositional phrase + dash

RELATIVE CLAUSES

1)	$063A_aB_a$	Relative pronoun
2)	$064A_a8$	Relative pronoun and verb phrase (relative clauses)
3)	$074A_a8$	Relative clause plus period
4)	$065A_a8$	Relative clause plus comma
5)	$063A_a8$	Comma plus relative clause plus comma
6)	$084A_a8$	Comma plus relative clause plus period

PUNCTUATION

1)	00611	Comma
2)	00621	Coordinate conjunction
3)	00631	Period
4)	00641	Exclamation point
5)	00651	Sentence begin symbol
6)	00661	
7)	$0068B_a$	Coordinating conjunction (or comma) plus noun phrase
8)	00753	"SHTO" (as conjunction)
9)	00715	Dash
10)	$0077B_a$	Dash plus noun phrase
11)	00716	Dash plus nominative noun phrase
12)	00722	Left quote
13)	00723	Right quote
14)	$041A_aB_a$	Left quote plus nominal

ERIC

CLAUSES AND SENTENCES

1)	12345	Sentence begin symbol + sentence + period
2)	9988	Regular sentence (subject Predicate) + period
3)	9987	Inverted sentence (predicate subject) + period
4)	9888	Regular sentence (see 2)
5)	9887	Inverted sentence (see 3)
6)	$985B_g$	Transitive verb (without object) + subject
7)	$986B_g$	Subject + transitive verb (without object)
8)	00854	"SHTO" clause plus period
9)	00855	Comma plus "SHTO" clause plus period
10)	00753	"SHTO" (introduces nominal clauses)

ADJECTIVES AND PARTICIPLES

Values for variables; as for nominals

1)	$003A_aB_a$	Adjective
2)	$023A_aB_a$	Participle, genitive government
3)	$033A_aB_a$	Participle, dative government
4)	$043A_aB_a$	Participle, accusative government
5)	$053A_aB_a$	Participle, instrumental government
6)	$073A_aB_a$	Participle, accusative animate
7)	$083A_aB_a$	Participle, intransitive or transitive participle plus object
8)	$037A_aB_a$	Unmodifiable adjective (e.g., NA)IJ)
9)	$038A_aB_a$	Comma + participial phrase
10)	$093A_aB_a$	Comma + participial phrase + period
11)	$0965B_a$	Comma + participial phrase + comma
12)	00672	Numeral requiring genitive plural nominal
13)	00675	Numeral less than 31 (combines with months to form adverbials of time)

APPENDIX B

```
 1)
          651
 2)     KROME                                                   Besides
          882                                                             (Furthermore)
 3)     TOGO                                                    that
        3712  3732  7112
 4)     NAS                                                     us
        7147  7146
 5)     VOZMUWALA                                               revolted
          224   227
 6)     OSUWESTVL=EMA=                                          practices
        5321
 7)     PRAVITEL*STVOM                                          (by) government
          135
 8)     S)A                                                     USA
          141   142   143   144   145   146
 9)     SISTEMA                                                 System
          121
10)     PEREXVATA                                               (of) interception
          112
11)     I                                                       and
          621
12)     DE)IFROVANI=                                            decoding
          132   141   144
13)     SEKRETNYX                                               (of) secret
          342   346   347
14)     SOOBWENIJ                                               messages
          142
15)     SVOIX                                                   (of) its
        3742  3746  3747
16)     SOQZNIKOV                                               allies.
          142   147
17)     .
          631

        END OF SENTENCE

 PAIRS =  506     RULES TESTED =  443
 LEFT MATCHES =  275     FULL MATCHES =  155     NON-TERMINAL NODES =   66
 HIGHEST STRUCTURE = 99.  /   1)  /  12345
```

FINAL PARSING LIST

1.	1	651	1	0
2.	2	882	2	0
3.	3	3712	3	0
4.	3	3732	4	0
5.	3	7112	5	0
6.	4	7147	6	0
7.	4	7146	7	0
8.	5	224	8	0
9.	5	227	9	0
10.	6	5321	10	0
11.	7	135	11	0
12.	8	141	12	0
13.	8	142	13	0
14.	8	143	14	0
15.	8	144	15	0
16.	8	145	16	0
17.	8	146	17	0
18.	9	121	18	0
19.	10	112	19	0
20.	11	621	20	0
21.	12	132	21	0
22.	12	141	22	0
23.	12	144	23	0
24.	13	342	24	0
25.	13	346	25	0
26.	13	347	26	0
27.	14	142	27	0
28.	15	3742	28	0
29.	15	3746	29	0
30.	15	3747	30	0
31.	16	142	31	0
32.	16	147	32	0
33.	17	631	33	0
34.	2	1882	2	5
35.	4	228	6	9
36.	6	8321	10	11
37.	7	135	11	13
38.	9	121	18	19
39.	11	682	20	21
40.	11	681	20	22
41.	11	684	20	23
42.	13	142	24	27
43.	15	7142	28	31
44.	15	7147	30	32
45.	1	1888	1	34
46.	6	8321	10	37
47.	10	5142	19	39
48.	12	132	21	42
49.	12	141	22	42
50.	12	144	23	42
51.	14	142	27	43
52.	9	121	18	47
53.	11	682	20	48
54.	11	681	20	49
55.	11	684	20	50

56.	13	142	24	51
57.	9	5141	38	40
58.	6	121	46	18
59.	5	9854	8	58
60.	5	9857	9	58
61.	10	5142	19	53
62.	12	132	21	56
63.	12	141	22	56
64.	12	144	23	56
65.	6	121	46	38
66.	5	9854	8	65
67.	5	9857	9	65
68.	9	121	18	61
69.	11	682	20	62
70.	11	681	20	63
71.	11	684	20	64
72.	4	9858	35	58
73.	4	9887	35	58
74.	9	5141	38	54
75.	10	5142	19	69
76.	4	9858	35	65
77.	4	9887	35	65
78.	6	121	46	52
79.	6	5141	65	40
80.	5	9854	8	78
81.	5	9857	9	78
82.	9	121	18	75
83.	9	5141	38	70
84.	4	9858	35	78
85.	4	9887	35	78
86.	6	121	46	68
87.	6	5141	65	54
88.	5	9854	8	86
89.	5	9857	9	86
90.	4	9858	35	86
91.	4	9887	35	86
92.	6	121	46	82
93.	6	5141	65	70
94.	5	9854	8	92
95.	5	9857	9	92
96.	4	9858	35	92
97.	4	9887	35	92
98.	4	9987	97	33
99.	1	12345	45	98

CONSTRUCT / LABEL		COMPONENTS / LABEL / PHRASE			
99.	12345	45.	1888		
				KROME TOGO	Besides that (Furthermore)
		98.	9987	NAS VOZMUWALA OSUWESTVL=EMA= PRAVITEL*STVOM S)A SISTEMA PEREXVATA I DE)IFROVANI= SEKRETNYX SOOBWENIJ SVOIX SOQZNIKOV .	us revolted practiced (by) government USA System (of) interception and decoding (of) secret messages (of) its allies.
45.	1888	1.	651		
		34.	1882	KROME TOGO	Besides that (Furthermore)
98.	9987	97.	9887	NAS VOZMUWALA OSUWESTVL=EMA= PRAVITEL*STVOM S)A SISTEMA PEREXVATA I DE)IFROVANI= SEKRETNYX SOOBWENIJ SVOIX SOQZNIKOV	us revolted practiced (by) government USA System (of) interception and decoding (of) secret messages (of) its allies.
		33.	631	.	

34.	1882	2.	882	KROME	Besides (Furthermore
		5.	7112	TOGO	that
97.	9887	35.	228	NAS VOZMUWALA	us revolted
		92.	121	OSUWESTVL=EMA= PRAVITEL*STVOM S)A SISTEMA PEREXVATA I DE)IFROVANI= SEKRETNYX SOOBWENIJ SVOIX SOQZNIKOV	practiced (by) government USA System (of) interception and decoding (of) secret messages (of) its allies
35.	228	6.	7147	NAS	us
		9.	227	VOZMUWALA	revolted
92.	121	46.	8321	OSUWESTVL=EMA= PRAVITEL*STVOM S)A	practiced (by) government USA
		82.	121	SISTEMA PEREXVATA I DE)IFROVANI= SEKRETNYX SOOBWENIJ SVOIX SOQZNIKOV	System (of) interception and decoding (of) secret messages (of) its allies
46.	8321	10.	5321	OSUWESTVL=EMA=	practiced
		37.	135	PRAVITEL*STVOM S)A	(by) government USA
82.	121	18.	121	SISTEMA	System

		75.	5142	PEREXVATA I DE)IFROVANI= SEKRETNYX SOOBWENIJ SVOIX SOQZNIKOV	(of) interception and decoding (of) secret messages (of) its all..es
37.	135	11.	135	PRAVITEL*STVOM	(by) government
		13.	142	S)A	USA
75.	5142	19.	112	PEREXVATA	(of) interception
		69.	682	I DE)IFROVANI= SEKRETNYX SOOBWENIJ SVOIX SOQZNIKOV	and decoding (of) secret messages (of) its allies
69.	682	20.	621	I	and
		62.	132	DE)IFROVANI= SEKRETNYX SOOBWENIJ SVOIX SOQZNIKOV	decoding (of) secret messages (of) its allies
62.	132	21.	132	DE)IFROVANI=	decoding
		56.	142	SEKRETNYX SOOBWENIJ SVOIX SOQZNIKOV	(of) secret messages (of) its allies
56.	142	24.	342	SEKRETNYX	(of) secret
		51.	142	SOOBWENIJ SVOIX SOQZNIKOV	messages (of) its allies

51.	142	27.	142	SOOBWENIJ	messages
		43.	7142	SVOIX SOQZNIKOV	(of) its allies
43.	7142	28.	3742	SVOIX	(of) its
		31.	142	SOQZNIKOV	allies

END OF STRUCTURE

1)	651	
2)	SEGODN=	Today
	712	
3)	NA)A	our
	3721	
4)	PARTI=	party,
	121	
5)	,	
	611	
6)	VES*	all
	3711 3714 7111 7114	
7)	SOVETSKIJ	Soviet
	311 314	
8)	NAROD	people,
	111 114	
9)	,	
	611	
10)	PROGRESIVNA=	progressive
	321	
11)	OBWESTVENNOST*	society
	121 124	
12)	MIRA	(of) world
	112	
13)	OTME(AQT	mark
	244	
14)	50'LETIE	50th anniversary
	134 131	
15)	GAZETY	(of) newspaper
	141 144 122	
16)	((	
	722	
17)	PRAVDA	"Pravda"—
	121	
18)	))	
	723	
19)	-	
	715	
20)	BOEVOGO	fighting
	312 332	
21)	ORGANA	organ
	112	
22)	CENTRAL*NOGO	(of) Central
	312 332	
23)	KOMITETA	Committee
	112	
24)	KOMMUNISTI(ESKOJ	(of) Communist
	322 323 325	
25)	PARTII	Party
	122 123 126 141 144	
26)	SOVETSKOGO	(of) Soviet
	312 332	
27)	SOQZA	Union.
	112	
28)		
	631	

END OF SENTENCE

```
PAIRS = 939      RULES TESTED = 965
LEFT MATCHES = 561      FULL MATCHES = 404      NON-TERMINAL NODES = 133
HIGHEST STRUCTURE =179.  /  1) · /  12345
```

FINAL PARSING LIST

1.	1	651	1	0
2.	2	712	2	0
3.	3	3721	3	0
4.	4	121	4	0
5.	5	611	5	0
6.	6	3711	6	0
7.	6	3714	7	0
8.	6	7111	8	0
9.	6	7114	9	0
10.	7	311	10	0
11.	7	314	11	0
12.	8	111	12	0
13.	8	114	13	0
14.	9	611	14	0
15.	10	321	15	0
16.	11	121	16	0
17.	11	124	17	0
18.	12	112	18	0
19.	13	244	19	0
20.	14	134	20	0
21.	14	131	21	0
22.	15	141	22	0
23.	15	144	23	0
24.	15	122	24	0
25.	16	722	25	0
26.	17	121	26	0
27.	18	723	27	0
28.	19	715	28	0
29.	20	312	29	0
30.	20	332	30	0
31.	21	112	31	0
32.	22	312	32	0
33.	22	332	33	0
34.	23	112	34	0
35.	24	322	35	0
36.	24	323	36	0
37.	24	325	37	0
38.	25	122	38	0
39.	25	123	39	0
40.	25	126	40	0
41.	25	141	41	0
42.	25	144	42	0
43.	26	312	43	0
44.	26	332	44	0
45.	27	112	45	0
46.	28	631	46	0
47.	1	651	1	2
48.	3	7121	3	4
49.	5	681	5	8
50.	5	684	5	9
51.	7	111	10	12
52.	7	114	11	13
53.	9	10321	14	15
54.	10	121	15	16
55.	11	121	16	18

56.	11	124	17	18
57.	13	248	19	20
58.	14	134	20	24
59.	14	131	21	24
60.	16	1722	25	26
61.	16	4121	25	26
62.	20	112	29	31
63.	22	112	32	34
64.	24	122	35	38
65.	24	123	36	39
66.	26	112	43	45
67.	4	5141	4	49
68.	6	7111	6	51
69.	6	7114	7	52
70.	9	681	14	54
71.	10	121	15	55
72.	13	248	19	58
73.	19	772	28	62
74.	21	112	31	63
75.	23	112	34	64
76.	25	122	38	66
77.	25	123	39	66
78.	25	126	40	66
79.	25	141	41	66
80.	25	144	42	66
81.	11	248	56	19
82.	13	9858	57	22
83.	13	9887	57	22
84.	16	1723	60	27
85.	16	121	61	27
86.	5	681	5	68
87.	5	684	5	69
88.	8	5141	12	70
89.	9	681	14	71
90.	15	141	22	84
91.	15	144	23	84
92.	15	122	24	84
93.	20	112	29	74
94.	22	112	32	75
95.	24	122	35	76
96.	24	123	36	77
97.	3	5141	48	49
98.	4	5141	4	86
99.	8	5141	12	89
100.	14	134	20	92
101.	14	131	21	92
102.	19	772	28	93
103.	21	112	31	94
104.	23	112	34	95
105.	7	5141	51	70
106.	13	248	19	100
107.	20	112	29	103
108.	22	112	32	104
109.	3	5141	48	86
110.	7	5141	51	89
111.	13	9858	57	90
112.	13	9887	57	90
113.	6	5141	68	70
114.	19	772	28	107

115.	21	112	31	108
116.	6	5141	68	89
117.	15	122	92	73
118.	8	9888	99	57
119.	14	134	20	117
120.	14	131	21	117
121.	20	112	29	115
122.	4	5141	98	70
123.	8	9888	99	72
124.	7	9888	110	57
125.	13	248	19	119
126.	19	772	28	121
127.	15	122	92	102
128.	4	5141	98	89
129.	3	5141	109	70
130.	7	9888	110	72
131.	6	9888	116	57
132.	14	134	20	127
133.	14	131	21	127
134.	3	5141	109	89
135.	6	9888	116	72
136.	13	248	19	132
137.	15	122	92	114
138.	8	9888	99	106
139.	4	9888	128	57
140.	14	134	20	137
141.	14	131	21	137
142.	7	9888	110	106
143.	4	9888	128	72
144.	3	9888	134	57
145.	13	248	19	140
146.	15	122	92	126
147.	6	9888	116	106
148.	3	9888	134	72
149.	14	134	20	146
150.	14	131	21	146
151.	8	9888	99	125
152.	13	248	19	149
153.	7	9888	110	125
154.	4	9888	128	106
155.	8	9888	99	136
156.	6	9888	116	125
157.	3	9888	134	106
158.	7	9888	110	136
159.	8	9888	99	145
160.	6	9888	116	136
161.	4	9888	128	125
162.	7	9888	110	145
163.	3	9888	134	125
164.	8	9888	99	152
165.	6	9888	116	145
166.	4	9888	128	136
167.	7	9888	110	152
168.	3	9888	134	136
169.	8	9988	164	46
170.	6	9888	116	152
171.	4	9888	128	145
172.	7	9988	167	46
173.	3	9888	134	145

174.	6	9988	170	46
175.	4	9888	128	152
176.	3	9888	134	152
177.	4	9988	175	46
178.	3	9988	176	46
179.	1	12345	47	178

CONSTRUCT / LABEL		COMPONENTS / LABEL / PHRASE			
179.	12345	47.	651		
				SEGODN=	Today
		178.	9988	NA)A	our
				PARTI=	party,
				,	
				VES*	all
				SOVETSKIJ	Soviet
				NAROD	people,
				,	
				PROGRESIVNA=	progressive
				OBWESTVENNOST*	society
				MIRA	(of) world
				OTME(AQT	mark
				50'LETIE	50th anniversary
				GAZETY	(of) newspaper
				((	
				PRAVDA	"Pravda" —
				))	
				-	
				BOEVOGO	fighting
				ORGANA	organ
				CENTRAL*NOGO	(of) Central
				KOMITETA	Committee
				KOMMUNISTI(ESKOJ	(of) Communist
				PARTII	Party
				SOVETSKOGO	(of) Soviet
				SOQZA	Union.
				.	
47.	651	1.	651		
		2.	712	SEGODN=	Today
178.	9988	176.	9888	NA)A	our
				PARTI=	party,
				,	
				VES*	all
				SOVETSKIJ	Soviet
				NAROD	people,
				,	
				PROGRESIVNA=	progressive
				OBWESTVENNOST*	society

				MIRA	(of) world
				OTME(AQT	mark
				50'LETIE	50th anniversary
				GAZETY	(of) newspaper
				((	
				PRAVDA	"Pravda"—
				))	
				-	
				BOEVOGO	fighting
				ORGANA	organ
				CENTRAL*NOGO	(of) Central
				KOMITETA	Committee
				KOMMUNISTI(ESKOJ	(of) Communist
				PARTII	Party
				SOVETSKOGO	(of) Soviet
				SOQZA	Union.
		46.	631	.	
176.	9888	134.	5141	NA)A	our
				PARTI=	party,
				,	
				VES*	all
				SOVETSKIJ	Soviet
				NAROD	people,
				,	
				PROGRESIVNA=	progressive
				OBWESTVENNOST*	society
				MIRA	(of) world
		152.	246	OTME(AQT	mark
				50'LETIE	50th anniversary
				GAZETY	(of) newspaper
				((	
				PRAVDA	"Pravda"—
				))	
				-	
				BOEVOGO	fighting
				ORGANA	organ
				CENTRAL*NOGO	(of) Central
				KOMITETA	Committee
				KOMMUNISTI(ESKOJ	(of) Communist
				PARTII	Party
				SOVETSKOGO	(of) Soviet
				SOQZA	Union
134.	5141	109.	5141	NA)A	our
				PARTI=	party,
				,	
				VES*	all
				SOVETSKIJ	Soviet
				NAROD	people,

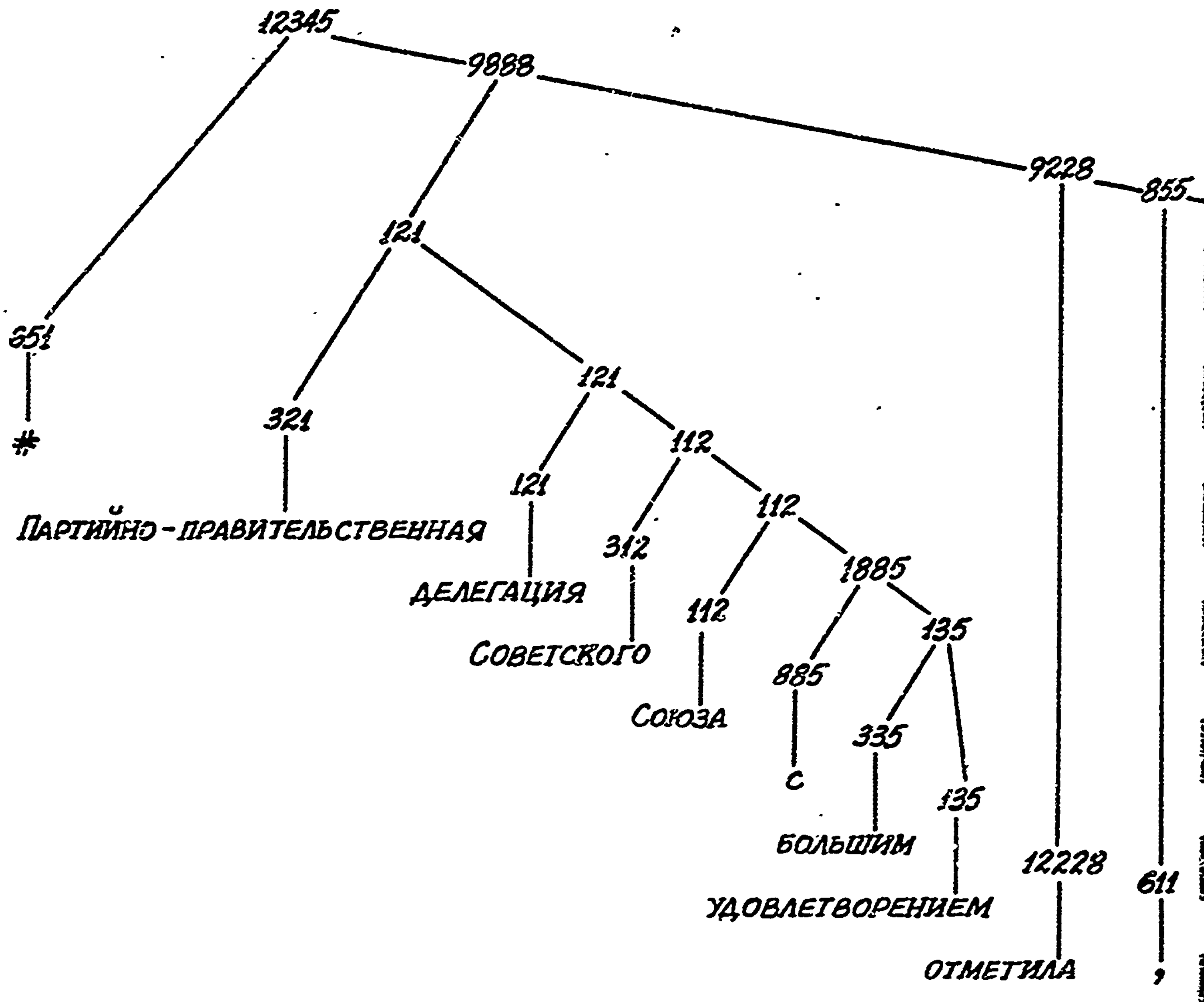
		89.	681	, PROGRESIVNA= OBWESTVENNOST* MIRA	progressive society (of) world
152.	248	19.	244	OTME(AQT	mark
		149.	134	50'LETIE GAZETY ((PRAVDA)) - BOEVOGO ORGANA CENTRAL*NOGO KOMITETA KOMMUNISTI(ESKOJ PARTII SOVETSKOGO SOQZA	50th anniversary (of) newspaper "Pravda"— fighting organ (of) Central Committee (of) Communist Party (of) Soviet Union
109.	5141	48.	7121	NA)A PARTI=	our party,
		86.	681	, VES* SOVETSKIJ NAROD	all Soviet people,
89.	681	14.	611		
		71.	121	PROGRESIVNA= OBWESTVENNOST* MIRA	progressive society (of) world
149.	134	20.	134	50'LETIE	50th anniversary
		146.	122	GAZETY ((PRAVDA)) - BOEVOGO ORGANA CENTRAL*NOGO	(of) newspaper "Pravda"— fighting organ (of) Central

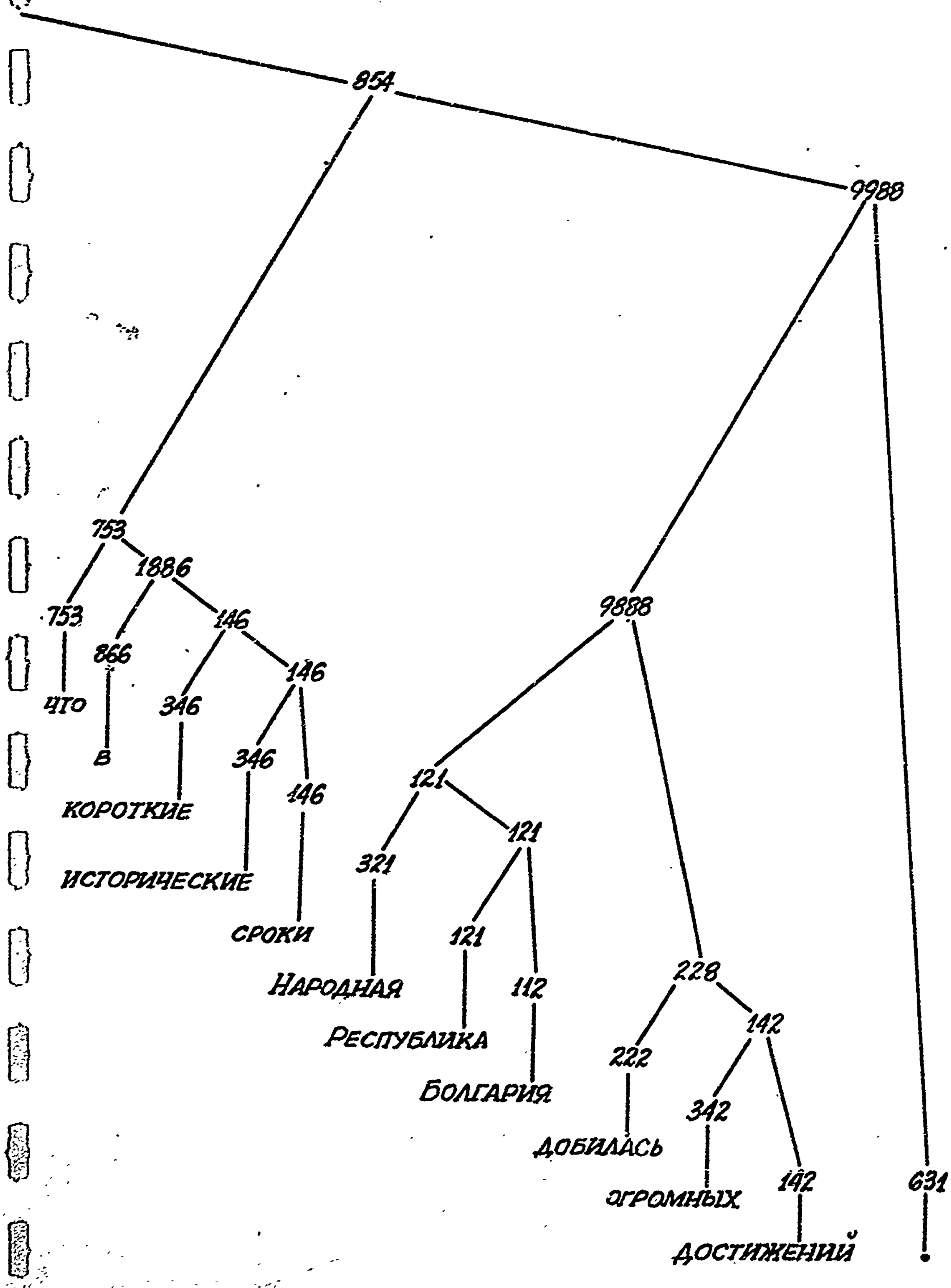
				KOMITETA KOMMUNISTI(ESKOJ PARTII SOVETSKOGO SOQZA	Committee (of) Communist Party (of) Soviet Union
48.	7121	3.	3721	NA?A	our
		4.	121	PARTI=	party,
86.	681	5.	611	,	
		68.	7111	VES* SOVETSKIJ NAROD	all Soviet people
71.	121	15.	321	PROGRESIVNA=	progressive
		55.	121	OBWESTVENNOST* MIRA	society (of) world
146.	122	92.	122	GAZETY ((PRAVDA))	(of) newspaper "Pravda"—
		126.	772	- BOEVOGO ORGANA CENTRAL*NOGO KOMITETA KOMMUNISTI(ESKOJ PARTII SOVETSKOGO SOQZA	 fighting organ (of) Central Committee (of) Communist Party (of) Soviet Union
68.	7111	6.	3711	VES*	all
		51.	111	SOVETSKIJ NAROD	Soviet people
55.	121	16.	121	OBWESTVENNOST*	society

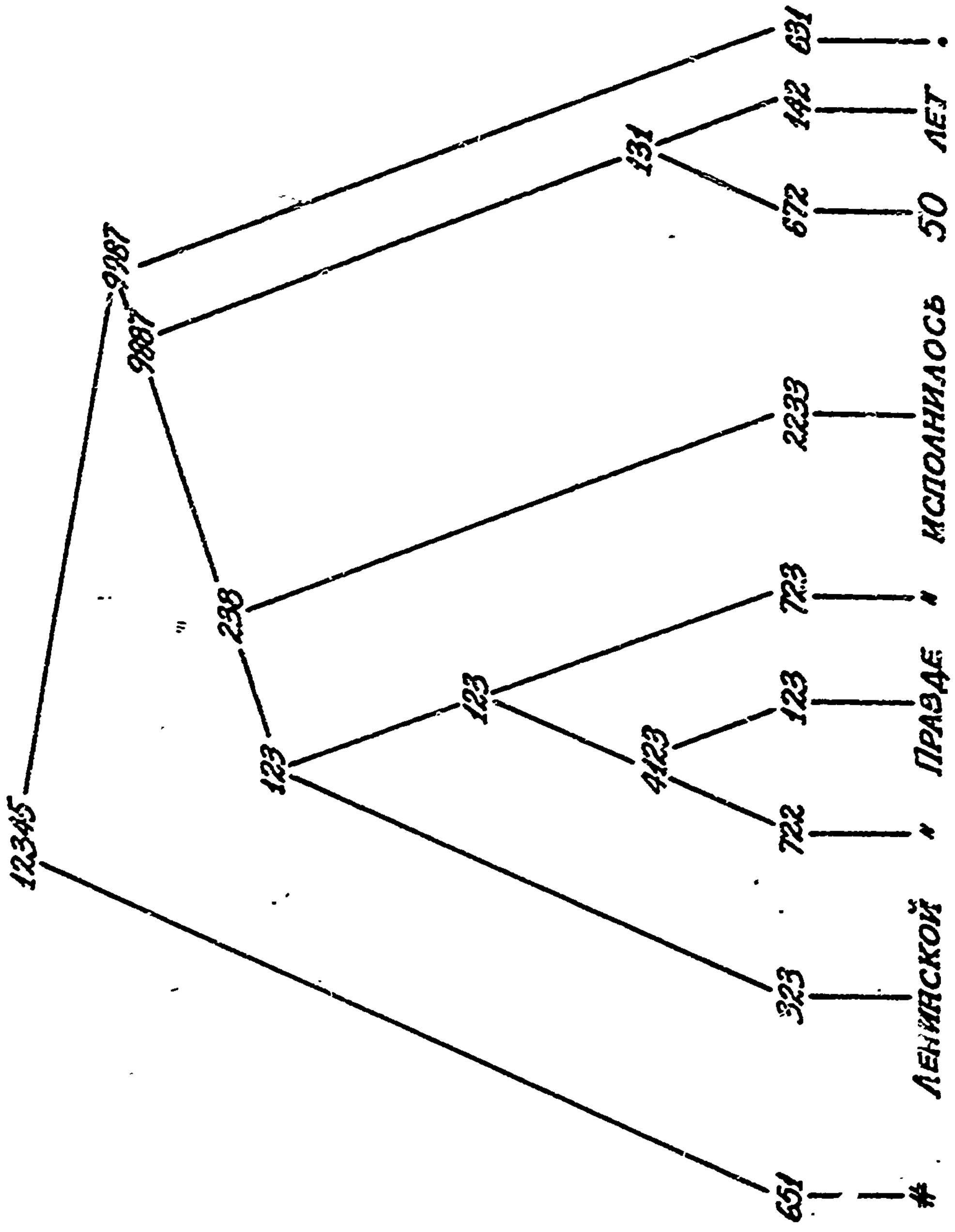
		18.	112	MIRA	(of) world
92.	122	24.	122	GAZETY	(of) newspaper
		84.	1723	((PRAVDA))	"Pravda"—
126.	772	28.	715	-	
		121.	112	BOEVOGO ORGANA CENTRAL*NOGO KOMITETA KOMMUNISTI(ESKOJ PARTII SOVETSKOGO SOQZA	fighting organ (of) Central Committee (of) Communist Party (of) Soviet Union
51.	111	10.	311	SOVETSKIJ	Soviet
		12.	111	NAROD	people
84.	1723	60.	1722	((PRAVDA	"Pravda"
		27.	723	))	
121.	112	29.	312	BOEVOGO	fighting
		115.	112	ORGANA CENTRAL*NOGO KOMITETA KOMMUNISTI(ESKOJ PARTII SOVETSKOGO SOQZA	organ (of) Central Committee (of) Communist Party (of) Soviet Union
60.	1722	25.	722	((	

		26.	121	PRAVDA	"Pravda"
115.	112	31.	112	ORGANA	organ
		108.	112	CENTRAL*NOGO KOMITETA KOMMUNISTI(ESKOJ PARTII SOVETSKOGO SOQZA	(of) Central Committee (of) Communist Party (of) Soviet Union
108.	112	32.	312	CENTRAL*NOGO	(of) Central
		104.	112	KOMITETA KOMMUNISTI(ESKOJ PARTII SOVETSKOGO SOQZA	Committee (of) Communist Party (of) Soviet Union
104.	112	34.	112	KOMITETA	Committee
		95.	122	KOMMUNISTI(ESKOJ PARTII SOVETSKOGO SOQZA	(of) Communist Party (of) Soviet Union
95.	122	35.	322	KOMMUNISTI(ESKOJ	(of) Communist
		76.	122	PARTII SOVETSKOGO SOQZA	Party (of) Soviet Union
76.	122	38.	122	PARTII	Party
		66.	112	SOVETSKOGO SOQZA	(of) Soviet Union
66.	112	43.	312	SOVETSKOGO	(of) Soviet
		45.	112	SOQZA	Union

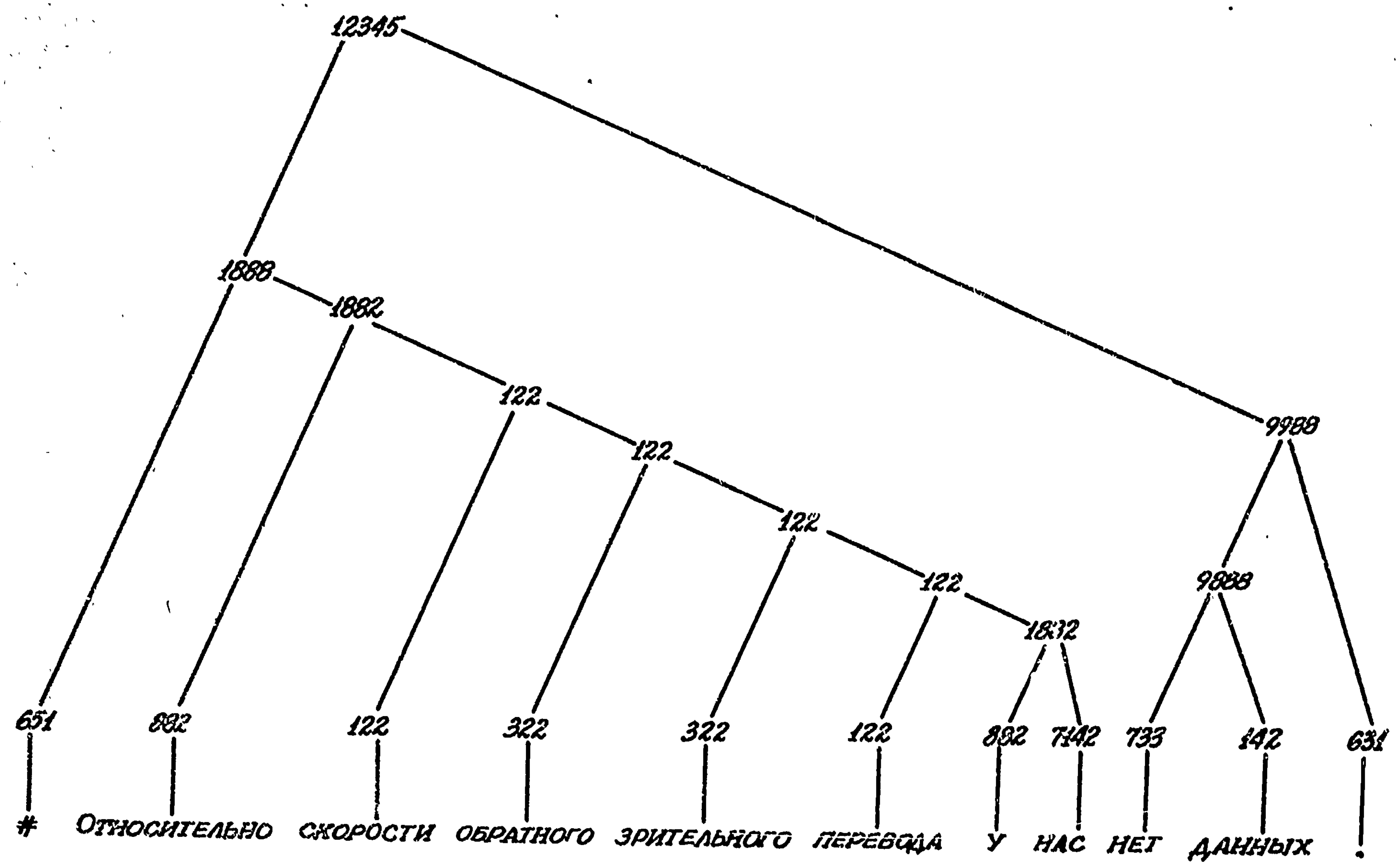
APPENDIX C

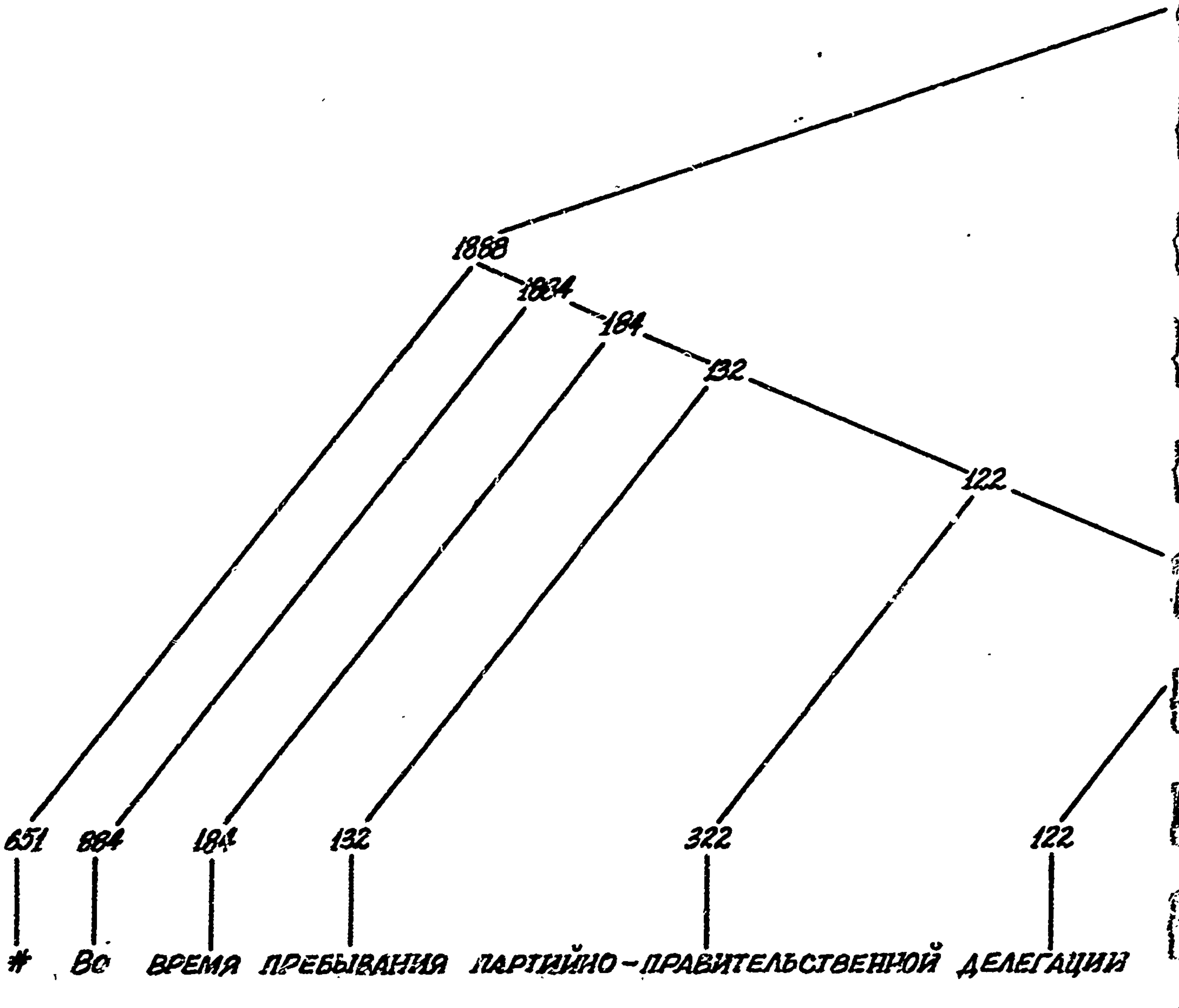

12345
9888
9228
855
121
851
#
321
ПАРТИЙНО-ПРАВИТЕЛЬСТВЕННАЯ
121
121
ДЕЛЕГАЦИЯ
112
312
СОВЕТСКОГО
112
112
СОЮЗА
1885
885
С
135
335
БОЛЬШИМ
135
УДОВЛЕТВОРЕНИЕМ
12228
ОТМЕТИЛА
611
,

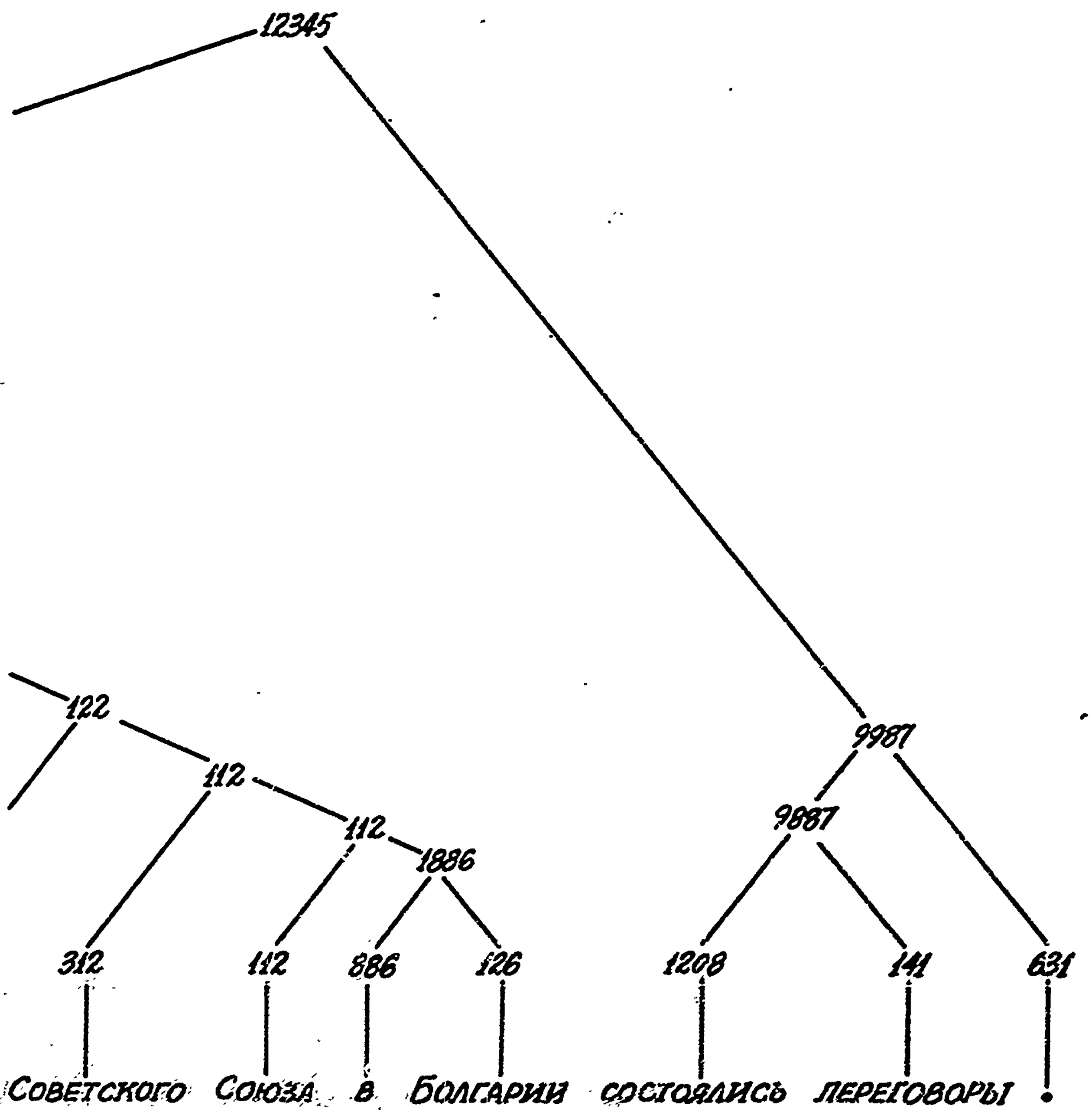

854
9988
753
1886
753
866
146
146
ЧТО
346
В
346
146
КОРОТКИЕ
ИСТОРИЧЕСКИЕ
СРОКИ
9888
121
121
321
121
112
НАРОДНАЯ
РЕСПУБЛИКА
БОЛГАРИЯ
228
142
222
342
ДОБИЛАСЬ
ОГРОМНЫХ
142
ДОСТИЖЕНИЙ
631

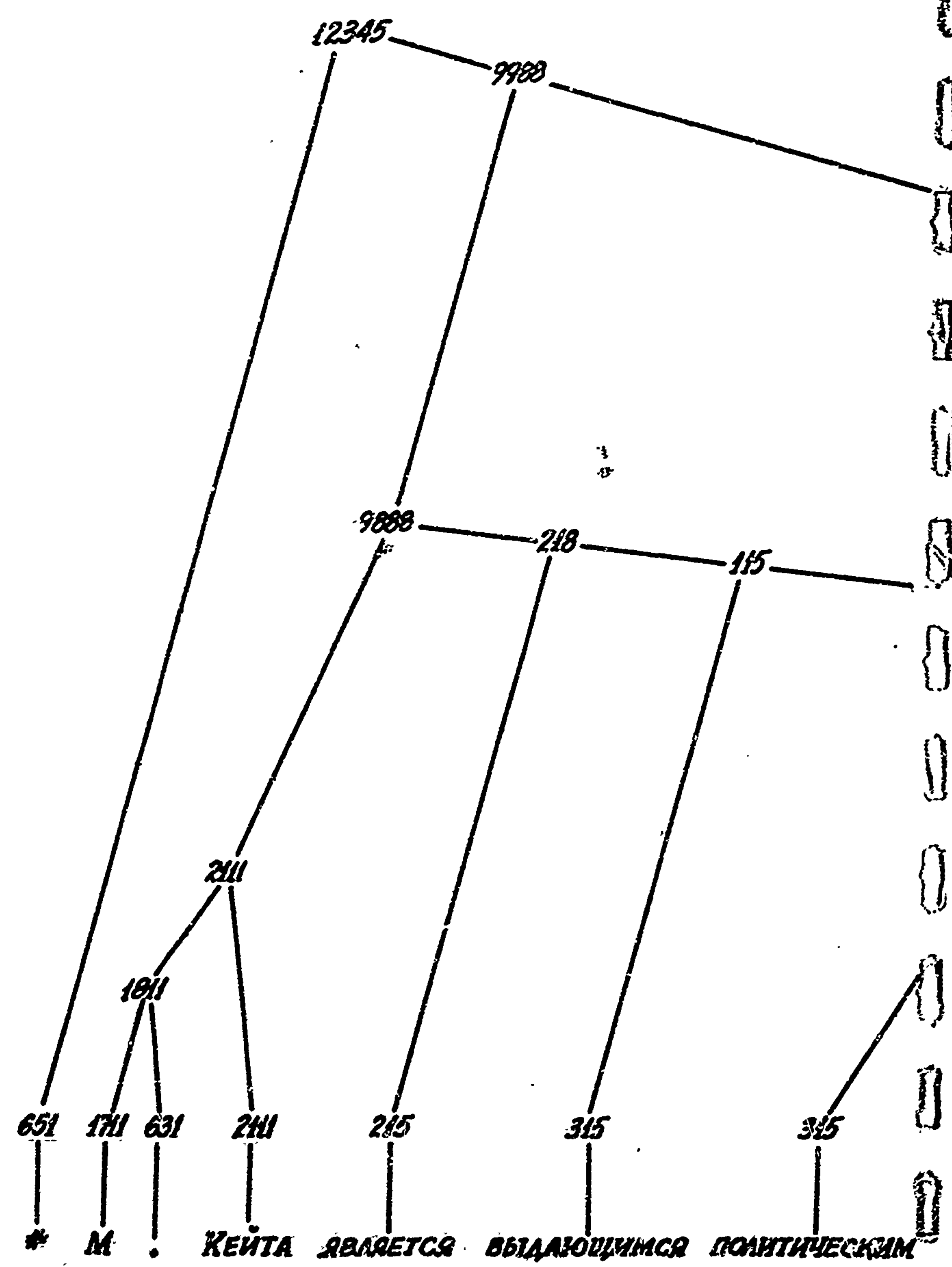


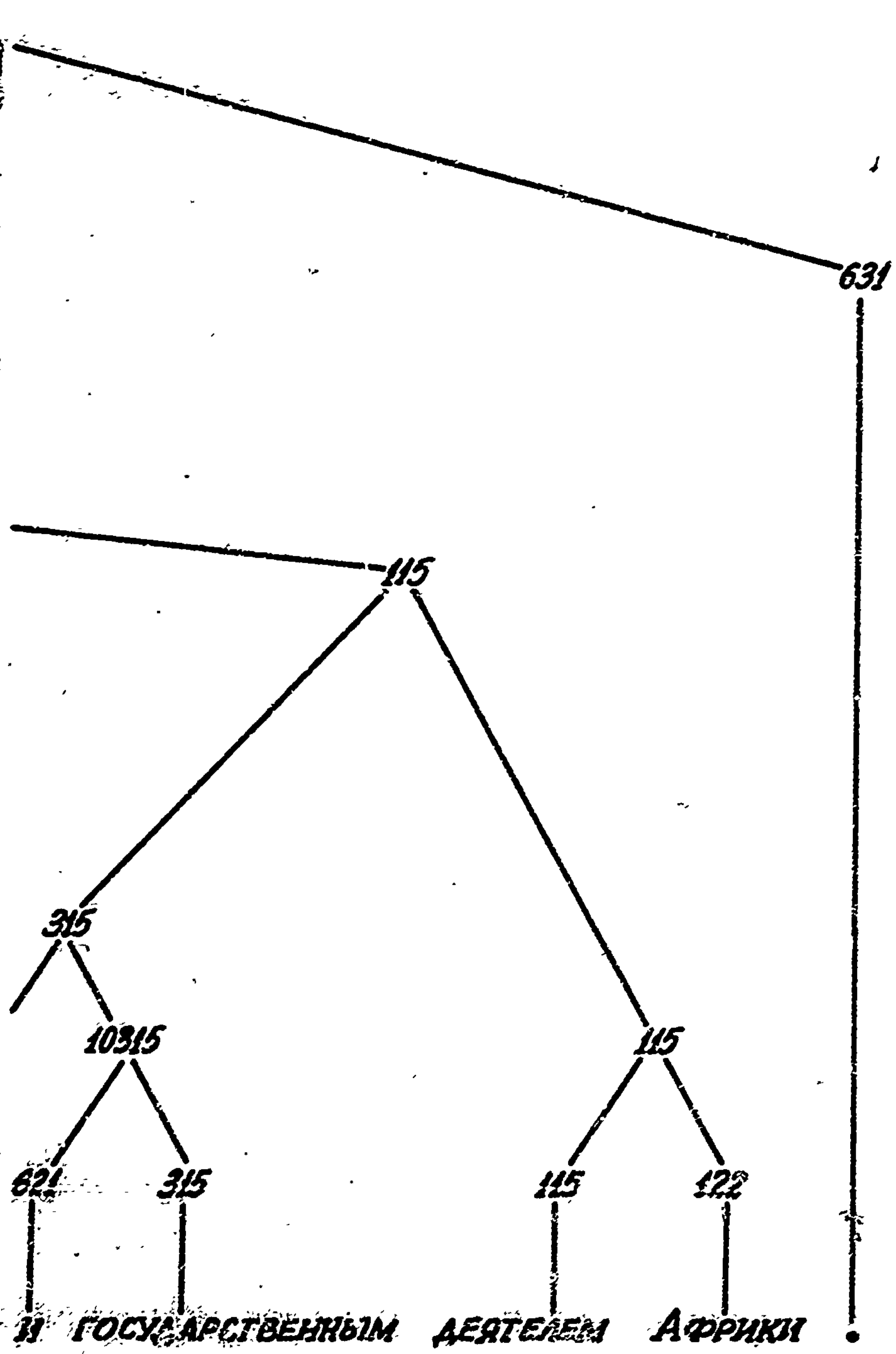
12345
9987
9887
238
123
123
4123
131
651
323
722
123
723
2233
672
142
631
#
ЛЕНИНСКОЙ
«
ПРАВДЕ
»
ИСПОЛНИЛОСЬ
50
ЛЕТ
!

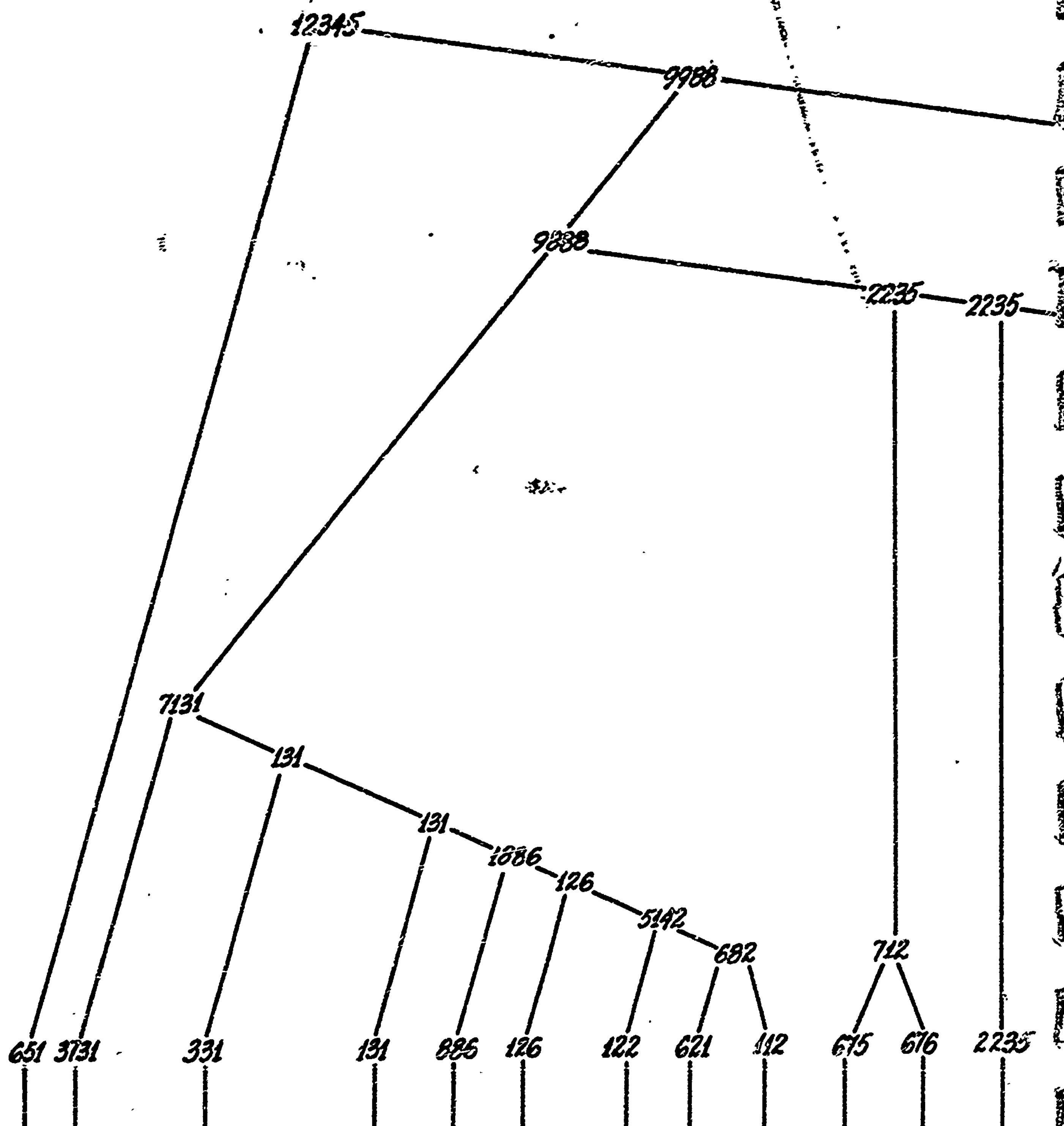

12345
1888
1882
9988
122
122
122
122
9888
1832
651
882
122
322
322
122
892
7142
733
142
631
#
ОТНОСИТЕЛЬНО
СКОРОСТИ
ОБРАТНОГО
ЗРИТЕЛЬНОГО
ПЕРЕВОДА
У
НАС
НЕТ
ДАННЫХ
!

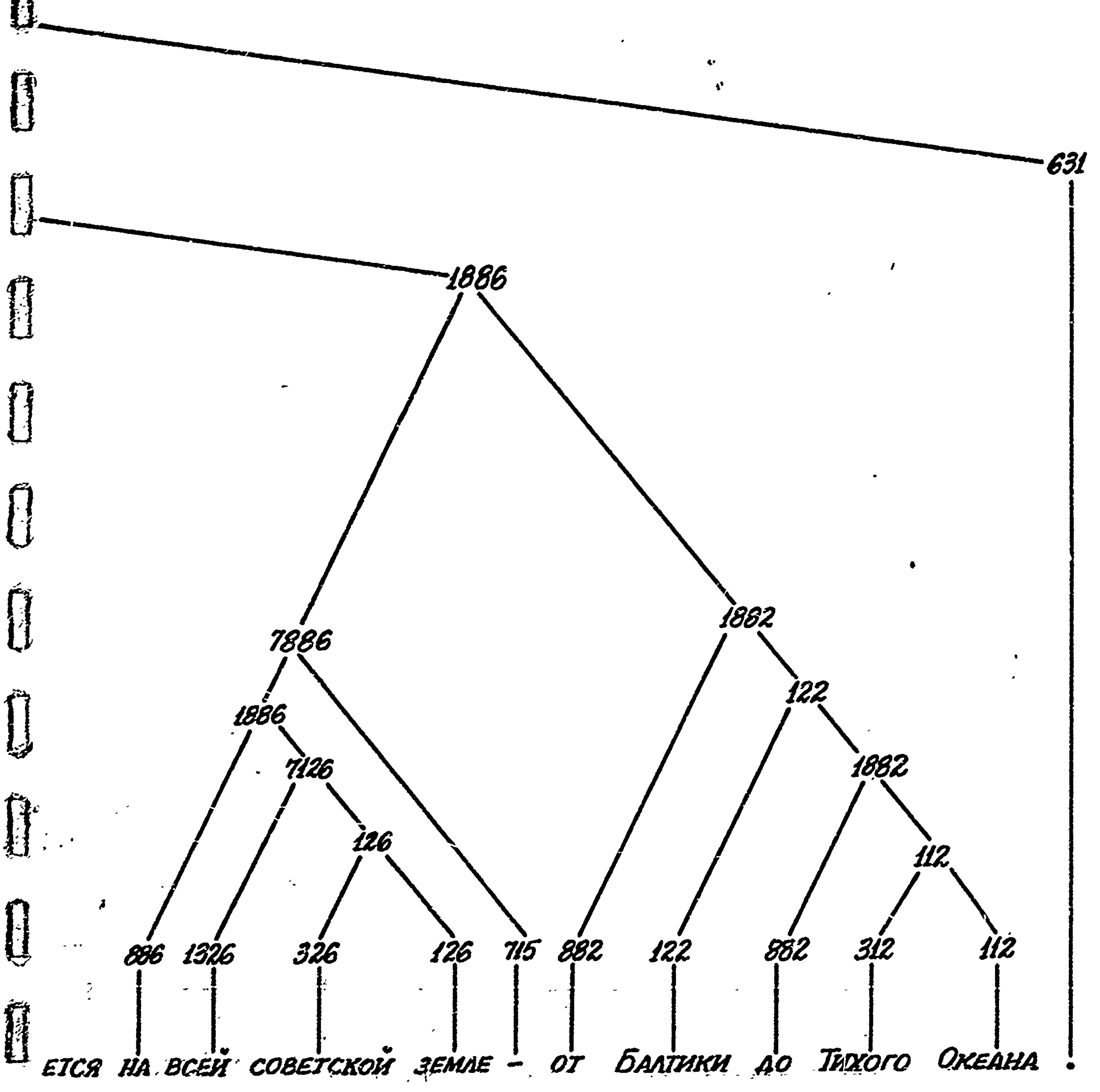

1888
184
132
122
651
884
132
322
122
#
Во
ВРЕМЯ
ПРЕБЫВАНИЯ
ПАРТИЙНО-ПРАВИТЕЛЬСТВЕННОЙ
ДЕЛЕГАЦИИ

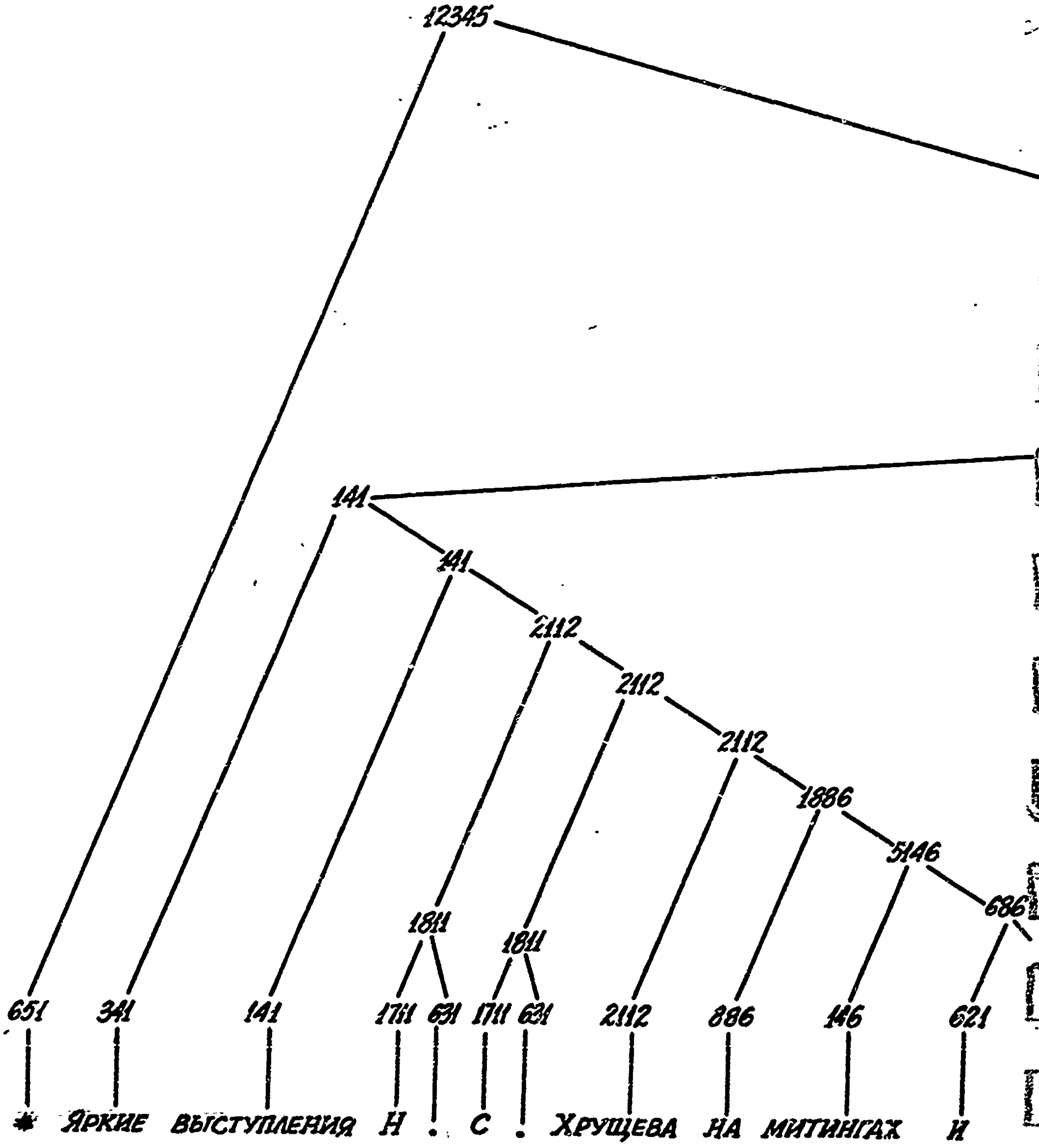

12345
122
112
112
1886
9987
9887
312
112
886
126
1208
141
631
Советского Союза в Болгарии состоялись переговоры .

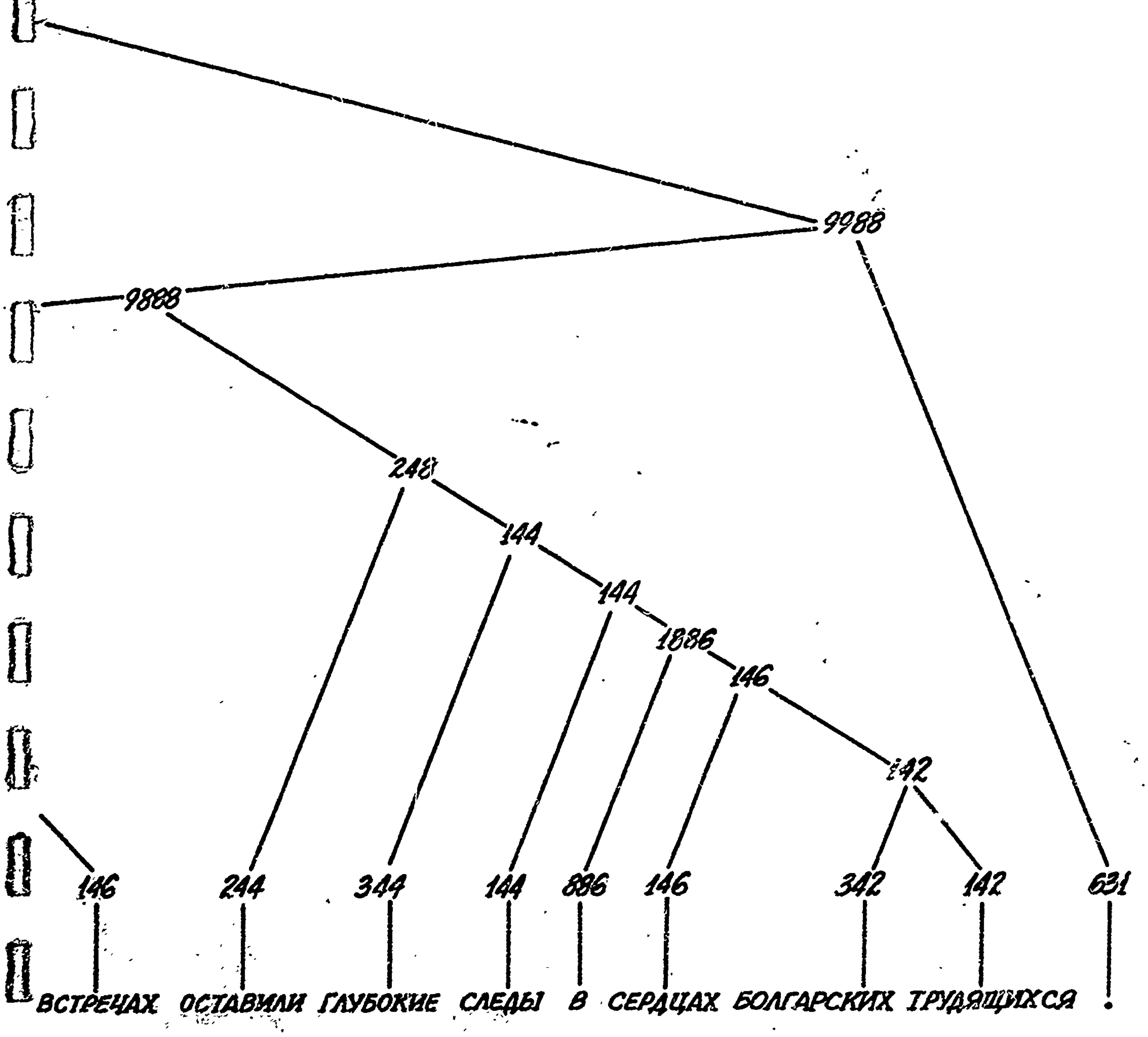

12345
9988
9888
218
115
2111
1811
651
1711
631
2111
215
315
315
#
M
.
КЕЙТА
ЯВЛЯЕТСЯ
ВЫДАЮЩИМСЯ
ПОЛИТИЧЕСКИМ

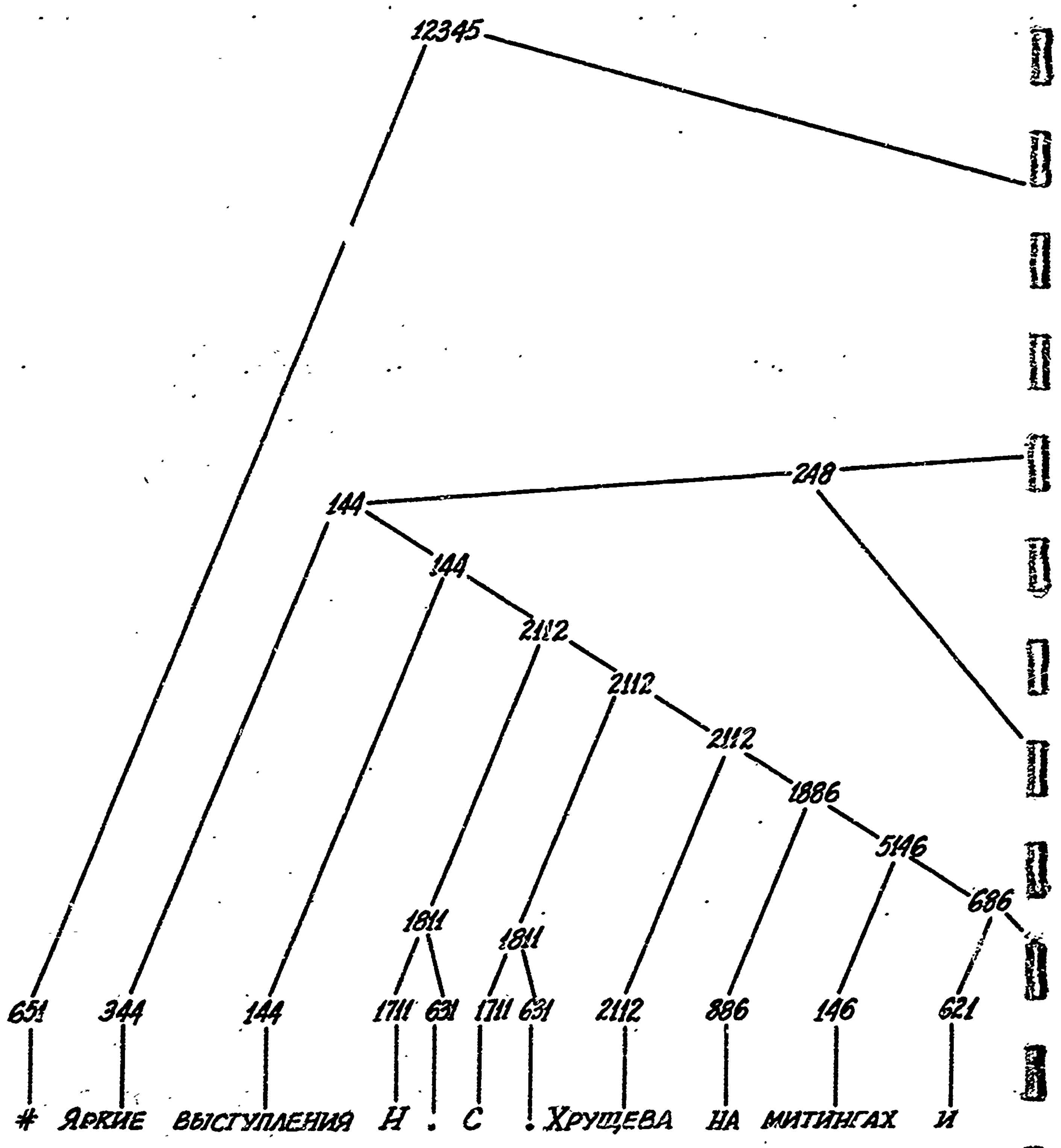

631
115
315
10315
115
621
315
115
122
И ГОСУДАРСТВЕННЫМ ДЕЯТЕЛЕМ АФРИКИ .

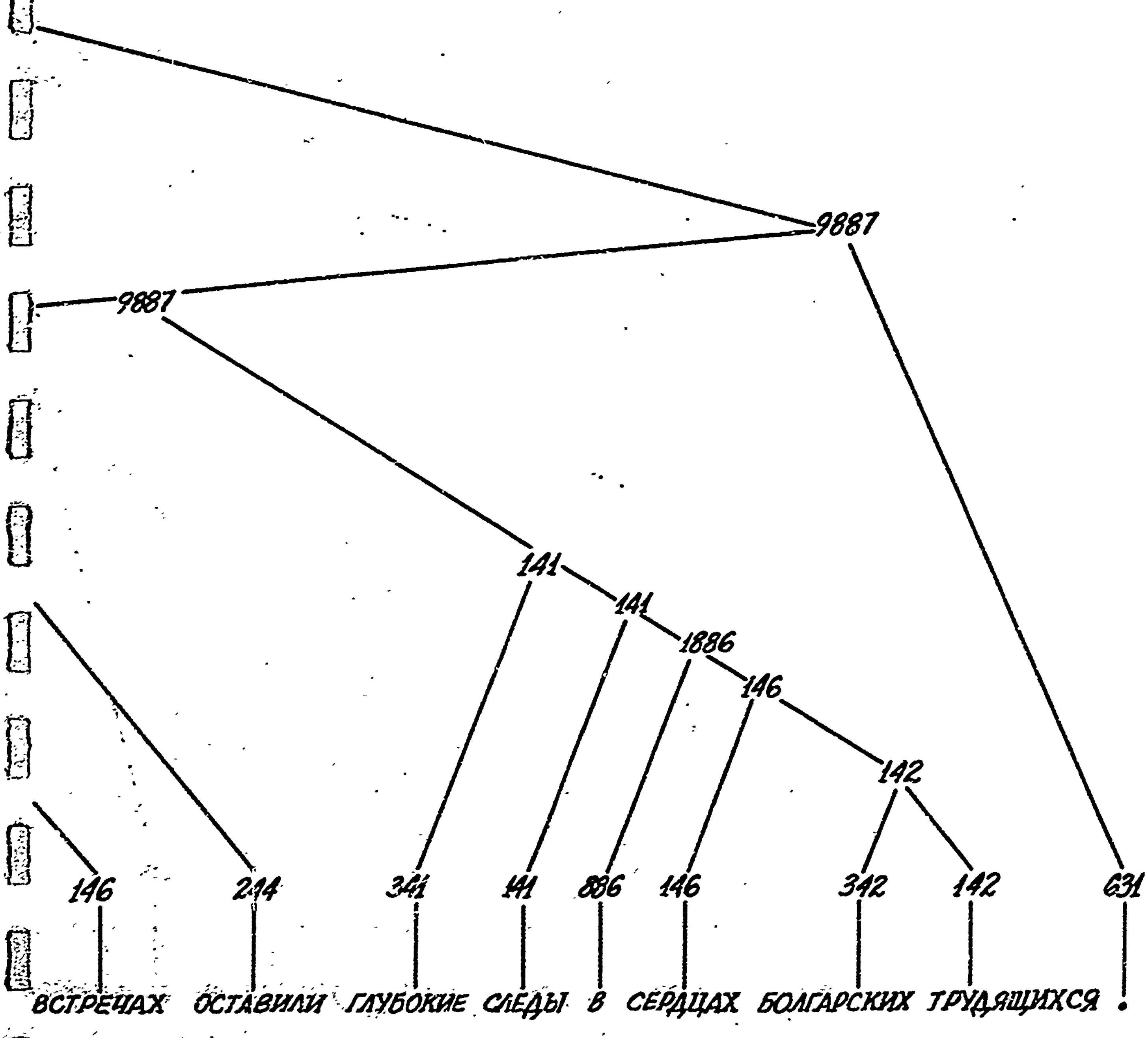

12345
9988
9888
2235
2235
7131
131
131
1886
126
5142
682
712
651
3731
331
131
886
126
122
621
112
675
676
2235
ЭТО ЗНАМЕНАТЕЛЬНОЕ СОБЫТИЕ В ЖИЗНИ ПАРТИИ И НАРОДА 5 МАЯ ОТМЕЧА

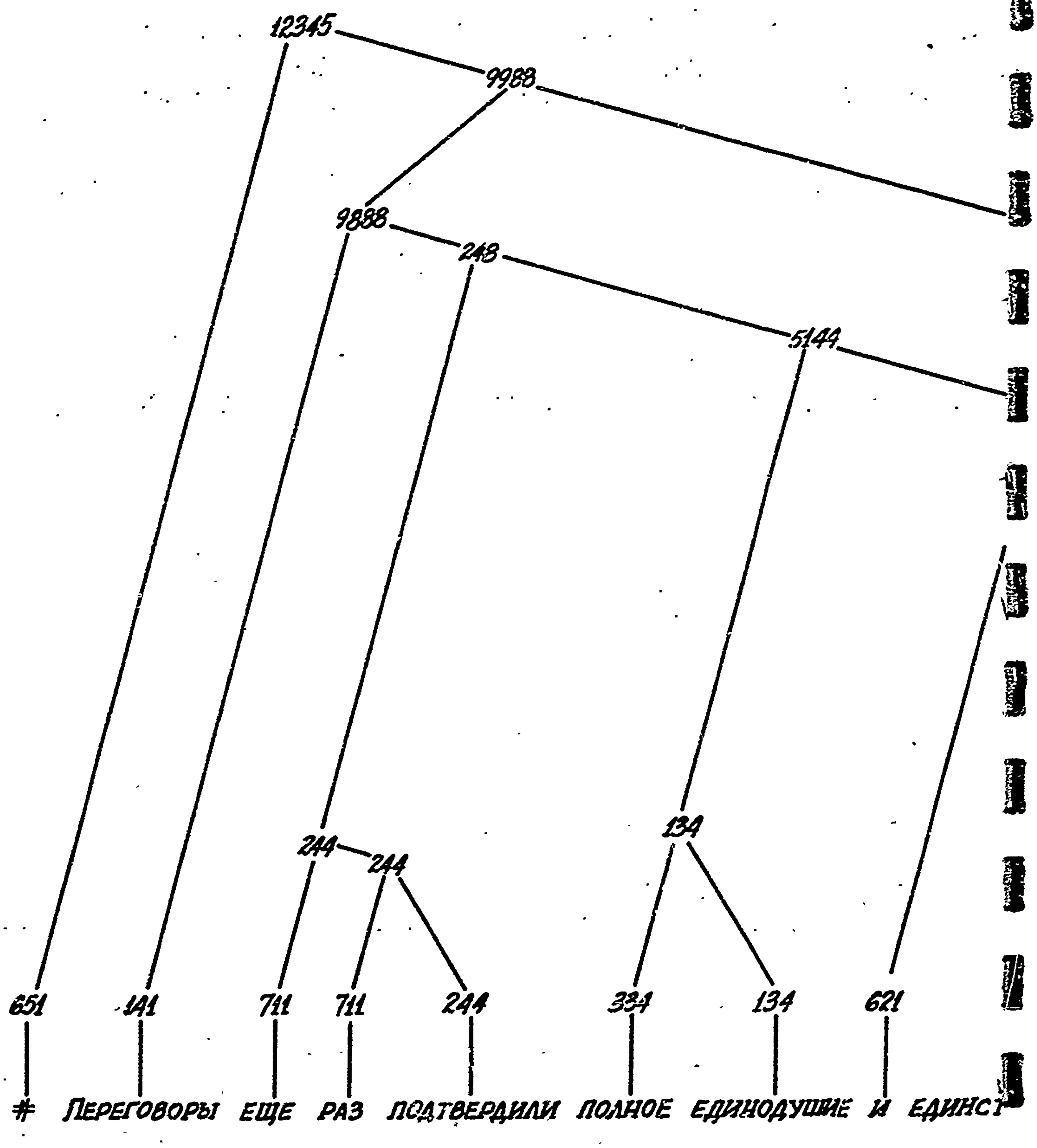

631
1886
7886
1882
1886
122
7126
1882
126
112
886 1326 326 126 715 882 122 882 312 112
ЕТСЯ НА ВСЕЙ СОВЕТСКОЙ ЗЕМЛЕ – ОТ БАЛТИКИ ДО ТИХОГО ОКЕАНА .

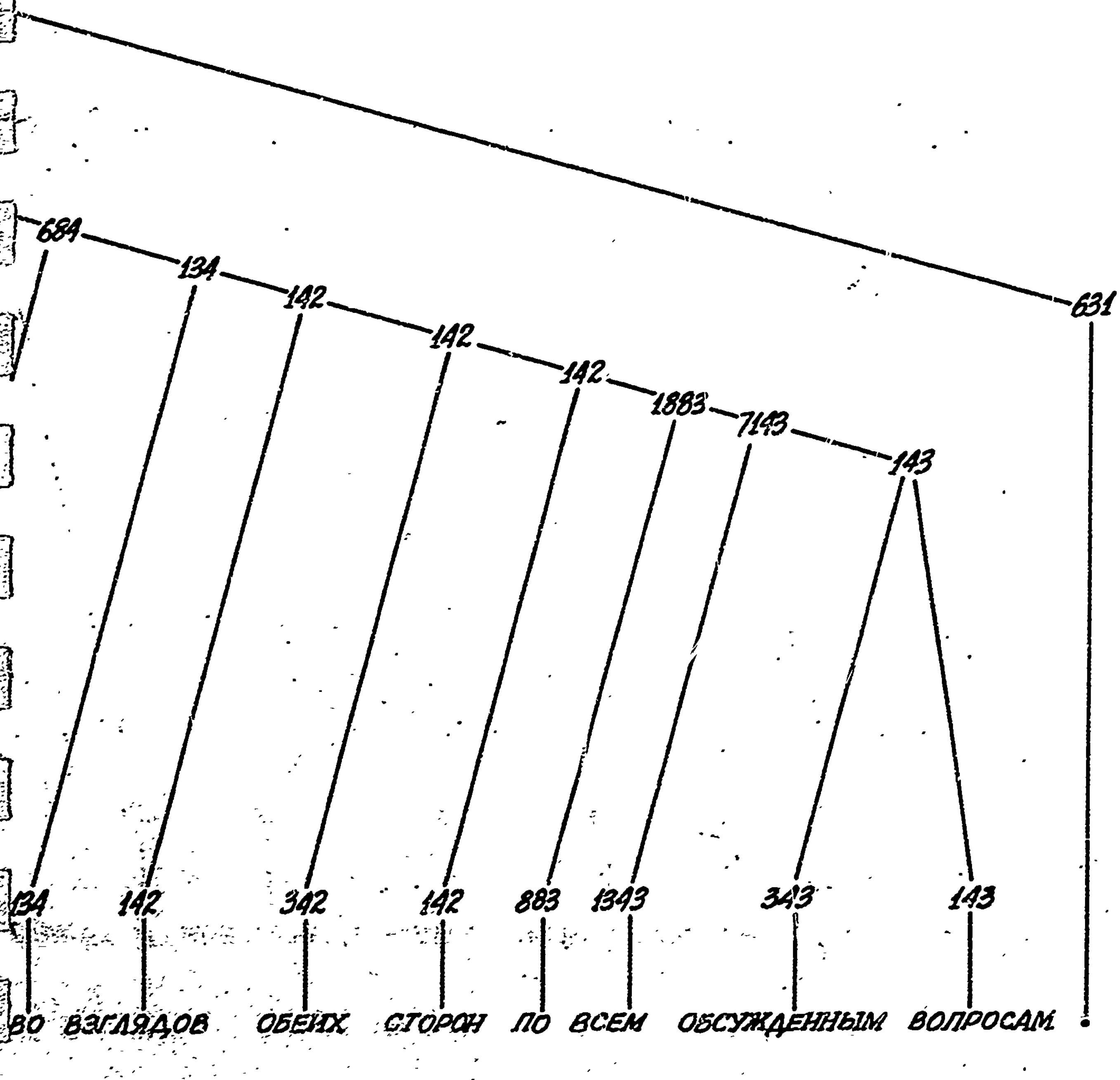

12345
141
141
2112
2112
2112
1886
5146
686
1811
1811
651 341 141 1711 631 1711 631 2112 886 146 621
* ЯРКИЕ ВЫСТУПЛЕНИЯ Н . С . ХРУЩЕВА НА МИТИНГАХ И

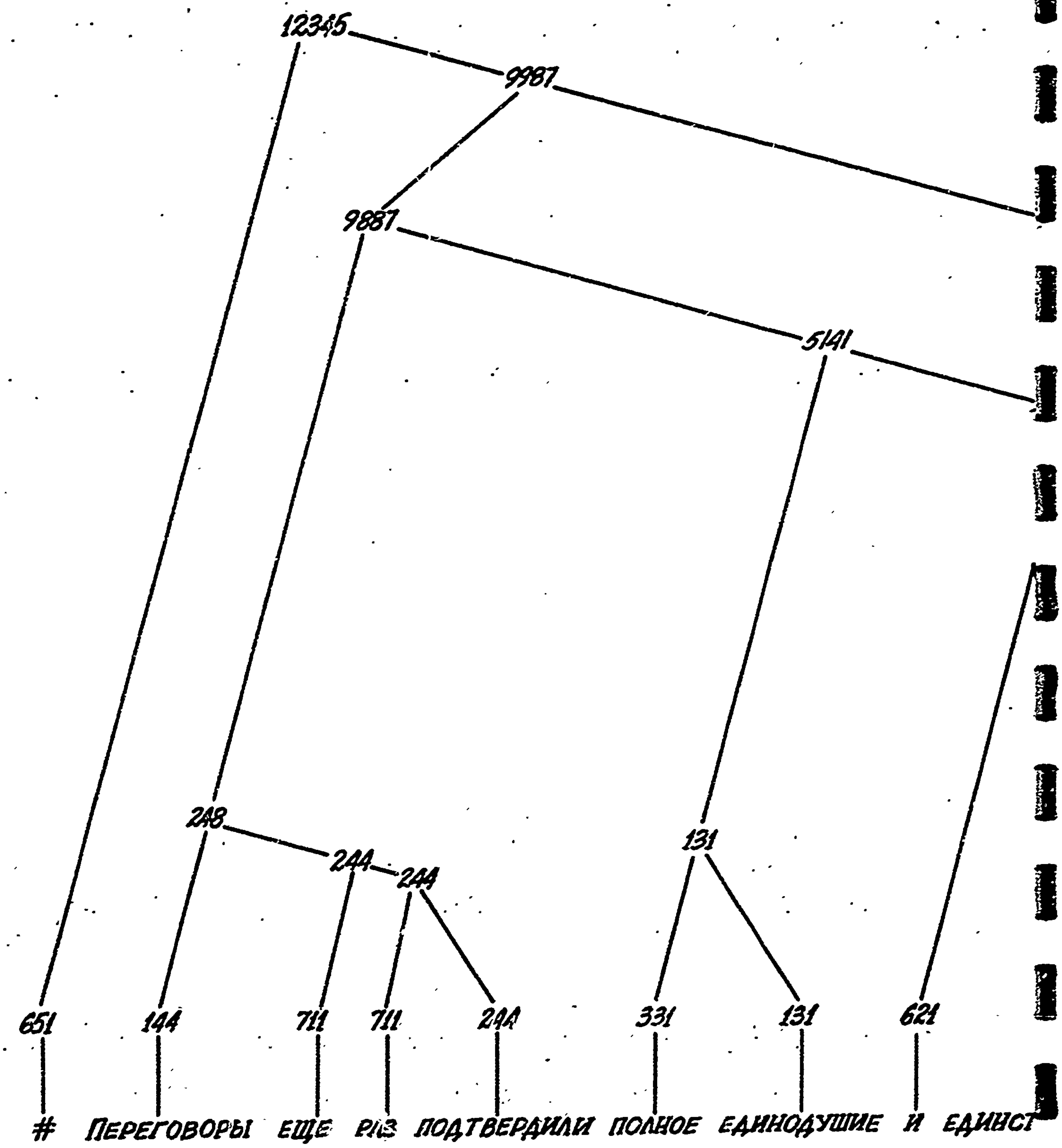

9988
9888
248
144
144
1886
146
142
146 244 344 144 886 146 342 142 631
ВСТРЕЧАХ ОСТАВИЛИ ГЛУБОКИЕ СЛЕДЫ В СЕРДЦАХ БОЛГАРСКИХ ТРУДЯЩИХСЯ .

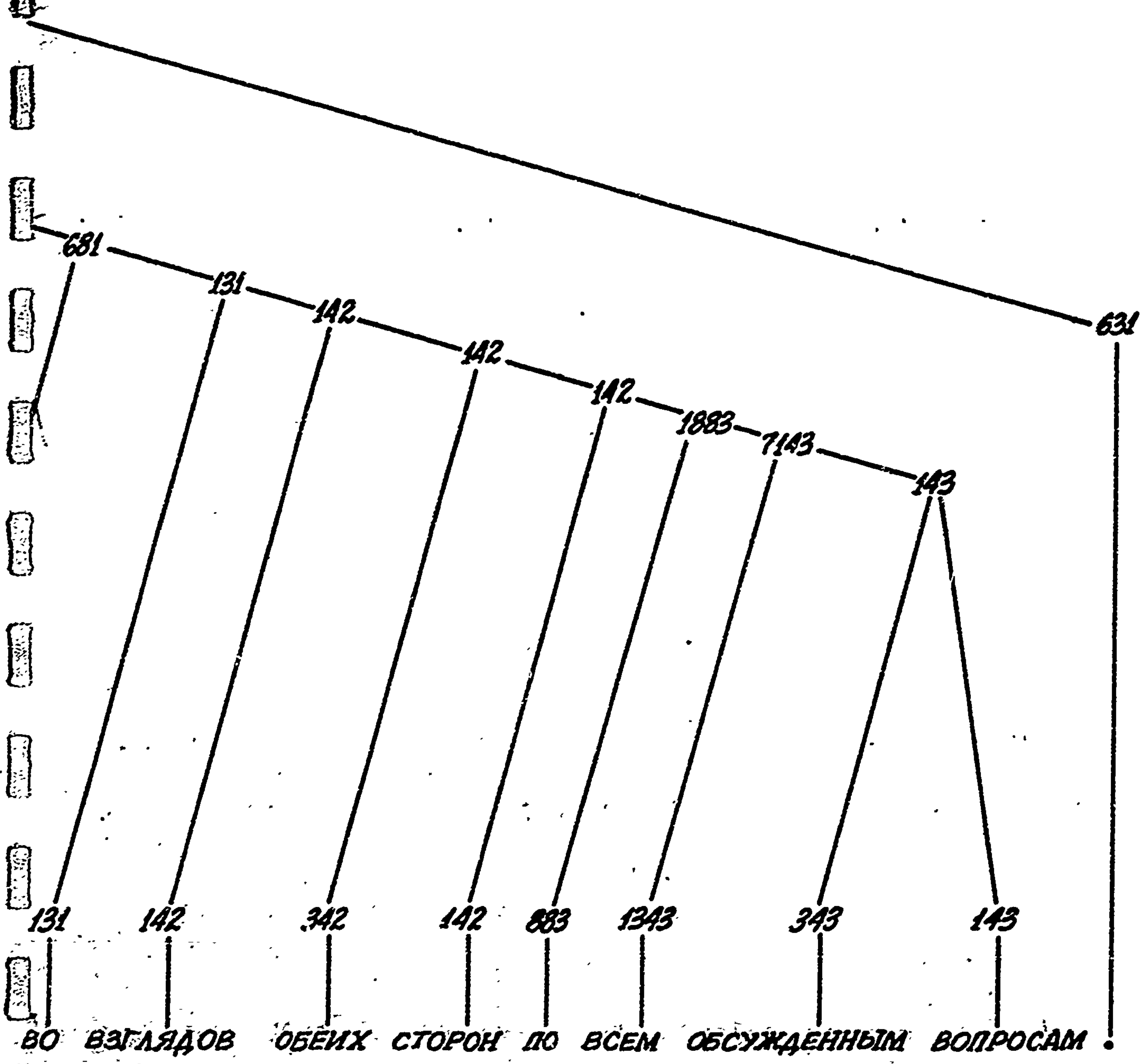

12345
248
144
144
2112
2112
2112
1886
5146
686
1811
1811
651
344
144
1711
631
1711
631
2112
886
146
621
ЯРКИЕ ВЫСТУПЛЕНИЯ Н . С . ХРУЩЕВА НА МИТИНГАХ И


9887
9887
141
141
1886
146
142
146 214 341 141 886 146 342 142 631
ВСТРЕЧАХ ОСТАВИЛИ ГЛУБОКИЕ СЛЕДЫ В СЕРДЦАХ БОЛГАРСКИХ ТРУДЯЩИХСЯ .

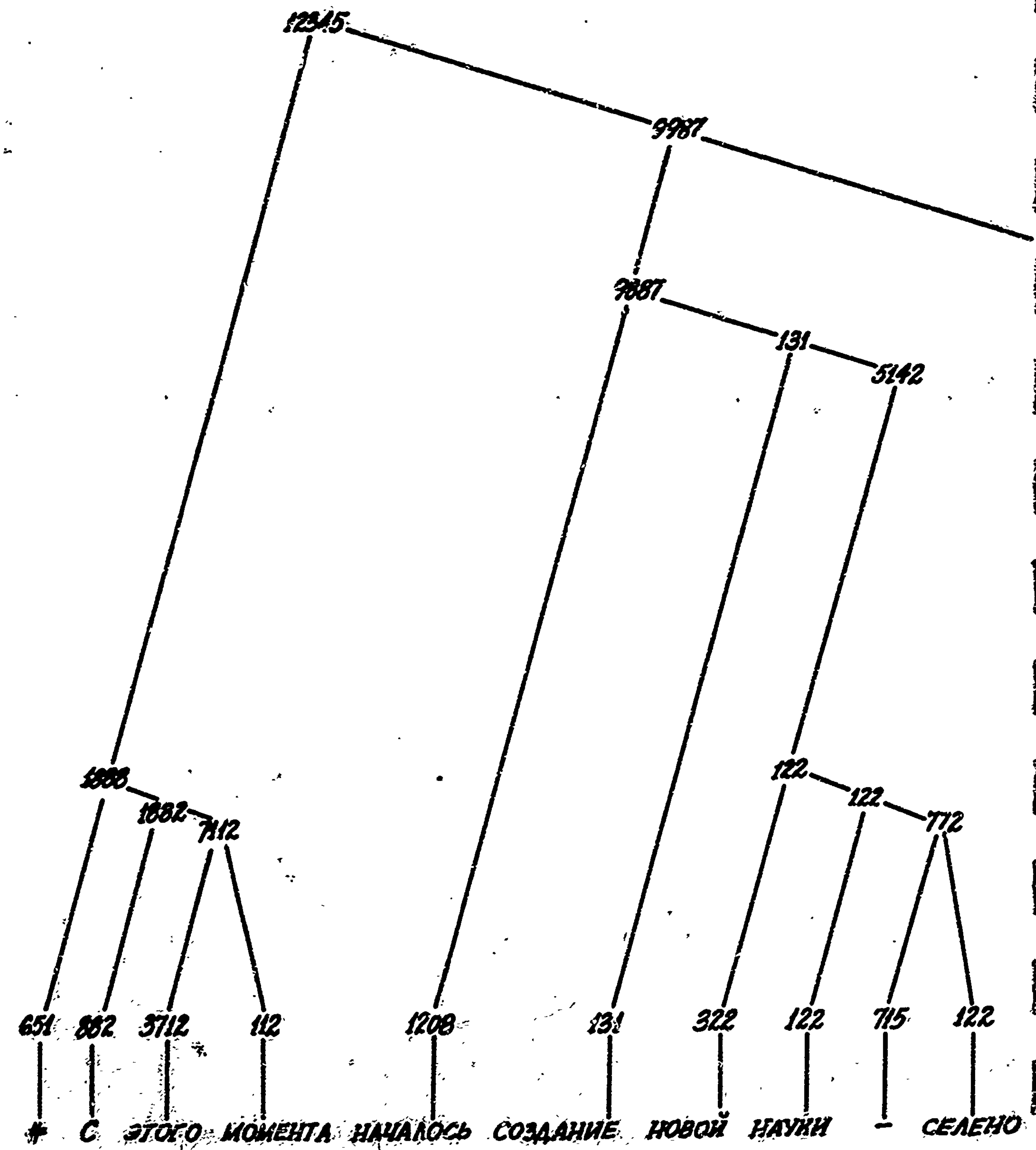

12345
9988
9888
248
5144
244
244
134
651
141
711
711
244
334
134
621
#
ПЕРЕГОВОРЫ
ЕЩЕ
РАЗ
ПОДТВЕРДИЛИ
ПОЛНОЕ
ЕДИНОДУШИЕ
И
ЕДИНСТ

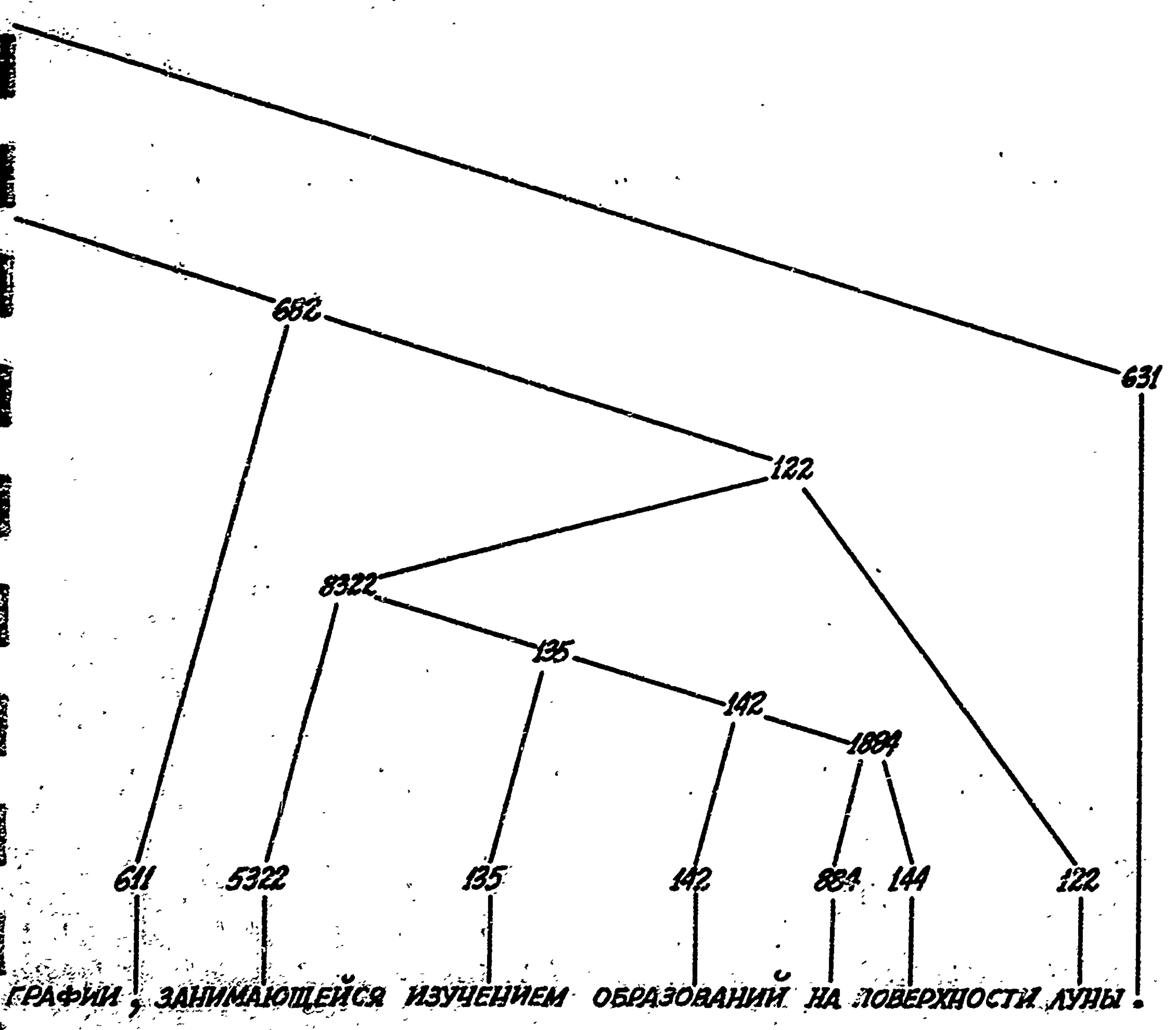

684
134
142
142
142
1883
7143
143
631
134
142
342
142
883
1343
343
143
ВО ВЗГЛЯДОВ ОБЕИХ СТОРОН ПО ВСЕМ ОБСУЖДЕННЫМ ВОПРОСАМ


12345
9987
9887
5141
248
244
244
131
651
144
711
711
244
331
131
621
ПЕРЕГОВОРЫ ЕЩЕ РАЗ ПОДТВЕРДИЛИ ПОЛНОЕ ЕДИНОДУШИЕ И ЕДИНСТ

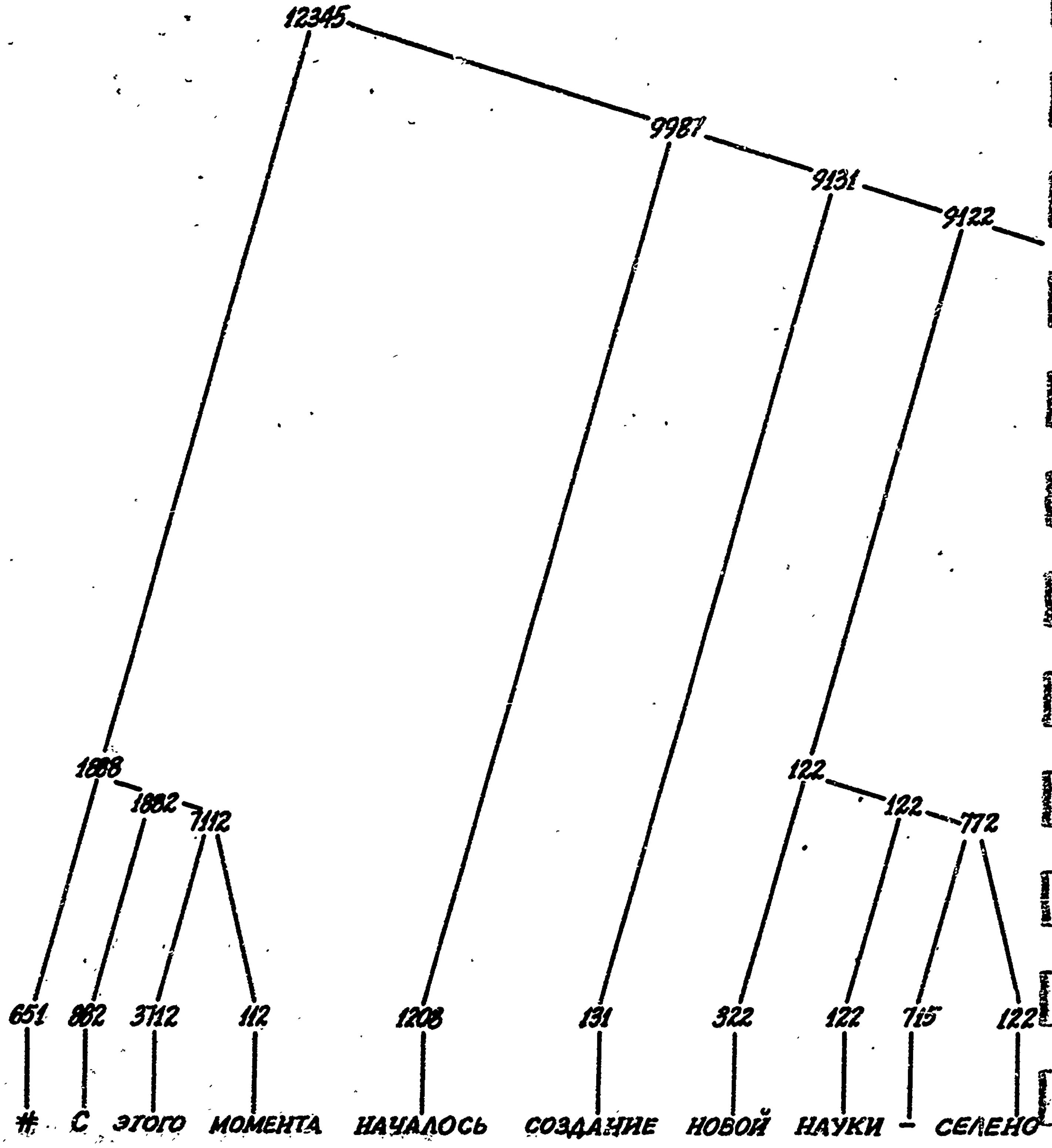


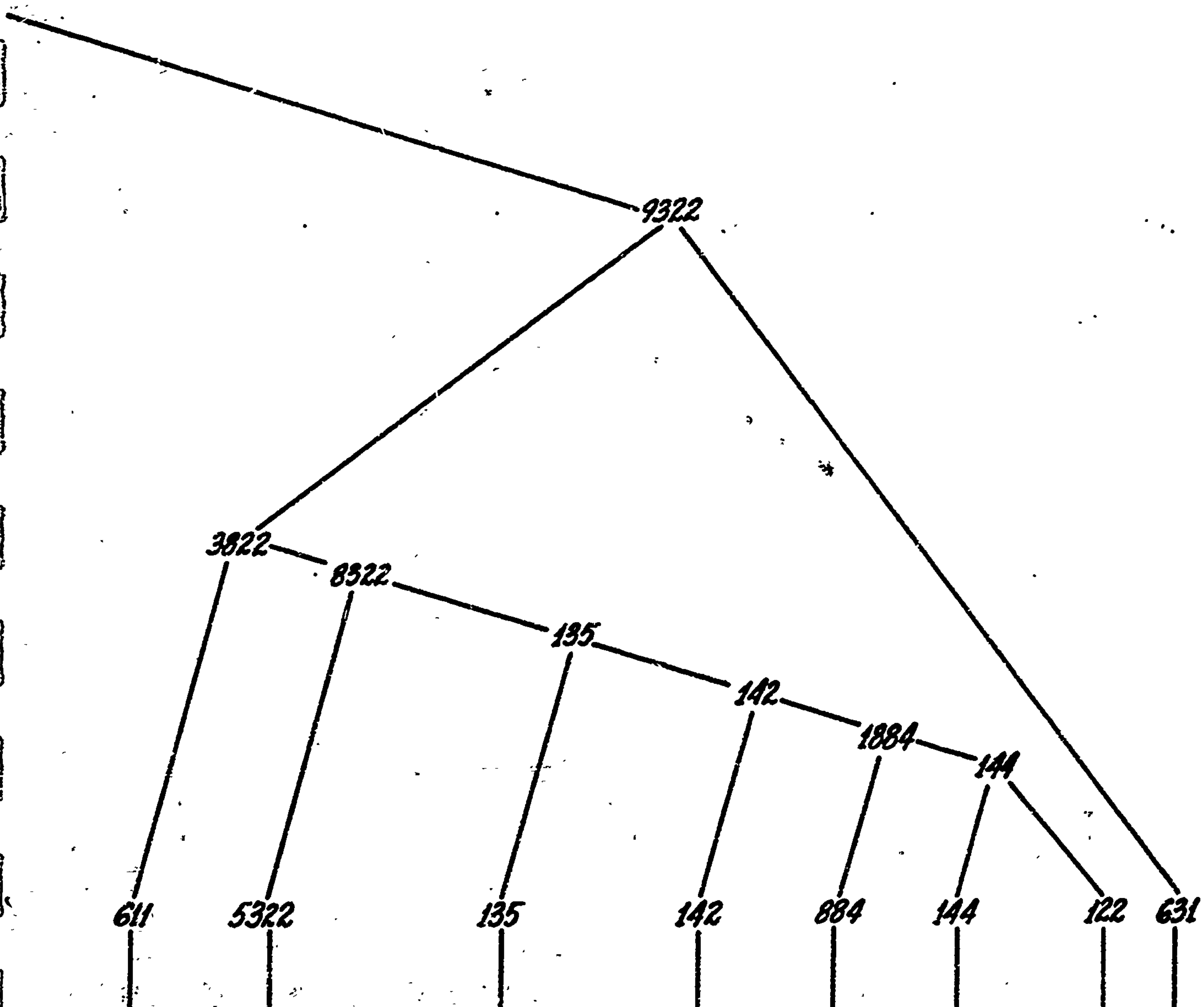
681
131
142
142
142
1883
7143
143
631
131
142
342
142
883
1343
343
143
ВО ВЗГЛЯДОВ ОБЕИХ СТОРОН ПО ВСЕМ ОБСУЖДЕННЫМ ВОПРОСАМ .


12345
9987
9887
131
5142
1888
1882
7112
122
122
772
651
882
3712
112
1208
131
322
122
715
122
С ЭТОГО МОМЕНТА НАЧАЛОСЬ СОЗДАНИЕ НОВОЙ НАУКИ — СЕЛЕНО

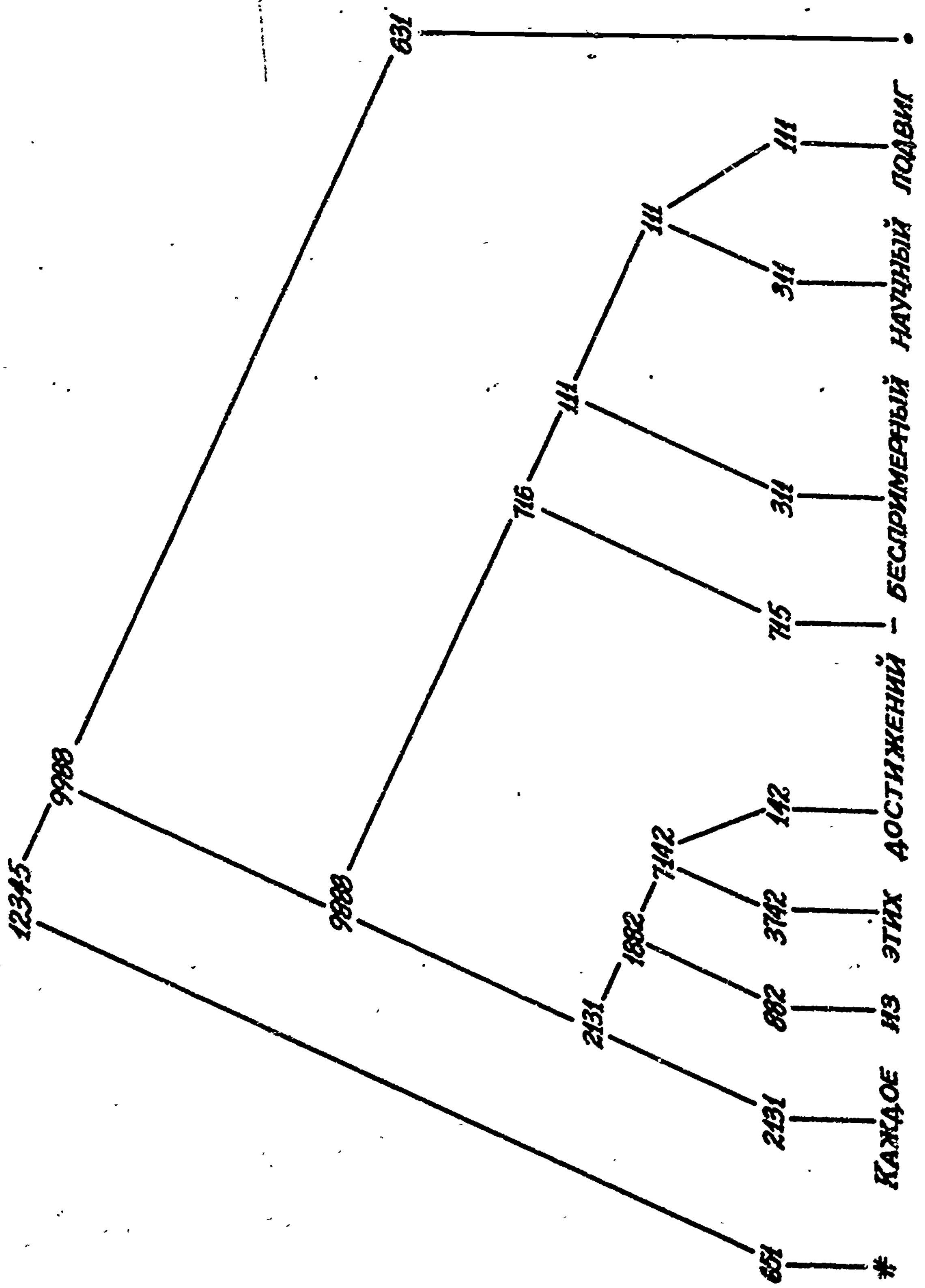


682
631
122
8322
135
142
1884
611
5322
135
142
884
144
122
ГРАФИИ , ЗАНИМАЮЩЕЙСЯ ИЗУЧЕНИЕМ ОБРАЗОВАНИЙ НА ПОВЕРХНОСТИ ЛУНЫ .


12345
9987
9131
9122
1888
1882
7112
122
122
772
651
882
3712
112
1208
131
322
122
715
122
#
С
ЭТОГО
МОМЕНТА
НАЧАЛОСЬ
СОЗДАНИЕ
НОВОЙ
НАУКИ
—
СЕЛЕНО



ГРАФИИ , ЗАНИМАЮЩЕЙСЯ ИЗУЧЕНИЕМ ОБРАЗОВАНИЙ НА ПОВЕРХНОСТИ ЛУНЫ .


12345
9988
831
9888
716
111
111
111
311
311
715
2131
1882
7142
142
3742
882
2131
651
*
КАЖДОЕ
ИЗ
ЭТИХ
ДОСТИЖЕНИЙ
–
БЕСПРИМЕРНЫЙ
НАУЧНЫЙ
ПОДВИГ
.

APPENDIX D

GRAMMAR RULES

1.	-100	192	100
2.	101	208	9888
3.	190	200	208
4.	-101	200	9860
5.	-190	480	488
6.	100	680	5140
7.	-101	641	9988
8.	100	770	100
9.	-101	716	9888
10.	-100	986	988
11.	-101	1208	9888
12.	-100	1723	100
13.	-100	1889	100
14.	-100	2192	100
15.	190	2200	208
16.	101	2200	9888
17.	-101	2200	9780
18.	-100	2889	9100
19.	190	3200	3208
20.	-101	3208	9888
21.	-101	4208	9888
22.	-100	5142	100
23.	-100	6110	140
24.	-100	6308	8100
25.	-100	7192	100
26.	-100	8408	9100
27.	-100	9192	9100
28.	-101	9208	9988
29.	-100	9300	9100
30.	-190	9980	9888
31.	-101	13208	9888
32.	-100	13900	10125
33.	-101	15208	9888
34.	200	190	208
35.	208	101	9887
36.	-200	101	9850
37.	-200	711	200
38.	-200	1889	200
39.	200	2190	1208
40.	-200	2101	9770
41.	200	5140	208
42.	-248	5141	9887
43.	200	7190	208
44.	200	7101	9850
45.	201	7101	9887
46.	208	7101	9887
47.	-200	7190	1208
48.	208	9101	9987
49.	-200	9190	9208
50.	-204	9688	9208
51.	-300	100	100
52.	-300	3100	3100
53.	-300	10300	300
54.	478	107	488
55.	458	105	488

56.	448	104	488
57.	438	103	488
58.	428	102	488
59.	-400	190	408
60.	-580	100	588
61.	-580	611	578
62.	-580	711	580
63.	-578	9900	9900
64.	672	102	131
65.	672	102	134
66.	621	100	680
67.	-611	100	680
68.	611	300	10300
69.	-621	300	10300
70.	-675	676	712
71.	611	757	758
72.	671	712	671
73.	651	712	651
74.	-611	755	756
75.	-611	854	855
76.	-651	1880	1888
77.	-611	6508	6308
78.	621	7100	680
79.	-611	7100	680
80.	-611	7408	8408
81.	-611	8300	3800
82.	-611	9687	9688
83.	621	9800	1511
84.	-671	9880	678
85.	-651	9900	12345
86.	722	100	4100
87.	722	101	1722
88.	715	101	716
89.	733	102	9888
90.	-715	100	770
91.	777	202	202
92.	777	203	203
93.	777	204	204
94.	777	205	205
95.	711	201	201
96.	711	202	202
97.	711	203	203
98.	711	204	204
99.	711	205	205
100.	711	207	207
101.	-712	208	208
102.	754	631	757
103.	754	611	755
104.	-717	621	717
105.	-778	779	712
106.	-753	1880	753
107.	-715	2101	9888
108.	711	2200	2200
109.	712	2202	2202
110.	712	2203	2203
111.	712	2204	2204
112.	712	2205	2205
113.	-712	2207	2207
114.	-711	2300	2300

115.	-711	3200	3200
116.	-711	3300	3300
117.	-711	4300	4300
118.	-717	5141	9887
119.	-711	5300	5300
120.	-711	7300	7300
121.	-715	9100	9770
122.	753	9887	754
123.	-753	9888	754
124.	-753	9980	854
125.	-880	100	1880
126.	-880	2100	1880
127.	-880	5100	1880
128.	-880	6300	6800
129.	-880	7100	1880
130.	-880	9100	2880
131.	-1208	101	9887
132.	-1208	1889	1208
133.	-1208	2101	9887
134.	-1208	2890	9208
135.	-1288	7101	9887
136.	-1208	9101	9987
137.	-1300	100	7100
138.	-1300	7100	7100
139.	-1711	631	1811
140.	-1722	723	1723
141.	-1711	885	885
142.	-1880	715	7880
143.	-1811	2100	2100
144.	-1888	9900	12345
145.	-1888	12208	12208
146.	-2100	192	2100
147.	2101	200	9780
148.	2101	208	9888
149.	-2190	200	1208
150.	-2101	716	9888
151.	-2101	1208	9888
152.	-2100	1889	2100
153.	2190	2200	1208
154.	2101	2200	9780
155.	-2101	2200	9888
156.	2101	2308	9888
157.	-2190	2300	2308
158.	-2101	3208	9888
159.	-2100	6308	2100
160.	-2100	9300	2100
161.	-2190	9780	9888
162.	2200	101	9770
163.	2200	101	9887
164.	-2200	190	208
165.	2204	758	208
166.	-2200	711	2200
167.	-2200	1889	2200
168.	2200	2101	9887
169.	-2200	2101	9770
170.	2200	7190	1208
171.	2200	7101	9770
172.	-2200	7101	9887
173.	-2204	9688	2208

174.	2300	100	100
175.	-2300	192	8300
176.	-2300	5192	8300
177.	-2300	7192	8300
178.	3100	100	100
179.	-3100	192	3100
180.	3100	2100	2100
181.	-3100	2192	3100
182.	-3100	7192	3100
183.	3208	101	9887
184.	3208	100	9887
185.	-3200	190	3208
186.	-3200	711	3200
187.	-3200	1889	3200
188.	3208	2101	9888
189.	3208	2101	9887
190.	-3200	2190	3208
191.	-3208	4200	3208
192.	3218	5259	3258
193.	3208	5209	3248
194.	3228	5259	3258
195.	3248	5289	3288
196.	3248	5279	3278
197.	3218	5269	3268
198.	-3228	5269	3268
199.	3208	7101	9888
200.	3208	7101	9887
201.	-3200	7190	3208
202.	3300	100	100
203.	-3300	193	8300
204.	-3300	5193	8300
205.	-3300	7193	8300
206.	-3700	100	7100
207.	-3700	7100	7100
208.	3800	611	9650
209.	-3800	631	9300
210.	-4100	723	100
211.	-4208	100	9887
212.	-4200	488	4208
213.	-4200	3208	3208
214.	-4208	7101	9887
215.	4300	100	100
216.	-4300	194	8300
217.	-4300	5194	8300
218.	-4300	7194	8300
219.	5140	200	208
220.	-5141	248	9888
221.	5141	641	9988
222.	-5140	680	5140
223.	-5141	716	9888
224.	-5141	1208	9888
225.	5141	2200	9888
226.	-5140	2200	208
227.	5140	3200	3200
228.	-5141	3248	9888
229.	-5101	4208	9888
230.	-5100	6308	5100
231.	-5100	8408	9100
232.	-5141	9248	9988

233.	-5100	9300	5100
234.	5269	3218	3268
235.	5249	3248	3248
236.	5259	3218	3258
237.	5259	3228	3259
238.	5269	3228	3268
239.	5279	3248	3278
240.	-5289	3248	3288
241.	-5209	10200	10200
242.	5300	100	100
243.	-5300	195	8300
244.	-5300	5195	8300
245.	-5300	7195	8300
246.	6101	208	9677
247.	-6190	200	9608
248.	-6100	1889	6100
249.	-6100	9860	9677
250.	-6301	208	6408
251.	-6300	712	6300
252.	-6301	1208	6408
253.	-6301	2200	6408
254.	-6301	3200	6408
255.	-6301	9208	7408
256.	-6300	9980	6408
257.	-6301	15208	6408
258.	6408	631	7408
259.	-6408	611	6508
260.	-6888	100	100
261.	6888	202	202
262.	6888	203	203
263.	6888	204	204
264.	6888	205	205
265.	-6888	207	207
266.	-6888	1208	1208
267.	-6888	2100	2100
268.	-6888	2200	2200
269.	-6888	3200	3200
270.	-6888	4299	4299
271.	-6888	7100	7100
272.	6800	9988	6408
273.	-6800	9987	6408
274.	7101	208	9888
275.	7190	200	208
276.	-7101	200	9860
277.	-7100	480	488
278.	-7100	680	5140
279.	-7101	1208	9888
280.	-7100	1889	7100
281.	7101	2200	9888
282.	-7190	2200	1208
283.	7190	2300	2308
284.	-7101	2308	9888
285.	-7101	3208	9888
286.	-7101	4208	9888
287.	-7100	6308	100
288.	-7101	9208	9938
289.	-7100	9300	7100
290.	7100	9770	9100
291.	-7190	9780	9888

292.	-7101	13208	9888
293.	-7101	15208	9888
294.	7300	100	100
295.	-7300	197	8300
296.	-7300	5197	8300
297.	-7300	7197	8300
298.	-7400	7197	8400
299.	-7880	1889	1880
300.	-8300	100	100
301.	-8300	1899	8300
302.	-8400	488	8300
303.	-9190	190	5140
304.	-9190	7190	5140
305.	-9608	101	9677
306.	-9677	631	9687
307.	-9608	7101	9677
308.	-9770	2100	9887
309.	-9770	7100	9887
310.	-9850	100	9887
311.	-9880	631	9980
312.	-9800	1511	9800
313.	-10200	756	9887
314.	-10238	855	9987
315.	-12208	758	208
316.	-12208	855	9208
317.	-13200	488	13208
318.	-15200	488	15208