SBORNÍK PRACÍ FILOZOFICKÉ FAKULTY BRNĚNSKÉ UNIVERZITY
STUDIA MINORA FACULTATIS PHILOSOPHICAE UNIVERSITATIS BRUNENSIS
K 1 (1979) – BRNO STUDIES IN ENGLISH 13

THE ORDERED-TRIPLE THEORY CONTINUED

A. Svoboda et al.

CONTENTS

INTRODUCTION

In our previous paper entitled An ordered-triple theory of language (Brno Studies in English 12), we have voiced some more or less general views on how to both generate and analyze expressions of a natural language if we take into account all the three components of semiotics: syntactics, semantics and pragmatics. Since 1973, when the manuscript was submitted to the editor, our team have focussed efforts on (i) elaborating the system of semantics and semantic analysis and (ii) testing a computer program of syntactic analysis compatible with the designed systems of semantic and pragmatic analyses.

Since we regard semantics as the keystone of our theory, we have paid much attention to the choice and development of a system of logical semantics satisfying the requirements of the semantics of a natural language. It is our team's logician, P. Materna, who assumed the responsibility for choosing Tichý's intensional logic and shaping it according to our demands. Most relevant sections of this semantic theory form the core of Part One of the present paper.

The computer program of syntactic analysis has been designed and tested by K. Pala, who also supervised the work of our programmers I. Palová and P. Čihánek. The results of their endeavour can be found in Part Two.

PART ONE

Chapter I

AGAINST EXTENSIONALISM

To solve the problem of an adequate analysis of language expressions is hardly thinkable without a carefully elaborated theoretical conception, which — in our opinion — is also a conditio sine qua non for the subsequent computer simulation of some important fragments of language communication. According to Morris (1946), any general theory of language has to make the distinction between syntactics, semantics, and pragmatics. In addition to that, such a theory has also to study the interplay of the above three components. As has been said before, Part One is mainly to deal with the semantic component of a general theory of linguistic analysis, its main task being to find and elaborate a method that would enable us to decipher what the language expressions are about; possessing this method would result in our ability to discover for any particular language the laws connecting the surface structures of its expressions, particularly words, phrases, and clauses, with their "meanings", i.e., with what these expressions speak about.

Whereas the work of discovering such laws concerning particular languages would be a job for the linguist, the general theory itself is some sort of methodological framework. One would expect, therefore, that working out a great part of this framework should be in the competence of logic. Indeed, the study of syntax of the "artificial languages" of logic necessarily differs from the study of syntax of natural languages, but what is standardly named "semantics" (in the Morrisian sense) should be independent of the specific features of particular languages. Yet every specialist in this area is aware of the traditional tension between logical semantics and "linguistic semantics". We do not intend to analyze the reasons of this tension here, but some aspects of it are probably important.

One of the most frequent objections raised by the linguists to the competence of logic in supporting linguistics with a general methodological framework consists in accusing the logicians of a systematic simplification of the real problems of linguistic analysis. On the other hand, it is possible to reproduce some complaints made by logicians, concerning the inability of the linguists to pass from empirical description to a deep analysis of the meanings of language expressions. Here we should like to point out that for the greater part this situation stems from some gross errors in the approach of standard logic to the semantic analysis of natural languages. Briefly, the source of these errors can be called "extensionalism". Extensionalism may acquire the form of a manifest confession (this is the case of, say, Quine) or it is a hidden feature of a logical theory (in this sense even Frege, Church, the modal logicians, Montague, etc., can be branded with it).

We have drawn our main theoretical inspiration from the conception contained in some papers and, primarily, in a hitherto unpublished manuscript by Pavel Tichý (1976). In Part One of the present paper all the basic ideas concerning what is traditionally called "semantics" are those offered by Tichý

(1976). In Chapters I–IV we shall not therefore use such formulations as "according to Tichý" or "after Tichý". In most cases, we consider phrases like "in our opinion" or "to our mind" superfluous because we intend to avoid any statements that would only depend on our subjective intuitions and were not founded on a persuasive argument accessible to objective control.

A moderate form of extensionalism is what might be called "Fregean-Churchian semantics" (cf. Frege 1892, Church 1956). According to Fregean-Churchian semantics, language expressions denote some objects (extensions, i.e., individuals, classes, relations, truth-values) on the one hand and express some intensions (Frege: "Sinn", Church: "sense", "intension", "concept") on the other. Thus individual constants denote individuals and express individual concepts; common nouns denote classes or relations-in-extension and express properties or relations-in-intension; sentences denote truth-values and express propositions.

Taking, e. g., the sentence

(1) Prague is the capital of Czechoslovakia

we can say that according to Fregean-Churchian semantics, (1) denotes truth and expresses the proposition that Prague is the capital of Czechoslovakia; "Prague" denotes Prague (an individual) and expresses perhaps nothing else because "Prague" is a proper name; "the capital of Czechoslovakia.

Without taking intensions into account (i.e., remaining within the sphere of "pure extensionalism"), we are not able to analyze (1) so as to save the obviously empirical character of it. Indeed, if "Prague" as well as "the capital of Czechoslovakia" denotes Prague and if "is" means identity (which is obvious, too), then (1) claims the identity of an object with itself, which is no empirical claim, of course. The analysis of an analogous example led Frege to the idea of establishing the realm of intensions in addition to the realm of extensions, but, unfortunately, his approach is not a solution to the problem either. (By the way, this problem is a general one concerning the semantic analysis of any empirical sentence.) Frege's notion of intension being expressed (in contrast to extension being denoted) is in so far indetermined as it cannot play any essential role in the semantic analysis of language expressions. Another weak point of extensionalism consists in that one of its consequences is fully unacceptable if we want (which is certainly the case) to distinguish between understanding a sentence and knowing its truth-value. It is perfectly clear that a vast majority of sentences we are able to understand is such that we do not know their truth-value. Yet, if we accept a quite intuitive principle according to which we cannot understand any expression unless we understand every component of it and if "to understand the expression A" means "to know what the expression A is about" (which also appears to be an intuitive principle), we come to the conclusion that for an extensionalist it is impossible to understand any sentence without knowing its truth-value (because a sentence is, according to him, about a truth value) and that a consistent extensionalist could not understand, e.g., the sentence

(2) The rector of Charles' University is a smoker

without knowing who the rector of Charles' University actually is and without knowing which are the members of the "class of smokers".

Thus it is necessary to refuse any form of extensionalism and to establish the connection between expressions and intensions as the fundamental semantic relation performing the role of the denoting (or: naming) relation. In consequence, (2) names not a truth-value but a proposition, "the rector of Charles' University" names not a definite person but the rector of Charles' University (which is an intension, too), "smoker" names not a class but a property. The proposition named by (2) informs us about a connection between the office of the rector of Charles' University and the property of being a smoker. The actual person occupying the office comes into play, not during the process of understanding (2), but during the process of verifying (2). Similarly, "the capital of Czechoslovakia" names an individual concept (analogous to the concept of the rector of Charles' University). (1) as well as (2) ceases to get a trivial and counterintuitive semantic interpretation. Take, e. g., (2). While someone's being or not being an element of a class is wholly dependent of experience (a class is simply determined by the elements it contains), his having or not having an empirical property (such as being a smoker) is empirically testable.

Remark: The word "Prague" is a specific case. Like other proper names, it directly names an individual, i. e., something that is an extension, Nevertheless, such cases are incorporable into the general intensionalist conception if we define extensions as "intensions of zero order".

Chapter II

INTENSIONS

The most important feature of the present semantic theory is that it is consistently intensionalist. Before proceeding any further, we have to explain in more detail what is meant by "intensions". What follows is an ontology that will enable us to construct exact definitions of the basic concepts of the theory.

The objects a language L is able to speak about are constructible over an "epistemic basis" (EB) with respect to L. To define such a basis, we informally introduce an auxiliary concept: the concept of "intensional basis" (IB) with respect to L. We suppose that the users of L have at their disposal an IB, i.e., a common collection of elementary mutually independent empirical tests and other means enabling them to decide whether the objects (and the strings of objects) established over what is called the universe of discourse (U) do or do not exhibit some empirical traits. Thus nothing is known a priori, i.e. before applying the members of IB to the objects over U, about the distribution of the empirical traits through U. Some distributions are, of course, impossible: it is impossible for an element of U simultaneously to exhibit and not to exhibit some trait. All other distributions are possible (conceivable) and we call the set of them "the logical space" or "the set of possible worlds" (relative to L). One of the possible worlds is the actual one. We would learn which of them it is if we applied all the members of IB to the (strings of) objects over U.

Now we can introduce the concept of minimal epistemic basis (MEB) relative to L. It is a collection of three mutually disjoint non-empty sets:

(i) the set ι (iota) is the universe of discourse relative to L and its members are individuals;

(ii) the set o (omicron) is the set of truth-values and its members are T (truth) and E (falsity);

(iii) the set μ (mu; in Tichý's manuscript ω, omega) its the set of possible worlds relative to L, and its members are all the possible distributions of empirical traits among the objects over ι, i.e. all the possible series of members of o as answers to applying the members of IB to the objects over U. An EB is any such collection of mutually disjoint non-empty sets as contains an MEB as its subcollection.

Let B be an EB. We define a type over B as follows:

(i) Every member of B is a type over B. (Thus the types over an MEB are ι, o, μ.)

(ii) Where α, $\beta_1, \ldots, \beta_n$ are types over B, $\alpha(\beta_1, \ldots, \beta_n)$ is a type over B; we conceive of $\alpha(\beta_1, \ldots, \beta_n)$ as the collection of all the functions that associate every member of $\beta_1 \times \ldots \times \beta_n$ with at most one member of α. Thus – if B is an MEB – $o(\iota)$, $o(\iota)(\mu)$, $o(\iota,o(\mu))(\mu)$, $o(o(\iota))$, etc., are types over B; $o(\iota)$ is the collection of all the functions from ι into o (including partial functions), $o(\iota)(\mu)$ is the collection of all the functions from μ into $o(\iota)$, $o(\iota,o(\mu))(\mu)$ is the collection of all the functions from μ into the set of all the functions from $\iota \times o(\mu)$ into o, $o(o(\iota))$ is the set of all the functions from $o(\iota)$ into o, etc.

(iii) There are no types over B except those definable by (i) and (ii).

It follows from the definition of types over B that the hierarchy of these types is infinite for any B.

Wherever we suppose an EB to be fixed, we can omit "over B" when speaking of types.

Let α be a type. Any member of α will be called an α-object.

Thus $o(\iota)$-objects are classes of individuals because there is no difference between a class of members of ι and the characteristic function of this class. (Usually, we take into account only those members of $o(\iota)$ that are total functions.) Similarly, $o(\iota, \iota)$-objects are binary relations-in-extension of individuals, $o(\iota,o(\iota))$-objects are binary relations-in-extension between individuals and classes of individuals, etc.

This modification of Church's simple theory of types (Church 1940), together with the above intuitions concerning IB and with the concept of MEB, enables us to give an exact definition of intensions, so that it may satisfy the current instuitions connected with the term "intension":

(i) Let α be a type such that there is no β such that $\alpha = \beta(\mu)$. In this case α-objects are intensions of 0-th order or extensions.

(ii) Let α-objects be intensions of k-th order. Then $\alpha(\mu)$-objects are intensions of $(k + 1)$-th order.

Thus whereas $o(\iota)$-objects are extensions (classes of individuals), $o(\iota)(\mu)$-objects are intensions of the 1-st order. Or: the truth-values are o-objects, i. e., extensions, whereas $o(\mu)$-objects are intensions of the 1-st order and $o(\mu)(\mu)$-objects are intensions of the 2-nd order.

Later on (in Ch. IV.) we shall be in need of a concise notation for what will be termed linguistic constructions. For this reason we shall define here the way of writing some kinds of type:

Let α be a type. (i) $\alpha^0 = \alpha$;
(ii) $\alpha^{k+1} = \alpha^k(\mu)$.

Examples: $o(\iota)^0 = o(\iota)$, $o(\iota)^1 = o(\iota)(\mu) = o(\iota)(\mu)^0$, $o(\iota)\mu)((\mu) = o(\iota)(\mu)^1 =$ $= o(\iota)^2$.

Notice that the intensions of k-th order where $k > 0$ are functions from μ. This is exactly what is desirable from the viewpoint of our intuitions, for we wish to conceive of intensions (in accordance with, e. g., Frege) as something that makes it possible to identify extensions. Take, e. g., properties of individuals as an instance of intensions. What is a property, say, of being black, of being a smoker, etc.? Obviously, we know such a property not because we know all the objects which are black, which are smokers, etc.; but we can decide for any state of affairs (or: in any possible world!) whether a given object is black, whether it is a smoker, etc. (The ability to make such a decision may only be present theoretically, in principle, and need not be technically realizable.) In other words, if we were given some possible world, we could generate a class of individuals that satisfy the criterion given by our knowledge of the property in question. Thus a property of individuals can be identified with a function that associates any possible world with a class of individuals. In a similar way it can be shown that a proposition is something that enables us to identify the truth-value of a certain condition, and consequently, propositions are $o(\mu)$-objects. The capital of Czechoslovakia is another example of an intension: it is an individual concept and therefore an $\iota(\mu)$-object: this function associates some possible worlds (including the actual one) with Prague, some other possible worlds with, say, Brno, etc.

Have a table of some important intensions of the 1-st order. We suppose that $\alpha, \beta_1, \ldots, \beta_n$ are any types.

Type	Objects	Corresponding extensions, i. e. extensions that are values of the intensions in a given possible world
$\iota(\mu)$	individual concepts	individuals
$o(\mu)$	propositions	truth-values
$o(\iota)(\mu)$	properties of individuals	classes of individuals
$o(\underbrace{\iota, \ldots, \iota}_{n})(\mu)$	n-ary relations-in-intension between individuals	n-ary relations-in-extension between individuals
$o(\alpha)(\mu)$	properties of α-objects	classes of α-objects
$o(\beta_1, \ldots, \beta_n)(\mu)$	relations-in-intension between $\beta_1, \ldots, \beta_n$-objects	relations-in-extension between $\beta_1, \ldots, \beta_n$-objects

As an example of an intension the type of which is presented in the last "type" row of the above table, we can adduce believing: it is obviously an $o(\iota, o(\mu))(\mu)$-object, i. e. a relation-in-intension between individuals and propositions. When applying this function to the actual world, we get the members of the corresponding relation-in-extension, i. e. the ordered pairs the first members of which are individuals and the second members of which are the propositions such that the first member believes the second member to be true.

Defined in this way, the intensions cease to be "obscure entities" (Quine) and become well-defined objects (provided functions are considered to be well-defined objects).'

A language is determined by EB, vocabulary and grammar. This means that once a language is given, no question concerning membership in one of the members of EB can arise. For example, for every user of a language, the individuals are given a priori. Furthermore, the universe of discourse is the same for all the possible worlds, so that we do not suppose there is any possible world, say W, with respect to which the universe of discourse contains a certain individual or individuals that with respect to another possible world, say W', are not contained in the universe of discourse. Thus if Pegasus exists in some possible world W and does not exist, e.g., in the actual world, it only means that – according to the present conception – some individual, say x, does in W, and does not in the actual world, match the characteristic of Pegasus. (It should be clear by now that Pegasus is an $\iota(\mu)$-object and not an ι-object.)

Chapter III

CONSTRUCTIONS

The vocabulary and grammar of a language L are – *qua* components of L – connected with EB. Now another important feature of the present theory consists in that this connection is conceived of not as a direct connection but as a connection mediated by what is to be called constructions. In the following paragraphs we intend to explain what is meant by constructions in our theory.

We shall begin with some examples. Let our MEB be extended so as to contain the set ν of natural numbers (0, 1, 2, ...). Whereas 3, 5 are ν-objects, + (plus), : (divided by) are $\nu(\nu,\nu)$-objects: they are functions associating any pair of natural numbers with at most one natural number; + associates, e. g., the pair ⟨3,5⟩ with 8, : associates ⟨8,2⟩ with 4, etc. (Notice that : associates, e. g., ⟨8,3⟩ with no natural number and that the same can be said about any pair the second member of which is 0.) Knowing that 0, 1, 2, 3, ... are ν-objects and +, :, ... are $\nu(\nu, \nu)$-objects, we put the following question:

What sort of entity is $3 + 5$?

Clearly, it is neither a ν-object nor a $\nu(\nu,\nu)$-object. We claim, therefore, that such entities as $3 + 5$, $7 : 2$, $7 : (3 + 4)$, etc., are no objects at all. They serve to construct objects if it is possible. Hence we call them constructions. A construction always constructs at most one object. Generally speaking, this object is different from the construction itself: 3 + 5 is something different from 8, because the former is compound and contains occurences of 3, 5, and +, whereas 8 is a simple object. Nevertheless, we can conceive of objects as an "extreme" case of constructions, namely as self-constructing constructions. (It is important to remember that when speaking about constructions we do not consider them to be inscriptions. Thus saying that 3 + 5 is a construction, we do not mean by this that the inscription "3 + 5" is a construction. The constructions themselves are independent of any language.)

Before we define exactly the notion of construction, we shall generalize our intuitions. For the sake of this generalization, the following question is of some interest to us:

Is $3 + x$ a construction?

By x we mean an abstract representative of ν-objects; in other words, any ν-object can be thought of as standing in place of it. Let such abstract representatives of (α-)objects be called (α-)variable (where α is a type). Now, $3 + x$ constructs no object, as it stands, but with, say, 5 in place of x, it constructs 8, with 7 it constructs 10, etc. Such entities that differ from constructions (as characterized up to now) only in that they contain at least one occurrence of a variable (which means that what they construct depends on what object is in place of such a variable), will be called open constructions. We can suppose that there are denumerably infinite sets of α-variables for any type α at our disposal. Any (total) function that associates every α-variable with exactly one α-object will be called valuation. The value of a valuation v at the variable a will be called the v-instance of a. The v-instance of a may be said to be v-constructed by a. Let A be the object that is constructed by an open construction A if all the occurrences of those variables on which it depends what object (if any) will be constructed by A are replaced by v-instances of them. Then we say that A v-constructs A. If A does not construct any object after such a replacement, we say that A is v-improper.

Examples: Let A be $3 + (8{:}x)$. Let v_1 replace x by 4. Then A v_1-constructs 5. Let v_2 replace x by 0. Then A is v_2-improper.

Let A′ be $x - x$. Then A′ v-constructs 0 for every valuation v.

Let A″ be $\frac{x + y}{5 - x}$. A″ is v-improper, e. g., for any valuation v that replaces x by 5. Let v_1 replace x by 4 and y by 3. Then A″ v_1-constructs 7.

The above basic notions need not be exemplified by such arithmetical examples only, of course. Later on, we shall return to MEB or to some extensions of MEB. Now we can define the general notion of construction.

A terminological note: Variables and objects will be called atoms here.

The notions of type and construction have to be relativized to an EB. The explicit relativization will be omitted in the present definition.

Let α, $\beta_1, \ldots, \beta_n$ be types.

(i) Every α-atom is an α-construction. If the atom b is an α-object B, then b v-constructs B for any valuation v. If the atom b is an α-variable b, then b v-constructs the v-instance of b.

(ii) Let $F_0, F_1, \ldots, F_n$, $n \geq 1$, be $\alpha(\beta_1$-$, \ldots, \beta_n)$-, β_1-$, \ldots, \beta_n$-constructions, respectively. Then $F_0(F_1, \ldots, F_n)$ is an α-construction called the application of F_0 to $F_1, \ldots, F_n$. If at least one of $F_0, F_1, \ldots, F_n$ is v-improper, then $F_0(F_1, \ldots, F_n)$ is v-improper. Otherwise let $F_0, F_1, \ldots, F_n$ be the objects v-constructed by $F_0, F_1, \ldots, F_n$, respectively. Then $F_0(F_1, \ldots, F_n)$ is v-improper if F_0 is not defined at the n-tuple $\langle F_1, \ldots, F_n\rangle$; otherwise $F^0(F_1, \ldots, F_n)$ v-constructs the value of F_0 on the n-tuple $\langle F_1, \ldots, F_n\rangle$.

(iii) Let $x_1, \ldots, x_n$, $n \geqq 1$, be β_1-, $\ldots$, β_n-variables, respectively. Let Y be an α-construction. Finally, let $v(A_1, \ldots, A_n/x_1, \ldots, x_n)$ be a valuation that differs from the valuation v at most in that it replaces $x_1, \ldots, x_n$ by the β_1-, $\ldots$, β_n-objects $A_1, \ldots, A_n$, respectively. Then $\lambda x_1 \ldots x_n$ (Y) is an $\alpha(\beta_1, \ldots, \beta_n)$-construction called the $x_1, \ldots, x_n$-abstraction of Y. This kind of construction v-constructs the function F defined as follows: Let $A_1, \ldots, A_n$ be β_1-, $\ldots$, β_n-objects, respectively. If Y is $v(A_1, \ldots, A_n/x_1, \ldots, x_n)$-improper, then F is not defined at $\langle A_1, \ldots, A_n\rangle$. Otherwise the value of F is the object that is $v(A_1, \ldots, A_n/x_1, \ldots, x_n)$-constructed by Y.

(iv) Nothing else is an α-construction.

Examples: Let MEB contain $U = \{A, B\}$.

We define the following functions:

G_1 is an $o(\iota,\iota)$-object defined as flollows:

	G_1	
A,A	T	(truth)
A,B	F	(falsity)
B,A	—	(not defined)
B,B	T	

G_2 is an $\iota(\iota,\iota)$-object defined as follows:

	G_2
A,A	A
A,B	A
B,A	B
B,B	—

Let x,y be ι-variables and let x v_1- and v_2-construct A and v_3- and v_4-construct B, y v_1- and v_3-construct A and v_2- and v_4-construct B.

Then (a) $G_2(x,A)$ is an application of G_2 to x,A. Similarly, (b) $G_2(x,y)$ is an application of G_2 to y, x. We can see that (a) v_1- and v_2-constructs A, v_3- and v_4-constructs B. (b) v_1- and v_3-constructs A, v_2-constructs B, and is v_4-improper.

Let us have the following construction:

(c) $G_1(G_2(x,A), G_2(y,x))$.

Since G_1 is an $o(\iota,\iota)$-atom, and therefore an $o(\iota,\iota)$-construction, and $G_2(x,A)$, as well as $G_2(y, x)$, are ι-constructions, (c) is — qua an application of G_1 to (a) and (b) — an o-construction. Clearly, (c) v_1-constructs T, v_2-constructs F and is v_3- and v_4-improper.

Let us have the following construction:

(d) $\lambda x(G_1(G_2(x,A), G_2(y,x)))$.

(d) is an x-abstraction of (c) and is therefore an $o(\iota)$-construction. Since (c) $v_1(A/x)$-constructs T and is $v_1(B/x)$-improper, (d) v_1-constructs the following function, say, H_1:

	H_1
A	T
B	—

Similarly, it can be shown that (d) v_2-constructs the function, say, H_2:

	H_2
A	F
B	—

etc. Finally, observe

(e) $\lambda xy(G_1(G_2(x,A), G_2(y,x)))$.

(e) is an $o(\iota,\iota)$-construction (it is the x,y-abstraction of (c)). We can see that for any v, (e) v-constructs the following function, say, H_3:

	H_3
A,A	T
A,B	F
B,A	—
B,B	—

From our definitions it follows that
(1) what an $x_1, \ldots, x_n$-abstraction v-constructs never depends on the variables $x_1, \ldots, x_n$;
(2) $x_1, \ldots, x_n$-abstractions are never v-improper.

Ad (1): Thus (e) is not an open (but a closed) construction.

Having defined the notion of construction, we can say the following:

Let L be a language. The expressions of L express some kinds of construction (over an EB) and denote what is constructed by these constructions. (In case the construction in question is an atom, what is denoted is identical with what is expressed.) Thus we can say that expressions of L express α-constructions where α-objects are intensions of the k-th order, $k \geq 0$.

To analyze semantically (or: to offer a semantic analysis of) an expression A (of L) means to find the construction expressed by A.

Chapter IV

LINGUISTIC CONSTRUCTIONS

The general features of our semantic theory have been set forth in the preceding two chapters. Any non-indexical (for this term see Chapter VI) expression of a language L (except for the "syncategorematic" expressions) names (denotes) an intension (of the k-th order, $k \geq 0$) and expresses a construction that constructs the intension in question. Now, it is probable that not every thinkable construction over an EB is expressible by the expressions of a language L. We shall define the class of what (after Tichý) we should like to call "linguistic constructions". The expressions of any language L will express just the members of this class.

Before defining linguistic constructions, let us introduce a sort of shorthand for the notation of some constructions:

Let w be a μ-variable. If A is an α^m-construction where α is a type of an extension, we write

(i) $A(w)^0 = A$,

(ii) $A(w)^{k+1} = A(w)^k(w)$,

where $k < m$.

Example: Let K be an $o(\iota)(\mu)(\mu)$-object, i. e., an $o(\iota)^2$-object. Then

$(Kw)^0 = K$, $K(w)^1 = K(w)$, $K(w)^2 = K(w)(w)$.

$K(w)^1$ is an $o(\iota)(\mu)$-construction, $K(w)^2$ is an $o(\iota)$-construction.

Definition of linguistic constructions:

(i) Let w be the alphabetically first μ-variable. Every atom different from w is a linguistic construction.

(ii) Let $F_0, F_1, \ldots, F_n$, $n \geq 1$, be $\alpha(\beta_1, \ldots, \beta_n)^{i_0}$-, β^{i_1}-, $\ldots$, β^{i_n}-linguistic constructions, respectively, where i_j for $j = 0, 1, \ldots, n$, is 0 or 1; if $i_r = 1$, then let F_j not be a variable. Then

$\lambda w(F_0(w)i_0(F_1(w)^{i_1}, \ldots, F_n(w)^{i_n}))$

is a linguistic construction called $i_0, \ldots, i_n$-compound of $F_0, F_1, \ldots, F_n$.

(iii) Let $x_1, \ldots, x_m$, $m \geq 1$, be distinct variables different from w. Then if Y is a linguistic construction,

$\lambda x_1 \ldots x_m(Y)$

is a linguistic construction called a O-abstract of Y.

(iv) Let $x_1, \ldots, x_m$ be as in (iii) and let Y be an α^1-linguistic construction. Then

$\lambda w(\lambda x_1 \ldots x_m(Y(w)))$

is a linguistic construction called a 1-abstract of Y.

(v) The set of linguistic constructions is the minimal set containing constructions satisfying (i) — (iv).

Examples: Let O be a relation-in-intension of being older (than), i.e., an A, B be two members of U, i.e., individuals (ι-objects). Let C be the individual concept of the (present) capital of Czechoslovakia (i.e., an $\iota(\mu)$-object). Then

(a) $\lambda w((wO)(A,B))$ $(= \lambda w(O(w)^1(A(w)^0, B(w)^0)))$ is a $1-0-0$-compound of O, A, B. We can see that (a) is an $o(\mu)$-construction constructing the proposition that A is older than B.

(b) $\lambda w(O(w)(C(w),B))$

is a $1-1-0$-compound of O, C, B. Again, (b) constructs a proposition: the capital of Czechoslovakia is older than B.

(c) $\lambda x(\lambda w(O(w)(x,A)))$,

where x is an ι-variable, is a 0-abstract of

(c') $\lambda w(O(w)(x,A))$,

because (c'), being a $1-0-0-$compound of O, x, A, is a linguistic construction.

(c) is an $o(\mu)(\iota)$-construction (cf. the definition of $x_1, \ldots, x_n$-abstraction in Chapter III). It constructs a function that associates every individual with the proposition that this individual is older than A.

(d) $\lambda w(\lambda x(\lambda w(O(w)(x,A))(w)))$

is a 1-abstract of

$\lambda w(O(w)(\mathrm{x},A))$ $(= \mathrm{c}'))$;
clearly, it is an $o(\iota)(\mu)$-construction constructing the property to be older than A.

(e) $\lambda w(\lambda xy(\lambda w(O(w)(x,y)))(w))$
is a 1-abstract of
$\lambda w(O(w)(x,y))$.
(e) constructs the relation-in-intension of being older (than). We can see that (e) is equivalent to O (in that it constructs the same object as O).

Reducts:

It can be proved that the construction
$\lambda w(\mathrm{Z})(w)$,
where Z is a construction, is equivalent to Z. Therefore, we can — where possible and necessary — replace the linguistic constructions by their reducts. The reduct of a 1-abstract
$\lambda w(\lambda x_1 \ldots x_{\mathrm{n}}(\mathrm{Y}(w)))$,
where Y is of the form $\lambda w(\mathrm{Z})$, is the result of replacing $\mathrm{Y}(w)$ by Z. The reduct of a linguistic construction A is the result of performing all such replacements within A.

Thus the reduct of (d) will be
(d′) $\lambda w(\lambda \mathrm{x}(O(w)(\mathrm{x},\mathrm{A})))$
and the reduct of (e) will be
(e′) $\lambda w(\lambda xy(O(w)(x,y)))$.

As has been said before, a language L is determined by EB, vocabulary (lexicon) and grammar. Grammar may be conceived of as a set of rules that connect expressions of L with the constructions expressed by them, i.e., with their analyses. The vocabulary contains both words (phrases) expressing (= denoting) atoms, indexical words (see Chapter VI) and syncategorematic words, the role of which is purely syntactic (they neither express nor denote anything). The set of rules contains "compound-rules", connecting expressions with compounds, and "abstract-rules", connecting expressions with abstracts.

Examples: Let F be an $o(\iota)(\mu)$-object and G an $\iota(\mu)$-object. Let **F** and **G** be words (phrases) of English (Czech) denoting F and G, respectively. Then the following simple compound-rule can be formulated:

Any construction of the form
(f) $\lambda w(F(w)(G(w)))$
is expressed as follows:

English:	**G** is an F,	if **F** is a common noun group,
	G is F,	if **F** is an adjective,
	G F (3rd pers. sg.),	if **F** is a verb;
Czech:	**G** je F,	if **F** is a common noun group or an adjective
	G F (3rd pers. sg.),	if **F** is a verb.

Thus let F be the property of being a smoker, let G be the individual concept of the (present) director of the Škoda works. Let **F** be in English the word "smoker", in Czech the word "**kuřák**", and **G** in English "the director of the Škoda works", in Czech "ředitel Škodovky".

Then our rule works as follows: The construction (f) is expressed in English:
The director of Škoda works is a smoker.
In Czech:

Ředitel Škodovky je kuřák.

Another rule might be formulated as follows:

Let $\lambda w(\mathrm{P})$ and $\lambda w(\mathrm{Q})$ be linguistic $o(\mu)$-constructions, let C be an $o(o,o)$-object (i.e., a logical connective). Let **P**, **Q** and **C** be the expressions of English (Czech) expressing $\lambda w(\mathrm{P})$, $\lambda w(\mathrm{Q})$, C, respectively. Then for English (Czech) the construction

$\lambda w(C(\mathrm{P},\mathrm{Q}))$ (i.e. the reduct of the $0-1-1$-compound of C, $\lambda w(\mathrm{P})$, $\lambda w(\mathrm{Q})$) is expressed as follows:

PCQ.

Example: Let C be $\wedge$ (conjunction), let $\lambda \mathrm{w}(\mathrm{P})$ be the construction

$\lambda w(\mathrm{O}(w)(\mathrm{A},\mathrm{B}))$

and $\lambda w(\mathrm{Q})$ the construction

$\lambda w(F(w)(B))$

(cf. the preceding examples).

Let **P** be the sentence

A is older than B	(*A je starší než B*),

Q the sentence

B is a smoker	(*B je kuřák*),

and **C** the word

and	(a).

According to our rule, the construction

$\lambda w(\wedge(O(w)(A,B),F(w)(B)))$

is expressed by

A is older than B and B is a smoker

(*A je starší než B a B je kuřák*).

Chapter V

TEMPORAL FACTOR

Now we shall show that an EB must be wider than an MEB to become an adequate base for a linguistic analysis. Let us have the sentence

(1) *Charles was a smoker and he is not a smoker.*

(1) is transformable to

(1′) *Charles was a smoker and Charles is not a smoker.*

Let *Ch* be the individual expressed (= denoted) by "Charles". Let *F* be the property of being a smoker, let "and" and "not" denote $\wedge$ (conjunction) and $\sim$ (negation), respectively. Over an MEB, we cannot but analyze (1′) as follows:

(1″) $\lambda w(\wedge(F(w)(Ch),\sim(F(w)(Ch))))$,

which is a reduct of a linguistic construction. (1″) is, however, a construction constructing contradiction, i.e. a proposition that is false in any possible world.

Asking what this absurdity has been caused by, we can see that understanding (1) as a sentence which denotes a non-contradictory proposition is caused by the time factor, i.e. by the fact that the tenses of the two clauses of (1) are different.

A semantic analysis only based on an MEB cannot take the time factor and, consequently, the tenses into account. Nevertheless, there are numerous possibilities of extending an MEB. One of them consists in including into EB the set of "time moments". Bearing in mind some additional extensions (including "space points", etc.), we define an EB_t as any such collection of mutually disjoint non-empty sets as contains an MEB and the set σ_1 of time moments.

Our intuitions connected with an EB_t are the following: in most cases, where A is an intension of the k-th order, $k \geq 1$, i.e. an α^k-object, its "revised" version A' becomes a function from σ_1 into α^k in an EB_t. This corresponds with the fact that any possible world can be conceived of not as a momentaneous distribution of empirical traits but as a set of such distributions possibly different at different time moments. Thus the property of being a smoker can give — as its value in the possible world W at the time moment t_1 — a class of individuals different from that in the world W at the time moment t_2: the people become smokers and cease to be smokers in the same world (e. g., in the actual one). Thus where F as the property of being a smoker was (over an MEB) an $o(\iota)(\mu)$-object, the revised (time dependent, or: t-) property F_t is an $o(\iota)(\mu)(\sigma_1)$-object (over an EB_t).

Similarly, any $o(\mu)$-construction over an MEB becomes an $o(\mu)(\sigma_1)$-construction over an EB_t. Such an $o(\mu)(\sigma_1)$-construction constructs a "t-proposition".

There is also an alternative approach which seems, however, to be more appropriate than the one offered above. We may conceive of the intensions as $\alpha(\sigma_1)(\mu)$-objects where α is a type. This means that, e. g., the property denoted by the word "smoker" would associate every possible world with the "history" of this property so that a concrete class would be the result not only of applying this property to a possible world, but also of applying the result of this application (i.e., an $o(\iota)(\sigma_1)$-object) to a time moment.

Our following examples presuppose the latter approach has been accepted.

The members of σ_1 can be ordered by the relations $<$ and $\leq$. We can define the $o(o(\sigma_1))(\sigma_1)$-objects *Pret* and *Fut* as follows:

Let s, s' be σ_1-variables. Let A be an $o(\sigma_1)(\mu)$-construction. Using the standard notation of quantifiers and connectives, we can say that the constructions

$\mathit{Pret}(s)(A(w)), \qquad \mathit{Fut}(s)(A(w))$

are equivalent to

$\exists s'(<(s',s) \wedge A(w)(s')), \quad \exists s'(>(s',s) \wedge A(w)(s')),$

respectively.

Thus (1) will be analyzed as follows:

$\lambda w(\lambda s(\wedge(\mathit{Pret}(s)(\lambda s(F(w)(s)(Ch))), \sim(F(w)(s)(Ch))))),$

which can be modified — according to our definition of Pret and the logical laws of λ-conversion — as

$\lambda w(\lambda s(\wedge(\exists s'(\wedge(<(s',s), F(w)(s')(Ch))), \sim(F(w)(s)(Ch))))).$

Immediately we can see that these last constructions construct no contradictory proposition (t-proposition).

There are some problems with the time factor that are not directly connected with tenses. But none of them seems to be unsolvable within our theory.

(What has been said about time up to now has been inspired by some remarks made by Tichý in his letters.)

As for the set of space points (σ_2), we need it in order to be able to analyze expressions containing local adverbs and/or local prepositions. Let $EB_{t,l}$ contain an MEB, σ_1 and σ_2. It seems that if a verb denotes an $o(\beta_1, \ldots, \beta_m)(\mu)$-object over an MEB, then it denotes an $o(\beta_1, \ldots, \beta_m)(\sigma_1)(\mu)$-object over an EB_t and it can (not necessarily) denote an $o(\beta_1, \ldots, \beta_m, \sigma_2)(\sigma_1)(\mu)$- or an $o(\beta_1, \ldots, \beta_m, o(\sigma_2))(\sigma_1)(\mu)$-object over an $EB_{t,l}$.

Thus let P be Prague (type: ι), IN be an $o(\sigma_2)(\iota)(\sigma_1)(\mu)$-object (denoted by the preposition "in" in English and "v(e)" in Czech) and S an $o(\iota)(\sigma_1)(\mu)$-object (the *t*-property of sleeping); further let S' be an $o(\iota, o(\sigma_2))(\sigma_1)(\mu)$-object (the t-relation-in-intension between a sleeping individual and the place where this individual sleeps). To use S is sufficient when our task is to analyze, e. g., the sentence

(2) *Charles is sleeping.*

whereas S′ must be used if we wish to analyze the sentence

(3) *Charles sleeps in Prague.*

The analyses of (2) and (3):

(2′) $\lambda w(\lambda s(\mathrm{S}(w)(s)(Ch)))$;

(3′) $\lambda w(\lambda s(\mathrm{S}'(w)(s)(Ch, IN(w)(s)(P))))$.

Clearly, our definitions of linguistic constructions must be modified so as to become definitions of *t*-linguistic or *t-l*-linguistic constructions.

Similarly, it is thinkable to extend an MEB or an EB_t or an $EB_{t,l}$ by adding the set, say, ν of natural numbers to the sets o, ι, μ, σ_1, σ_2. The collection $EB_{t,l,n}$ obtained in this way would be appropriate in case we should like to deal with such part of a natural language as contains means for expressing mathematical constructions.

Chapter VI

INDEXICALITY

The investigation of indexical expressions (indexicals) is often considered to be the principal task of pragmatics. (Cf. Montague 1968.) Within our theory, indexicals represent the sphere of external pragmatics, which is to be clearly distinguished from internal pragmatics (see Ch. VII) on the one hand and semantics on the other. Have the following sentence:

(O) *I am hungry now.*

As an expression of a natural language the sentence (O) is not fully analyzable within semantics. Some of its components make it impossible for us to determine which proposition (O) denotes. We should rather say that there is a great number of propositions offering themselves as candidates for the object which (O) is about. The best-suited representative of such a set of propositions seems to be an open construction, i.e., a construction containing free variables (in our case, one of them standing for "I" and another for "now").

We can see that some expressions of a natural language are "dead" in that they do not name any definite object over an EB. Some of them remain to play this role for ever (e. g., syncategorematic expressions). Yet some of them "come to life" when uttered in a concrete situation. Studying utterances instead of sentences, i.e. events instead of abstractions, we study the spatio-temporal circumstances that — together with the sentence itself — determine which proposition is named by uttering the sentence in the very act of communication. Whereas semantics is concerned with propositions, external pragmatics is concerned with associating situations with propositions. No semantic analysis can, of course, determine the object named by "I" (or by "now") in (O). Such expressions are ostensive in that their communicative role is bound to concrete circumstances of the act of uttering an expresion the components of which they are.

It is to be emphasized that the above explanation is not meant to justify any kind of fusion of semantics and pragmatics. On the contrary, just since the role of indexicals clearly differs from that performed by the other expressions, we must carefully distinguish between the semantic analysis on the one hand and the theory of "situation-proposition"-relations (or the theory of indexicals) on the other. In some respects, semantics has priority over pragmatics: What a sentence uttered *hic et nunc* is about (i.e. which proposition it denotes) cannot be established without our knowing the semantics of the given language; but (a) we can learn what a sentence just uttered is about without knowing the circumstances of the act of utterance in case the sentence in question does not contain any indexicals, and (b) even if the sentence does contain some indexicals, we can, on the basis of the pure semantics, establish at least the scheme of possible $o(\mu)$-constructions expressed by this clause.

Let us return to what has been said at the beginning of the present chapter. Take the following sentence:

(1) *I am hungry.*

In the vocabulary of English satisfying the principles of our theory we can find that "to be hungry" names an $o(\iota)(\sigma_1)(\mu)$-atom (say, H) and "I" is an indexical word that can name an individual. As an indexical word, however, it denotes no concrete individual.

Let A,B, C, ... be members of the universe in question. Using (1) in particular situations, we can express by it the following constructions:

(1_1) $\lambda w(\lambda s(H(w)(s)(A)))$;

(1_2) $\lambda w(\lambda s(H(w)(s)(B)))$;

(1_3) $\lambda w(\lambda s(H(w)(s)(C)))$;

. .

. .

. .

Thus using (1) in situations S_1, S_2, S_3, ..., we simultaneously use a function that associates "I" with various individuals dependently on S_1, S_2, S_3, ... This function, say F_I, is not what can be called a semantic function: its domain is the set of situations, and, therefore, such a function is a pragmatic function. Associating "I" with the individuals A, B, C, ..., F_I thereby associates the sentence (1) with the constructions (1_1), (1_2), (1_3), ..., and, consequently, with different propositions.

Now, have the open construction

(1)′ $\lambda w(\lambda s(\mathrm{H}(w)(s)(x)))$,

where x is an ι-variable. If x v-constructs A, v′-constructs B, v″-constructs C, ..., then the v-instance of (1)′ is (1_1), the v′-instance of (1)′ is (1_2), the v″-instance of (1)′ is (1_3), ... Therefore, we could conceive of F_I as a principle that determines — dependently on situations — which valuation should be selected as regards x in (1)′ if (1)′ is associated with (1). Similarly, let F_{HE} be another function, such as associates the situations with individuals and is used whenever we use the English word "he" (or, the Czech word "on", etc.). Then F_{HE} will play the role of a principle that determines which valuation should be selected as regards x in (1)′ if (1)′ is associated with (2):

(2) *He is hungry.*

Our example can be generalized for all the cases in which the sentence in question contains some indexical words (as "we", "now", "this", "here", etc.). Such sentences (let us call them" indexical sentences" or simply "i-sentences") are able to express a construction whenever the situation determines what entities are meant by the various indexical words. These entities may be of various types; in case of "I", "he", "it", etc., they are ι-objects, in case of "we", "they", etc., they are $o(\iota)$-objects, in case of "now", etc., they are σ_1 or $o(\sigma_1)$-objects, etc., etc.

Ours is the following question: How to analyze an i-sentence?

Any adequate answer to our question must take into account that an i-sentence does not determine a proposition unless there is given a concrete situation of use. Thus without pragmatic determination no analysis of an i-sentence is possible, which amounts to saying that without pragmatics it is impossible for us to learn what is meant by such an i-sentence.

Yet, something is given if an "isolated" i-sentence is to be dealt with: where $i_1, \ldots, i_n$ are the indexical expressions contained in the sentence **A**, we can associate the sentence **A** with such an open construction A such as contains variables $x_{i_1}, \ldots, x_{i_n}$ in place of those atoms which would respectively be named by $i_1, \ldots, i_n$ if $i_1, \ldots, i_n$ were not indexical. Besides, we can put the pragmatic functions $F_{i_1}, \ldots, F_{i_n}$ within brackets behind the variables. Thus the sentence

(3) *He is hungry now.*

would be associated with the following construction:

(3′) $$\lambda w(\mathrm{H}(w)(\mathrm{s})[F_{NOW}](\mathrm{x}[F_{HE}]))$$

where s is a σ_1-variable and x an ι-variable.

A situation may be conceived of as an n-tuple consisting of a time moment t, the place, the speaker m, the hearer h, the objects that are spoken about, etc. Given a situation S containing the time moment T and the individual K that is being spoken about, we have

$$F_{NOW}(\mathrm{S}) = T, \qquad F_{HE}(\mathrm{S}) = K$$

and the relevant "pragmatic instance" of (3′) is

(3″) $$\lambda w(\mathrm{H}(w)(T)(K)).$$

Now we can say that (3) expresses (3″) in the situation S. If no situation is given, (3) expresses no construction, but it is associated with (3′). This reflects the fact that (3) qua a "dead sentence" names no specified proposition.

If we want to analyze, e. g., English sentences, we can choose one of, at least, two approaches: we can either refuse to analyze any i-sentence and demand that all indexical "parameters" be substituted for by non-indexical expressions, or we can associate such an i-sentence with an open construction in the above way. In the latter case, however, we do not say that the sentence in question expresses the respective open construction; it does not express any construction at all.

Remark: The present conception basically differs from that advocated by Montague, D. Scott, etc., in that we do not mix together possible worlds and situations; semantics (possible worlds!) and pragmatics (situations!) should be kept apart — the concern of the former is different from that of the latter.

Chapter VII

THE SEMANTIC TRIANGLE AND PRAGMATICS

This chapter is meant to illustrate the relation between semantics as presented above and pragmatics. The most suitable starting point for graphical illustration seems to be the following triangle:

$$\begin{tikzcd}
& \text{word or phrase} \arrow[dl, "\text{denotes (names)}"'] \arrow[dr, "\text{expresses}"] & \\
\text{function (concept)} & & \text{construction} \arrow[ll, "\text{constructs (yields)}"']
\end{tikzcd}$$

A word or a phrase expresses a certain construction, which in turn constructs (or, yields) the respective function (in our case, a concept). This function is what the given word or phrase denotes (or, names). The denoting (naming) relation between words (phrases) and functions is established through the mediation of constructions. Functions (concepts) are intensions; hence words and phrases denote intensions. Extensions, i.e. individuals, classes of individuals, relations between individuals, relations between classes of individuals, etc., are in our cases the values of the functions in questions. There is only one case in which a function is both an intension and an extension: the nullary function. The value of a nullary function is the function itself. Words and phrases denoting nullary functions are called proper names. Proper names are the only cases in which language expressions denote extensions. Thus we know what proper names speak about if we know their extensions, i. e., if we know precisely which individual, class of individuals, etc., of our universe of discourse is denoted by a given proper name. As has been shown before, this is not the case with other words and phrases denoting non-nullary functions. We know what such an expression speaks about because we know, or can construct, the respective function (concept), which may be conceived of as instructions of how to look for the values of this function in different possible worlds. Whether we find the value in a given (e.g., the actual) world or not is by no means the condition of understanding the given expression.

Similar to words and phrases, sentences or clauses (further only sentences) also express constructions and denote (name) functions. The construction expressed by a sentence contains all the constructions expressed by the separate words and phrases in the given sentence as well as their mutual arrangement. Such a construction yields a function which is, of course, not independent of the functions denoted by the separate words and phrases of the sentence, but which — at the same time — is not a mere collection of the separate functions either. The construction expressed by a sentence yields a function from possible worlds into truth-values, i. e. a function that assigns a truth-value to any possible world in which it is defined. This kind of function is called a proposition. In contradistinction to Frege (cf. Frege 1892), a sentence expresses a construction and denotes a proposition:

$$\begin{tikzcd}
& \text{sentence} \arrow[dl, "\text{denotes (names)}"'] \arrow[dr, "\text{expresses}"] & \\
\text{proposition} & & \text{construction} \arrow[ll, "\text{constructs (yields)}"']
\end{tikzcd}$$

What has happened to Frege's truth-values? They simply represent the values of the proposition in the possible worlds, i. e., the extensions, which are not directly denoted by the sentence.

To understand a sentence means to know what it denotes; it means to know the respective proposition, the function from possible worlds into truth-values. To know this function is tantamount to knowing the way of how to look for the truth-value of the sentence in any possible (i.e. also the actual) world. We know what the sentence

The President of Czechoslovakia opened the Championship

denotes and we understand it, not because we know whether it is true or false, but because we know the functions (concepts) denoted by "the President of Czechoslovakia", "opened", "the Championship", and we also know the proposition, i.e., how to verify the sentence in any possible world. If it is true in the actual world, we say that it is a fact; but fact or no fact, it has no bearing whatsoever on our understanding the sentence.

In the same way as we understand "Pegasus", "mermaid" and other mythological and fairy-tale expressions without finding the values of their respective functions (i.e. the occupants of their offices) in the actual world, we also understand sentences without finding the truth-values of "their" propositions in the actual world. In addition to that, by using such sentences in a way suggesting that "their" propositions are true in some possible world, we are in fact building up or "discovering" the subset of possible worlds in which these propositions are true. This is not only the case of fairy tales, the same holds good for scientific theories. We understand them because we know what they denote, we know the concepts and propositions. Furthermore we are entitled to assume that the propositions are true in some possible worlds. Hence to verify a theory means to find out whether the actual world is among them, i.e., to find out whether all the values of the propositions in the actual world are "truth".

What is meant by "using sentences in a way suggesting that 'their' propositions are true"? Up to now we have been concerned with a sentence from the semantic point of view and have paid little attention to the language user and the pragmatic aspects of a sentence. A detailed analysis of any sentence will reveal that apart from expressing a construction and denoting a proposition, it conveys some further information closely connected with the language user. By formal, explicit means, the sentence shows whether the language user considers the proposition denoted by the sentence to be true in some (usually the actual) world (assertion) or whether he wants to know what the truth-value is (yes-no question), or wants the proposition to be true in the actual world (command), or wishes it to be true (desideration), etc. A sentence demonstrates these attitudes of the language user apart from what it expresses and denotes.

As has been shown in the previous paper on the ordered-triple theory, the above mentioned attitudes (broad modalities) are not the only ones the language user is obliged to take and the sentence has to demonstrate. Another kind of such attitudes is represented by the subjective estimates of probability (probability of the proposition being true) and yet another by the distribution of various degrees of communicative dynamism over the elements of a sentence (phenomena coming under the heading of Functional Sentence Perspective, see Firbas 1964, 1966). Each attitude of one kind is regarded as a value at the coordinate representing the respective kind of attitude. At present we take the following three coordinates into account:

A^1 with the values "assertion", "yes-no question", "wh- question", "command", "wish",

A^2 with the values "100 %", "90 %", "80 %", "70 %", 60 %", "50 %",

A^3 with the values "1", "2", "3", . . ., "n" (representing the various patterns of sentence perspetive according to possible distributions of theme and rheme).

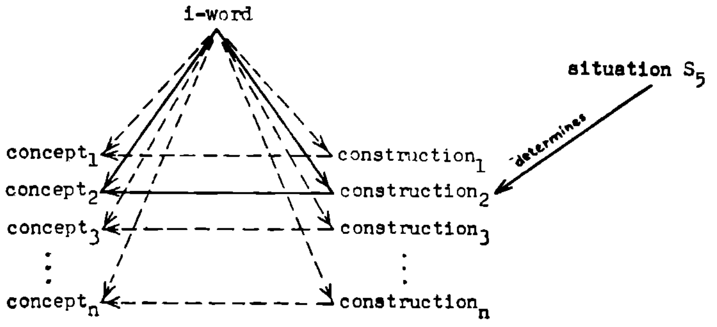
In comparison with our previous paper, logico-semantic considerations compel us to drop two, out of the original five, coordinates; the coordinate representing consent and dissent has proved to be incompatible with the present system of semantics, and the coordinate representing the tenses has lost its former significance, since it became clear that the basic problems of tense and time must be solved within semantics (see Chapter V).

Coordinates A^1, A^2 and A^3 (amply exemplified — under the names of A^2, A^3 and A^5, respectively — in the previous paper) form a space called the attitudinal space A. The combination of three attitudes (one of each kind) given by an ordered triple of values at A^1, A^2 and A^3 is conceived of as a certain point in A. Hence the sentence demonstrates, not one single attitude, but a combination of several attitudes, i. e. a certain point in the attitudinal space A, as is shown below:

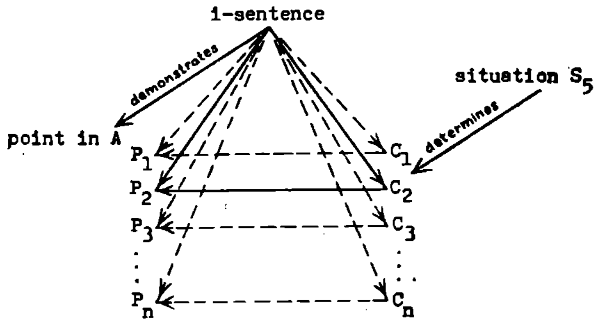
$$\begin{tikzcd}
 & & \text{sentence} \arrow[dll, "\text{demonstrates}"'] \arrow[ddl, "\text{denotes}"] \arrow[ddr, "\text{expresses}"] & \\
\text{point in A} & & & \\
 & \text{proposition} & & \text{construction} \arrow[ll, "\text{constructs}"']
\end{tikzcd}$$

The demonstration of a point in A is an integral part of every sentence; the sentence cannot exist without it as a sentence of a natural language. As it is always present, we have to reckon it in when analyzing the sentence. Apart from finding the construction, which in turn yields the respective proposition, we also have to find the point in A, i. e. to decipher the language user's attitudes. The proposition and the point are independent of each other. As long as some sentences express one and the same construction, they also denote the same proposition, no matter what combinations of attitudes (which of the points in A) they demonstrate. While propositions and constructions are dealt with within semantics, the attitudinal space A belongs to the sphere of pragmatics. To distinguish it from that part of pragmatics which deals with indexical expressions and related phenomena (external pragmatics, see Chapter VI), we call it internal pragmatics. Internal pragmatics and external pragmatics are to be carefully kept apart. Internal pragmatics is concerned with the above attitudes of a language user, which come into full play at the level of the sentence. External pragmatics deals with indexical expressions, which can be examined at all the levels, but first of all, at the level of the word.

An indexical word (i-word) taken by itself does not name any definite semantic function (concept), because it does not express any definite construction that would yield the function. It can only be associated with an open construction, which is but a scheme according to which a certain set of "closed" constructions can be built. Let the constructions numbered 1, 2, 3, . . ., n be members of such a set. Each of the constructions constructs the corresponding concept, which can be denoted by a given i-word. As the i-word can express any of the constructions and can denote any of the concepts at the same time, it does not express any particular construction nor any particular concept. The necessary choice is made by the situation in which the i-word is used. We can say that a definite situation (given by the ordered n-tuple of external pragmatic indices) determines which particular construction the given i-word expresses and, consequently, which particular concept it denotes:



If a sentence contains at least one i-word, we call it an i-sentence. Similar to i-words, i-sentences can only be associated with open constructions, so they do not express any definite constructions and do not denote any definite propositions either. It is the situation again that determines which construction is expressed and which proposition is denoted by the given i-sentence (C^1, C_2, C_3, ..., C_n are constructions, P_1, P_2, P_3, ..., P_n are corresponding propositions):



It is worth mentioning that an i-sentence demonstrates one point in the attitudinal space A in the same way as any other sentence. No matter which construction (and proposition) is determined by a certain situation, the point in A demonstrated by a given i-sentence remains the same.

We have established the relations between the semantic triangle and internal pragmatics on the one hand and between the triangle and external pragmatics on the other. What has not been examined so far is the relation between the two kinds of pragmatics.

From the viewpoint of sentence analysis, the demonstration of a point in A is independent of the situation.

Only in cases of internal pragmatic homonymy (there is more than one point in A that a given sentence can demonstrate), external pragmatics assists in disambiguating the sentence in this respect.

From the viewpoint of sentence synthesis, i.e., from the viewpoint of building up a sentence out of a given proposition and a point in A, the relation between internal and external pragmatics is much more complex than that mentioned above. Internal pragmatics represents a collection of attitudes that the user of a given natural language is able to employ, it is an attitude-supplying device, which is placed at every language user's disposal. The number of attitudinal combinations, i.e., the number of points in A, that can be employed is given by the range (richness) of the attitudinal space in the given natural language. The moment a sentence is to be uttered in the very act of communication, i. e. the temporal and the local factor, the context, both verbal and situational, the speaker, the hearer, etc., are at play, the number of points that can be employed will decrease. Thus external pragmatics operates as a restrictive factor in relation to internal pragmatics. The restriction, however, does not go as far as to determine only one of the possible combinations of attitudes, i.e. one definite point in A; it only restricts the range of points that can be employed in a given situation.

PART TWO

Chapter VIII

TOWARDS A COMPUTER ANALYSIS

The preceding chapters dealt with a theoretical conception of language which was supposed to become a suitable starting point for an adequate analysis of a natural language. If we want to analyze the expressions of a natural language, we need a collection of rules by means of which we shall be able to reveal what the language expressions are about or what objects they speak about. Our aim is to find such a collection of rules as would enable us to pass from natural language expressions to their semantic representations (meanings). By the semantic representation of an expression E, we shall understand what has been defined (see p. 130) as the linguistic construction expressed by an expression E. Apart from expressing a construction and denoting a proposition, any clausal expression of a natural language demonstrates the attitude of the language user towards the respective proposition. Hence in the course of an adequate analysis, we have to find not only the respective construction, but also the respective point in the attitudinal space.

In both theoretical linguistics and computer modelling (cf., e. g. Winograd 1972) the separate language phenomena are divided according to their functions at the following three levels: semantic, syntactic, and morphological. We shall try to show what connection can be found between the above three levels and the relations examined in the course of a computer analysis based on our ordered-triple theory.

From the viewpoint of a computer analysis of a natural language, the relation of denoting is of no immediate concern for us, because this relation is not direct, but mediated by the relations of expressing and constructing. The relation of constructing is not subject to a natural language analysis either, because the laws of constructing functions out of constructions are the laws of the employed system of intensional logic and are quite independent of the analyzed natural language. What remains to be of interest are the relations of expressing and demonstrating. As to the relation of expressing, there is a natural language expression on the one hand, and the respective linguistic construction on the other. Taking into account that the language expression has its syntactic and morphological structure and the respective construction is what is regarded as the respective semantic representation, we have to conclude that the relation of expressing includes:

(i) what is usually termed semantics (because the constructions construct what the corresponding expressions are about);

(ii) what is usually termed formal syntax (because without knowing the syntactic structure of a clausal expression the respective construction cannot be built);

(iii) what is usually termed morphology (because the syntactic structure of a clausal expression cannot be built without finding the properties of the clause-constituents, formally signalled by various morphemes).

As to the relation of demonstrating, there is a clausal expression on the one hand, and the respective n-tuple of attitudes (the respective point in the attitudinal space) on the other. The attitudes are signalled by both syntactic and morphological means, but they have nothing in common with the semantic representations (constructions). As for the above three levels, the relation of demonstrating only includes phenomena adduced sub (ii) and (iii), but does not include semantics (sub (i)). It follows that in the course of the analysis of a clausal expression E, the syntactic and the morphological phenomena group according to whether they serve semantics (building the respective construction) or internal pragmatics (constituting the respective point in the attitudinal space). These two groups do not exclude each other, because there are phenomena that simultaneously perform both the functions.

Although the ordered-triple theory as applied to a computer analysis does not employ the system of different levels, we shall keep these well-established distinctions in mind throughout the following discussion and shall always try to show when and where the correspondences may be found. In order to do this, we shall compare our conception with the functional generative model (Sgall et al. 1969), which uses the system of several levels. This comparison may be of special interest, because the functional generative model (FGM) has been built with due regard to the peculiarities of such synthetic language as Czech, and the first application of our conception to a computer analysis is designed for Czech too.

FGM works with strictly separated levels, which are formally defined as pairs of pushdown automata transferring the units of one level to the units of another level (a lower level, because FGM is defined as a generative procedure). FGM distinguishes the following levels:

(a) tectogrammatical level (or: the level of the semantic structure of a sentence, in FGM it is regarded as the semantic level);
(b) phenogrammatical level (corresponding to what is usually characterized as the syntactic level);
(c) morphemic level ⎫ (constituting together what is usually
(d) (morpho-) phonemic level ⎭ characterized as morphological level);
(e) phonetic level (the level of phonetic representation).

In FGM the generation of sentences proceeds from the highest level to the lower ones. At the highest, tectogrammatical level, FGM generates the semantic representations of sentences (FGM terminology). Their counterparts at the phenogrammatical level are the syntactic representations in the form of dependency trees. These are further adapted at the levels (c) and (d) till the final result in the form of phonetic representations of the generated sentences is obtained at the level (e).

The form of the semantic representations generated at the tectogrammatical level gives us no lead as to what logical apparatus they are connected with or what type of logical analysis they presuppose (although the authors are considering the possibility of connecting semantic representations with a logical calculus). For this reason it is rather difficult to specify their nature. Apparently, they are neither structural descriptions of sentences corresponding to the deep structures generated by the base as known from Chomsky's standard theory (1965), nor semantic representations in the sense proposed by Montague (1974). Perhaps, they might be characterized as (to a certain degree) semanticized syntactic representations of sentences. Figuratively, they are slightly "deeper" than representations covered by the term "deep structures" but at the same time they are "less deep" than semantic representations based on the apparatus of intensional logic. As we see it, the semantic representations in FGM are language dependent.

In the ordered-triple theory we distinguish three relatively independent components:

(a) pragmatics
(b) semantics
(c) formal syntax

The pragmatic component is subdivided into internal pragmatics and external pragmatics. Internal pragmatics is directly connected with the syntactic component, which – in the case of analysis – yields the necessary data for the identification of the respective point in the attitudinal space. External pragmatics presupposes a systemic connection with the semantic component through the mediation of open constructions and situations regarded as n-tuples of indices conveying the information about the time of utterance, the place, the speaker, the hearer, etc.

The semantic component includes constructions expressed by natural language expressions (i.e. semantic representations built on the basis of the apparatus of intensional logic) and intensions constructed by these constructions. What is also included is a collection of transition rules according to which the above language expressions are connected with the respective constructions (the transition from syntax to semantics).

The syntactic component includes natural language expressions and their formal structures. It contains the morphological and syntactic rules of a given language. In the case of analysis, the application of the rules leads to building syntactic representations of sentences in the form of labelled tree-graphs, which represent the starting point for building the respective constructions within the semantic component.

On comparing the above two conceptions, we can see that in FGM there is no separate level corresponding to our pragmatic component. As for external pragmatics, FGM takes indexical expressions into account but it only does so at the "semantic" level without showing how they are to be connected with the communicative situations. As for internal pragmatics, our attitudes have their counterparts in some of the rules at the tectogrammatical and the phenogrammatical level.

Most of the correspondences can be found within the sphere of syntax. The morphological rules of our syntactic component roughly correspond to those appearing at the morphemic and morphophonemic level in FGM. Our syntactic rules have their counterparts at the phenogrammatical level. It is possible to say that the ways of representing the syntactic structures of sentences in the two conceptions are very close. The differences can be seen only in formal universals, i.e. in formal devices handling the morphological and the syntactic rules. FGM uses the context-free formalism, the push-down store automata, and for the representation of the syntactic structures of sentences, the dependency trees. As is to be shown later on, the applied ordered-

-triple theory uses procedural rules and the labelled tree-graphs. Substantial universals, i.e. the collection of the employed grammatical categories, are basically the same in both the conceptions.

A striking difference lies in the sphere of semantics. Our semantic component is based on the theory of intensional logic with constructions as semantic representations of natural language expressions and with intensions as objects that these expressions speak about. In FGM there is hardly anything to correspond to our constructions and intensions. At the tectogrammatical level (which is the semantic level of FGM), FGM uses notions like "agent", "patient", etc., which have no counterpart in our conception (but could be introduced by means of meaning postulates). The above notions are abstractions lying somewhere between language expressions and constructions, i.e. on the path from syntax to semantics. From the point of view of logical semantics, they are — in our opinion — irrelevant. Since the authors of FGM do not go beyond these notions, they are compelled to leave referential semantics aside. The different approach to semantics has, of course, its consequences in the solution of the problem of the transition from semantics to syntax and vice versa.

At this juncture we should like to give the reasons leading us to an attempt at a computer implementation of our conception.

(i) From the very beginning we have aimed at a consistent theory of a natural language in all its aspects. In such a case it seems meaningful to consider the possibility of verifying the theory experimentally by applying it to the solution of certain linguistic problems in a computer. Our choice has fallen on the following problems:

(a) is it possible to algorithmize the transition from natural language expressions to the respective constructions and the respective points in the attitudinal space; if so,

(b) what collection of algorithms shall we need; and

(c) how can these algorithms be written in the form of programs which can be verified in a computer?

(ii) The second reason for a computer implementation of our theory has risen from practical needs. Many institutions work with rather extensive data base systems which, however, can only be communicated with by means of specialized artificial languages. There are demands for program modules enabling us to communicate with the data base systems in natural languages. So the attempt to build a question-answering module for the dialogue with a data base system, based on our ordered-triple theory, has become part of a research project sponsored by the Ministry of Education of the Czechoslovak Socialist Republic, the main task of which is to prepare a data base system for the needs of this institution. Apart from that, there are other fields, especially those of artificial intelligence and robotics, where question-answering and language understanding systems are built to operate as parts of more extensive computer systems (e.g., cognitive robots, and the like). What we are interested in is whether the ordered-triple theory is a suitable starting point for building a simple and not too large question-answering module which can be used for the dialogue with a data base system in Czech.

The above reasons have affected the handling of both theoretical and technical problems. The choice between the generative and recognition procedure has been made in favour of the latter. Although the generative procedure seems to be faster in offering results, the recognition procedure yields a more reliable basis for the attempts to reverse its direction automatically. As for the building of the system of algorithms, the adopted solution uses the strategy of building blocks, where the basic blocks (algorithms) are designed to cover the core of the theoretical model (the core of the ordered-triple theory). This enables us (1) to improve the individual blocks without changing the structure of the whole system of algorithms, and (2) to add new blocks if necessary. As to the computer, we had no choice. The only computer at our disposal was TESLA 200, which from the viewpoint of speed and the extent of memory has not met our demands. For this reason the syntactic analyzer for Czech (described in Chapter X) has been restricted not to exceed the core memory of TESLA 200. The problem of a suitable programming language has been solved in favour of LISP 1.5. Its advantages will become quite clear in the course of further discussion.

When discussing the reasons for a computer implementation of our theory, we said we were interested in whether the relations introduced within the ordered-triple framework could be written as algorithms that might be used in the course of the analysis of a natural language. The relations concerning a clausal expression (illustrated in Chapter VII, p. 139) are the following:

(i) denoting (between expressions and propositions)
(ii) expressing (between expressions and constructions)
(iii) constructing (between constructions and propositions)
(iv) demonstrating (between expressions and n-tuples of internal pragmatic indices)
(v) determining (between n-tuples of external pragmatic indices and constructions)

In the present chapter (see p. 142) we have already given reasons why — from the viewpoint of the analysis of a natural language — the relations (i) and (iii) are not a matter of immediate concern to us. The remaining relations (ii), (iv), and (v) are, in fact, systematic transitions from structures to structures. It follows that these transitions must be subject to algorithmization. Another question is what these algorithms will look like.

Let us take the relations representing transitions from natural language expressions to some other structures, i.e. the relations (ii) and (iv). In our opinion, the structure of an expression written (printed) in natural language notation is not suitable for a direct transition to another structure (the respective construction or the n-tuple of internal pragmatic indices). The solution adopted here consists in changing the linear sequence of natural language units into a tree structure containing data about the morphological and the syntactic properties of these units. It is only on the basis of the latter structure that the transition to constructions (expressing) and that to n-tuples of internal pragmatic indices (demonstrating) can be made. In short, our recognition procedure consists of the following three algorithms A, B, C, as shown in Figure 1.

The algorithm A is represented by the syntactic analyzer, which consists of the syntactic vocabulary (containing also morphological features) and the procedural grammar (containing the morphological and syntactic rules of a given natural language). It retrieves the input clausal expression E and

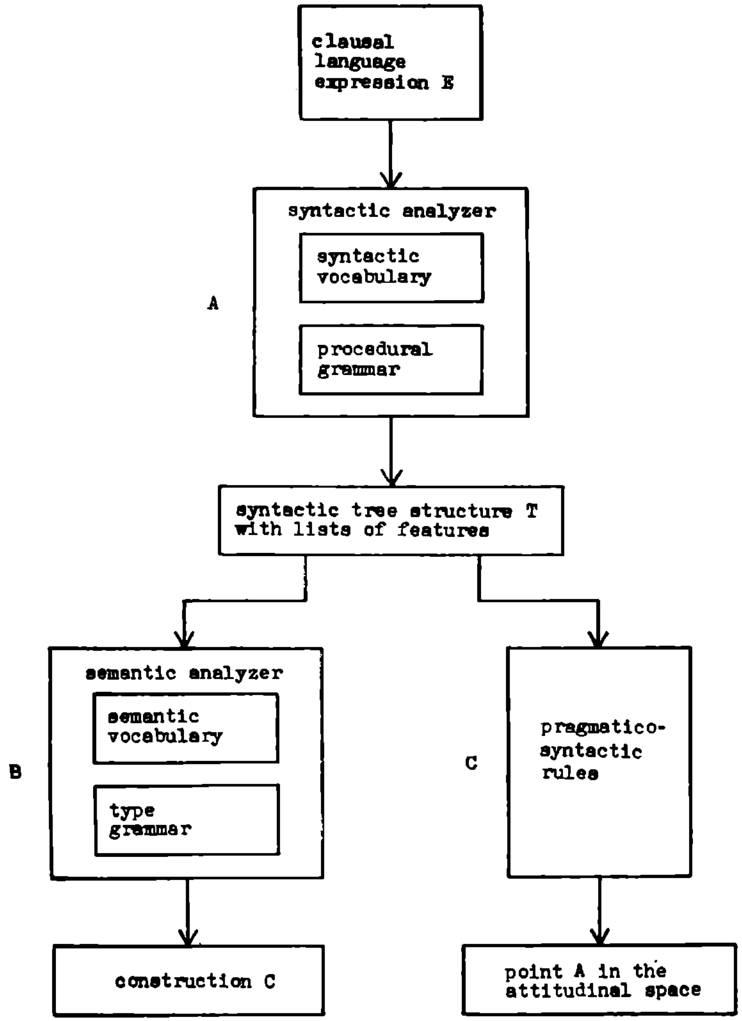

clausal language expression E
syntactic analyzer
syntactic vocabulary
procedural grammar
A
syntactic tree structure T with lists of features
semantic analyzer
semantic vocabulary
type grammar
B
C
pragmatico-syntactic rules
construction C
point A in the attitudinal space

Figure 1

produces its syntactic tree structure T with lists of features attached to the respective nodes. From the viewpoint of the whole recognition procedure, the output of the algorithm A (the syntactic tree structure) only constitutes an intermediate result, which provides the algorithms B and C with the relevant data.

The algorithm B is represented by the semantic analyzer, which consists of the semantic vocabulary (containing the entries of the syntactic vocabulary with their (logical) types and idiosyncratic features) and the type grammar (containing the rules of composition and decomposition of types as well as the operations of application and abstraction). It retrieves the syntactic tree T and the respective entries of the semantic vocabulary and produces the corresponding construction C in the form of both the semantic tree and its linearized counterpart. The output of the algorithm B (the construction C) is the final result of the semantic branch of the recognition procedure.

The algorithm C is represented by the pragmatico-syntactic rules, which retrieve the relevant data offered by the syntactic tree T and produce the respective point A (given by an n-tuple of internal pragmatic indices) in the attitudinal space as the final result of the internal pragmatic branch of the recognition procedure.

At the present stage of our research project (Spring 1977), we have programmed and successfully tested the syntactic analyzer for Czech. Its characteristics (with examples of the analysis) are given in the following two chapters of the present paper. The preliminary version of the semantic analyzer has already been programmed and is now tested in the Laboratory of Computing Machines. The algorithm C has not yet been programmed except for the part retrieving the data relevant to internal pragmatics from the syntactic tree structure T.

For the time being, we are only interested in expressions of what has been termed language "dead" (cf. Svoboda et al. 1976), and for this reason we do not intend either to deal with or to algorithmize the relation of determining (see (v) on the above list). In our opinion, this can only be done after all the other parts of the proposed recognition procedure have been successfully tested.

Chapter IX

A PROCEDURAL GRAMMAR (OF CZECH)

Before writing a formal grammar of a natural language, we have to decide what kind of formal apparatus to choose. The following are the theoretical possibilities that have been weighed up from the viewpoint of our requirements:

(a) context-free rules
(b) dependency rules
(c) context-sensitive rules
(d) standard generative transformational framework (Chomsky 1965)
(e) functional generative framework (Sgall et al. 1969)

Taken alone, the items (a) and (b) are too simple to constitute a suitable basis for an adequate grammar of a natural language. Grammars with a reasonable degree of adequacy make use of either (a) or (b) in combination with a more powerful formal device like (c) or (d). This results in theoretically interesting grammars, which — to their disadvantage — are too complicated from the point of view of computer applications, because they are not easy to algorithmize. (For attempts to build such grammars and to test them in a computer, see, e. g., Zwicky, Friedman, Hall, Walker 1965, Machová 1976).

Looking for a way out of this situation, Woods (1969) and Winograd (1972) have convincingly shown that it is possible to build grammars that are practically manageable because they are not too large, but which are — at the same time — well-suited to be employed within comparatively extensive computer systems. Woods based his grammar on appropriately adapted finite automata, taking full advantage of their recursivity and flexibility (nowadays, grammars of this type are known as augmented transition network (ATN) grammars). Winograd made use of context-free rules written as procedures, which represent a simple and yet comparatively powerful device for building a formal grammar. Both Woods' and Winograd's grammar offer a very interesting solution of our problem. Nevertheless, when building an adequate grammar of Czech, we had to make our choice. At this juncture the decisive factor was that Czech — from the viewpoint of syntax — is a synthetic language with free word order. As Winograd's approach was better suited for handling the free order of sentence constituents, we gave it priority over an ATN grammar and decided to build a procedural grammar based on Winograd, which — of course — was to be as different from Winograd's as Czech is different from English.

In our grammar we use context-free rules (CF-rules) reformulated as procedures. Before proceeding any further, we should like to explain what is meant here by procedures. A procedure is understood as a sequence of steps where some of the steps are represented by action statements (actions that are to be carried out) and the other steps are represented by conditional statements (conditions that are to be tested in order to decide which action is to be executed next).

An example of an action statement: "Look up the word in the vocabulary." or "Attach the node to the tree structure of the analyzed sentence."

An example of a conditional statement: "Is the analyzed word an adjective or a noun?" or "Is the noun in the nominative or in the accusative?", etc.

The best way to express a procedure formally is to write it as a flow-diagram consisting of a sequence of both action and conditional statements. From the above characteristics it can be easily seen that the notion of procedure is close to that of program. The statements within procedures may be quite complex and are frequently expressed in linguistic terms (see examples above). By writing the corresponding computer program, the statements are changed into sequences of simpler instructions as required by the employed programming language.

Now we shall show how the CF-rules are reformulated as procedures. This can be demonstrated in two steps:

(a) We select a set of CF-rules and divide them into subsets according to the particular kind of sentence constituents they are designed to analyze.

(b) To each of the subsets of CF-rules, we add a series of "operational data". Thereby we get the procedures, which are most easily expressed as flow-diagrams.

As an example, have the subset of the following CF-rules (1), (2), (3).

$$\text{(1) NG} \rightarrow \left\{\begin{array}{l} \text{(PROND) (PRONP) (NUMO) (NUMK) (AD) (A) (N) (NPR)} \\ \text{PRONUN} \\ \text{PRONPER} \\ \text{PRONQ} \\ \text{PRONR} \\ \text{NG NGGEN} \\ \text{NG PREPG} \end{array}\right\}$$

$$\text{(2) PREPG} \rightarrow \{\text{PREP NG}\}$$

$$\text{(3) NGGEN} \rightarrow \{\text{NG (with the feature GEN)}\}$$

It is apparent that the CF-rules (1), (2), (3) are not concerned with such grammatical categories as case, gender and number. The consequences are that on the one hand, the above subset of CF-rules is very simple and transparent, but on the other hand, the CF-rules (1), (2), (3) generate (or recognize) noun groups and prepositional groups irrespective of the concord of the above categories within the respective groups, which amounts to generating (or recognizing) also ill-formed (ungrammatical) phrases in Czech.

The former consequence makes us preserve the CF framework as it is. It really yields a simple and transparent description of the basic syntactic structures of Czech in terms of parts of speech and sentence constituents. The latter consequence calls for some improvement (modification) of the CF framework.

The attempts to include the grammatical categories in the CF-rules have consisted in adding special indices to the categorial symbols, which leads to the introduction of such complex symbols as $N_{\mathrm{NOMSGMASK}}$ or $V_{\mathrm{3SGFUTINDACTIMPF}}$. It is quite clear that these "sophisticated" symbols are not easy to handle. Moreover, they multiply the number of CF-rules in the grammar.

We have evaded this difficulty by incorporating the data concerning grammatical categories into procedures. This has been done in the following way:

(a) the respective procedure includes steps in which the particular part of speech is identified;

(b) for any input word, there are steps by which its grammatical categories (features) are taken from the vocabulary and are attached to the corresponding data concerning the part of speech;

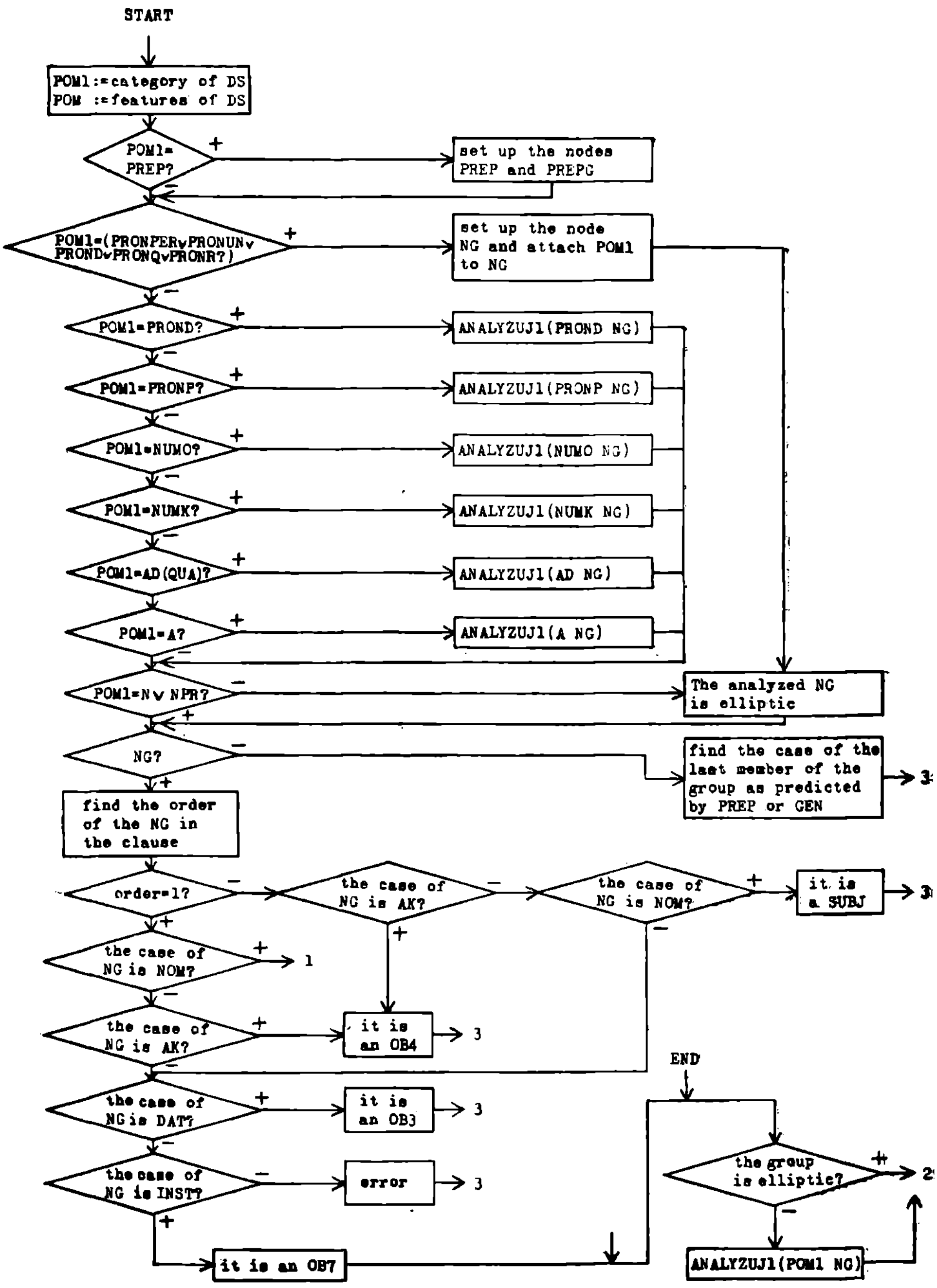
(c) there are steps by which the features are tested and the grammatical concord and other contextual dependencies checked;

(d) there are steps by which the structure of the analyzed sentence is built and later expressed in the form of a labelled tree-graph.

The sequences of the above steps will be called procedural rules (P-rules). Each of them consists of (1) the grammatical part, containing the particular grammatical rules (e.g., (a)), and (2) the operational part, containing the set of auxiliary operations (see, e.g., the steps (b), (c), (d)).

Thus the P-rule based on the CF-rules (1), (2), (3) (see p. 149) and therefore recognizing Czech noun groups and prepositional groups is a procedure which has a Czech clause as its input, finds the respective words in it, provides them with the corresponding data concerning the parts of speech and relevant features, and tests the grammatical concord by intersecting the case-number--gender triples of features (which are already accompanying the elements of the respective noun group). If the concord-test fails, the analyzed noun or prepositional group is classified as ungrammatical, the P-rule announces an error and stops working; if the concord-test is positive, the P-rule goes on and — as its output — builds the respective part of the tree structure connected with the analyzed constituent. Then it examines the next input word and, according to its part of speech, decides whether to pass the control to another P-rule or to call itself again. To give the reader an idea of the procedural character of this rule, we present its flow-diagram (Fig. 1, pp. 151—52).

In the preceding paragraphs we have shown what the P-rules look like and how they can be arrived at on the basis of a chosen set of CF-rules. The collection



START
POM1:=category of DS
POM :=features of DS
POM1=PREP?
set up the nodes PREP and PREPG
POM1=(PRONPER∨PRONUN∨PROND∨PRONQ∨PRONR?)
set up the node NG and attach POM1 to NG
POM1=PROND?
ANALYZUJ1(PROND NG)
POM1=PRONP?
ANALYZUJ1(PRONP NG)
POM1=NUMO?
ANALYZUJ1(NUMO NG)
POM1=NUMK?
ANALYZUJ1(NUMK NG)
POM1=AD(QUA)?
ANALYZUJ1(AD NG)
POM1=A?
ANALYZUJ1(A NG)
POM1=N∨NPR?
The analyzed NG is elliptic
NG?
find the case of the last member of the group as predicted by PREP or GEN
find the order of the NG in the clause
order=1?
the case of NG is AK?
the case of NG is NOM?
it is a SUBJ
the case of NG is NOM?
1
the case of NG is AK?
it is an OB4
3
the case of NG is DAT?
it is an OB3
3
the case of NG is INST?
error
3
it is an OB7
END
the group is elliptic?
2
ANALYZUJ1(POM1 NG)
+
−

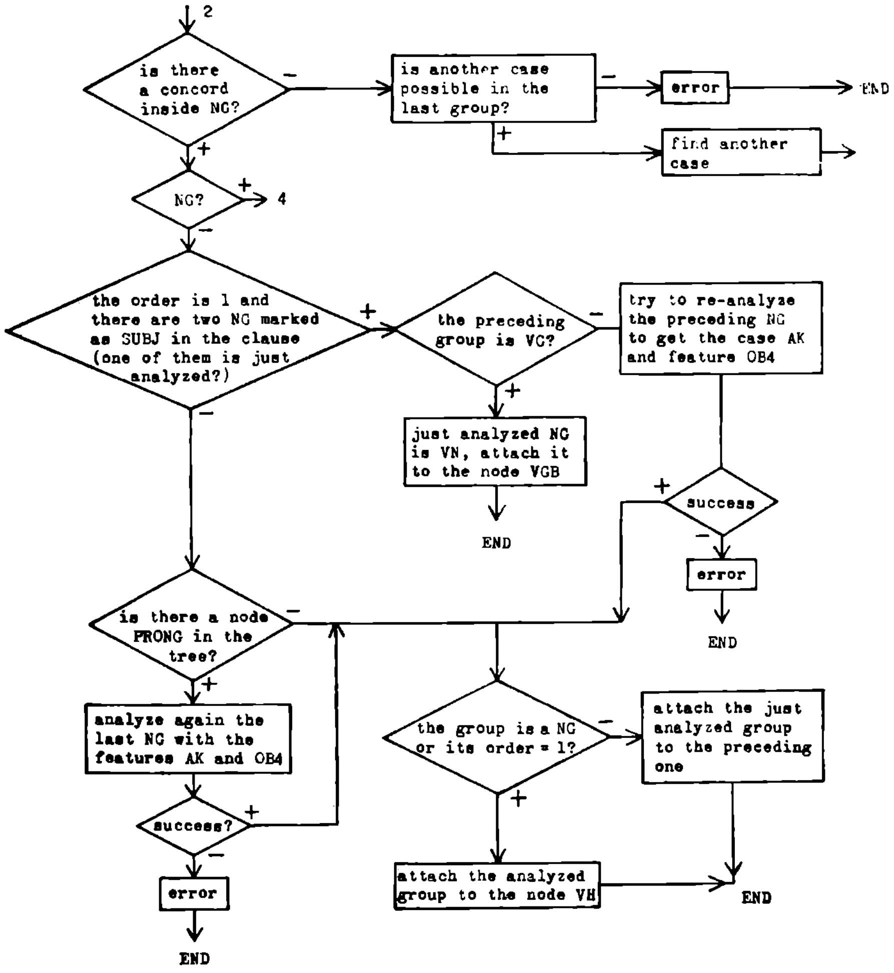

2
is there a concord inside NG?
is another case possible in the last group?
error
END
find another case
NG?
4
the order is 1 and there are two NG marked as SUBJ in the clause (one of them is just analyzed?)
the preceding group is VG?
try to re-analyze the preceding NG to get the case AK and feature OB4
just analyzed NG is VN, attach it to the node VGB
END
success
error
END
is there a node PRONG in the tree?
analyze again the last NG with the features AK and OB4
success?
error
END
the group is a NG or its order = 1?
attach the just analyzed group to the preceding one
attach the analyzed group to the node VH
END

Figure 1

POM1	–	variable under which the currently analyzed word (DS) is stored
POM	–	variable under which the features of DS are stored
ANALYZUJ	–	universal analyzing (LISP) function which builds the respective subtree in the currently analyzed group and attaches it to its main node (NG, VG, etc.)

of P-rules enables us to build a formal grammar of Czech, which we shall call a procedural grammar (for further specification, see later on).

Apart from purely grammatical information (see (a), p. 150), the P-rules also contain data concerning the strategy of the analysis (see (b), (c), (d) above), which might be singled out as a special collection of rules. In some formal grammars the above two collections of data are strictly separated. Our procedural grammar does not require such separation, which does not mean that we do not take the difference between the two kinds of rules into account.

If the procedural grammar is to work, it is necessary to connect it with a vocabulary. The procedural grammar and the vocabulary of Czech word-forms constitute the syntactic analyzer (for Czech), which is described in Chapter IX.

At this point we should like to touch upon the question of how the morphological rules of such language as Czech are incorporated into the syntactic analyzer. Basically, this can be done in the following two ways:

(i) The collection of morphological rules represents the basis of an independent and possibly complete morphological analyzer, consisting of the tables of suffixes with the data concerning the grammatical categories, the vocabulary of word-stems with the data concerning the parts of speech, the list of non-flectional words (adverbs, prepositions, etc.), the list of flectional words with irregular or unique flection (some pronouns, irregular verbs) and, finally, the rules determining the strategy of the whole morphological analysis. The input of this analyzer are clausal expressions or fragments of continuous text; the output is a vocabulary containing separate word--forms with the information concerning the respective part of speech and possible grammatical categories. The vocabulary of this kind can represent an input for the above procedural grammar.

(ii) The morphological rules are "incorporated" in the vocabulary in advance. It is possible to make a vocabulary of all the word-forms together with the relevant grammatical data attached to them. Such a "ready-made" vocabulary, which already includes morphology can be prepared manually beforehand, and from the viewpoint of our procedural grammar, it is quite irrelevant whether the vocabulary represents the output of the morphological analyzer or is the result of the pre-computer human activity. (It may be objected that theoretically, this solution is not quite "pure" as it resigns from one part of the linguistic analysis. Technically, it leads to the multiplication of entries in the vocabulary, but with about 2500 entries (sufficient for currently used applications) and a medium-size computer, the reactions are as quick as those sub (i).)

Since we have no suitable morphological analyzer for Czech at our disposal, we have decided in favour of the solution (ii) and have prepared a "ready-made" vocabulary meeting the requirements of our procedural grammar.

(A note: We devised a morphological analyzer for the nominal parts of speech which was able to analyze Czech nouns, adjectives, numerals, and pronouns. It was successfully tested in the SAAB D21 computer but it could not be connected with our syntactic analyzer, because the latter was programmed in LISP 1.5 and tested in the TESLA 200 computer, which was not directly compatible with SAAB D 21. The other "verb" part of the morphological analyzer only exists in the form of a flow-diagram and has not been programmed.)

After all the necessary detours, we shall characterize a procedural grammar (P-grammar) and define our P-grammar designed for Czech:

A P-grammar is a set of P-rules. Each rule is of the form A(X, Y, . . ., Z) where A is the name of a P-rule (or the corresponding LISP function) and the symbols X, Y, Z denote the sentence constituents that can be analyzed by the above rule (they are the arguments of the corresponding LISP function).

The symbols X, Y, Z may or may not be present. In the latter case the name of a P-rule itself indicates for what sentence constituent it is designed. (This is based on the LISP idea of functions without arguments.) Particularly for Czech:
Definition: PG = {ANAL(NG, PREPG, NGGEN), ANALVG( ),
ANALAD( ), ANALVETA(ZDROJ)},
where
ANAL(NG, PREPG, NGGEN) is the P-rule designed to analyze Czech noun groups and prepositional groups of all the case-number-gender combinations (for the characteristics of this rule, see below);

ANALVG( ) is the P-rule designed to analyze Czech verb groups consisting of less than four verb-constituents;
ANALAD( ) is the P-rule designed to analyze Czech adverbial groups of time, manner, place, and measure;
ANALVETA(ZDROJ) is the top-level P-rule designed to control the activity of the other P-rules and complete the whole analysis by producing the tree structure of the analyzed sentence.

We shall give a brief characteristics of the individual P-rules and shall also show the way in which the P-rules deal with the respective constituents. As ANAL has already been characterized (see p. 150—152 and Fig. 1), just a few additional points will be mentioned here. One important trait of all the P-rules is their recursivity. On finishing the analysis of a given noun group (NG) or a prepositional group (PREPG) and finding that the next group is an NG or a PREPG again, ANAL can re-call itself. Further, ANAL can recognize both the maximum NGs (i.e. NGs containing all the permissible elements) and the minimum (non-zero elliptic) NGs (i.e. those with most of the possible elements omitted). The maximum NGs are to be dealt with later on; an example of the minimum NG is the Czech demonstrative pronoun ten (that) used in the sense "that man over there". PREPGs are analyzed in the same way as NGs except for the prepositions (which in most cases predict the grammatical case of the given PREPG quite unambiguously). As its part, ANAL contains SHODNOST (agreement), which tests the grammatical concord within both NGs and PREPGs. As the result of its activity, ANAL produces such tree structures as shown in Figures 2 and 3. If possible (i.e. if the nominative is not homonymous with one of the other grammatical cases), ANAL also determines whether the analyzed NG is the subject (SUBJ) or and object (OB4, OB3, OB2, OB7, etc.) of the respective clause.

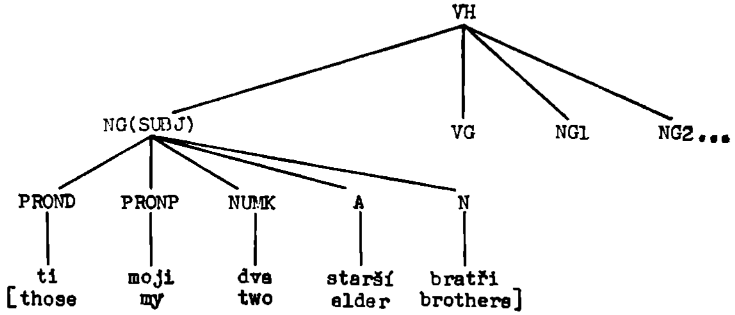


Figure 2

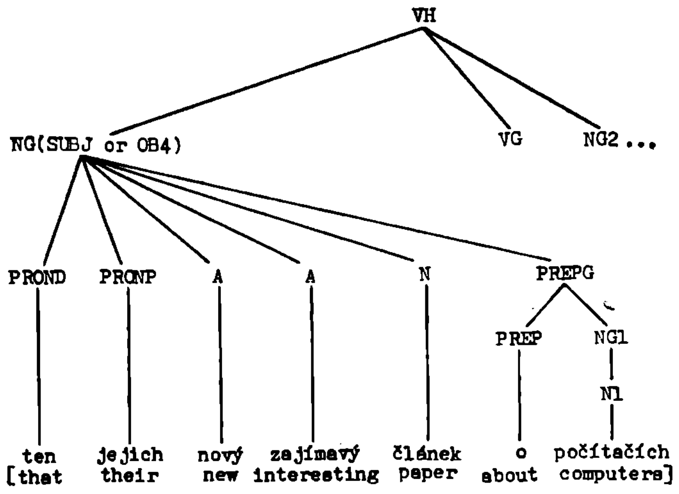


Figure 3

ANALVG( ) evaluates Czech verb groups (VGs), consisting of one, two, or three verb-constituents, as subtrees of the respective sentence tree structure. An important trait of ANALVG( ) is its context-sensitivity, without which the Czech analytic and often discontinuous verb-forms cannot be successfully analyzed. ANALVG is especially well-suited for handling the verb-forms within clauses displaying free word order, and in this respect it is more powerful than the corresponding part of the ATN grammars. ANALVG finds the separate verb-constituents and their arbitrary combinations within the analyzed clause, makes the resulting analytical verb-form out of them, and tests them for the grammatical concord with the respective subject NG. Finally, it produces the tree structure typical of the given verb-form. (See Figures 4 and 5.)

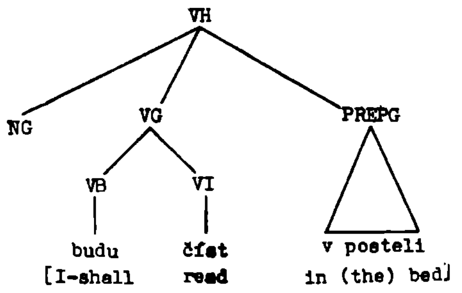


Figure 4

ANALAD( ) analyzes Czech adverbs that constitute independent adverbial groups (ADGs) within a given clause. ANALAD recognizes ADGs represented by adverbs of time, place, and manner, or their combinations with adverbs

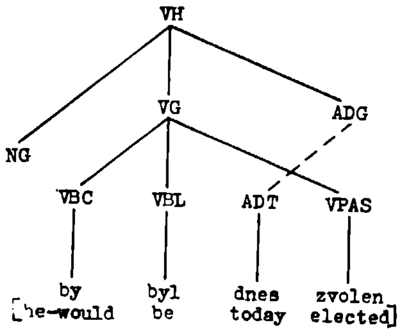


Figure 5

of measure. It also includes the strategy for dealing with context dependencies typical of ADGs. At present, however, ANALAD cannot recognize what is called adverbial grammatical cases (NGs or PREPGs that function as ADGs of place, time, manner, cause, etc.). Recognizing the adverbial status of a given NG or PREPG appears to be a complicated task, which requires to take the idiosyncratic features of nouns and prepositions into account. Examples of the tree structures produced by ANALAD are diagrammed in Figures 6 and 7.

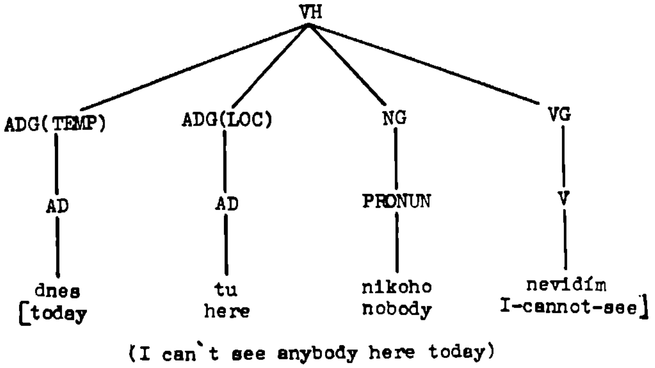


Figure 6

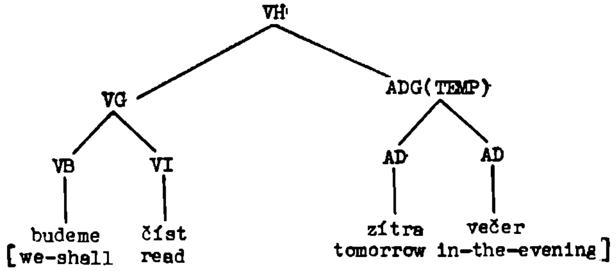


Figure 7

The main purpose of the top-level rule ANALVETA(ZDROJ) is to control the activity of the other P-rules and to coordinate their actions. ANALVETA activates the other P-rules according to the signals coming from the analyzed clause, re-analyzes – if necessary – some of the already produced groups or even the whole clause, finds the necessary features for the node VH (main clause), and builds the ultimate tree structure of the input clause, as exemplified in Figure 8.

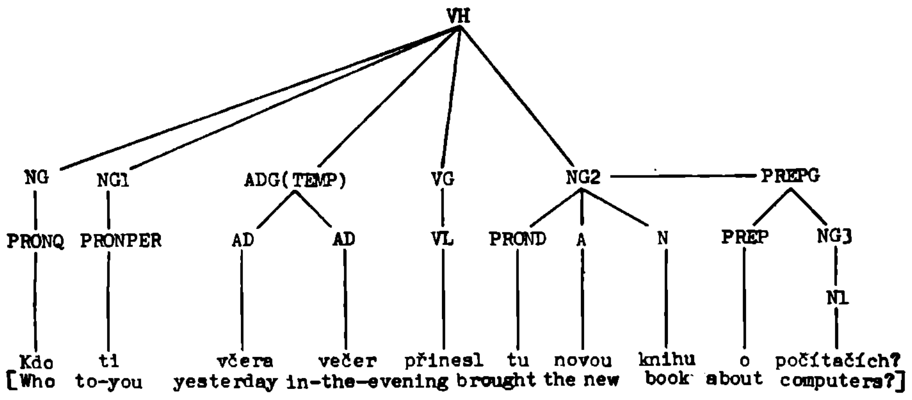

VH
NG
NG1
ADG(TEMP)
VG
NG2
PREPG
PRONQ
PRONPER
AD
AD
VL
PROND
A
N
PREP
NG3
N1
Kdo ti včera večer přinesl tu novou knihu o počítačích?
[Who to-you yesterday in-the-evening brought the new book about computers?]

Figure 8

It has to be noted here that ANALVETA(ZDROJ) is able to analyze what may be called the maximum clauses, because it includes the maximum grammatical rules. Thanks to them it can capture all the possible combinations of NGs, PREPGs, VGs, and ADGs, and in this way deal with any Czech clause containing from one to seven – exceptionally even more – groups. (More than seven groups are not expected to occur in a normal Czech clause.)

In the preceding paragraphs we have assumed that behind our P-grammar, there is a "ready-made" vocabulary to which the respective P-rules have direct access. Let the vocabulary be called the syntactic vocabulary. It represents an independent and at the same time changeable unit, which is directly connected with the P-grammar. Since the organization of its entries and their features is quite different from that used in the vocabularies of the CF-type grammars, we shall devote the following part of the present chapter to the syntactic vocabulary.

In CF-grammars the vocabulary (of Czech) is written as a set of rules having, for example, the form $N_{NOMSGMASK} \rightarrow$ {muž, učitel, ...} or $V_{3SGPREINDACTIMPF} \rightarrow$ {spí, sedí, čte, ...}, etc. In order to capture all the possible combinations of the grammatical categories (features) connected with the individual word-forms, one is compelled to introduce such complex symbols as above. Since within the CF-framework the complex symbols cannot be further analyzed in any reasonable way, there is no immediate access to the particular grammatical categories. These categories can be neither tested nor retrieved in order to guide the course of the analysis.

In our P-grammar the syntactic vocabulary does not belong to any P-rules, the entries are defined independently and take the form of a hierarchic list (in LISP, the property list). The separate definition of the syntactic vocabulary enables any P-rule to have direct access, not only to any entry word-form, but also to any of its grammatical categories (features). As an illustration, we present two typical entries, one exemplifying nouns (here the Czech noun muži [men]) and their features, the other exemplifying verbs (here čte [he/she/it reads or is reading]) and their features.

```
MUZI ((N)      (DAT SG MASK)  (LOK SG MASK)   (NOM PL MASK)  (INST PL MASK)
        |            |              |               |               |
        |            |              |               |               |
     part of    first case-      second          third           fourth
     speech     -number-         case-number-    case-number-    case-number-
                -gender triple   -gender triple  -gender         -gender
                                                 triple          triple
(HUM)))
  |
  |
 idiosyncratic
 feature

(CTE ((V)    (TO   SG  PRES  IND   ACT    ND)      (NOM AK) ))
       |      |    |    |     |     |      |           |
       |      | number  |     |     |      |         nominal
   part of   person     |    mood   |      |         case framework
   speech    (here:   tense        voice   |         required by the
             3rd)                        aspect      verb
```

The above entries include all the grammatical categories revealed by the respective word-forms. The noun muži is homonymous since it may represent Dative Sg., Locative Sg., Nominative Pl., or Instrumental Pl. In the course of the analysis, it is the task of the respective P-rule (here ANAL) to decide which case-number-gender combination is to be selected as correct. It is apparent that the context has to be consulted, sometimes as large as the respective clause.

The internal organization of entries in the form of a hierarchic list has the following advantages:
(a) it has the same form as (or very similar to) the output of the morphological analyzer (if there were any, of course);
(b) any entry may be extended by additional data concerning the word-form it contains, i.e. any entry may be provided with additional syntactic, semantic, and pragmatic features relevant to further analysis;
(c) the data included in an entry may be further structured in any desired way according to the employed system of features and may be retrieved at any point of the analysis;
(d) any entry organized in the above way may be easily programmed in the programming language LISP 1.5 by means of the standard LISP function DEFLIST (DEFLIST defines the vocabulary as a property list, which may be modified by means of the standard functions PUT and GET).

Chapter X

THE SYNTACTIC ANALYZER (FOR CZECH)

The syntactic analyzer (SNA) for Czech is based on the syntactic vocabulary of Czech and the P-grammar described above. Basically, it uses the top-to-bottom and the left-to-right strategy, but whenever necessary it can switch over to its bottom-to-top and right-to-left counterpart.

SNA is written in the programming language LISP 1.5 as a set of 34 LISP functions. It has been implemented in the TESLA 200 computer in the Laboratory of Computing Machines of the Brno Institute of Technology. The program has been written by I. Palová (1976). From the above 34 functions, 4 are regarded as basic, the remaining as auxiliary. The basic functions (or programs) correspond to four P-rules described in Chapter IX. The names of the P-rules and those of the respective LISP functions are the same.

(1) ANALVETA(ZDROJ) is a LISP function of one argument that takes the form of the LISP list and represents the analyzed clause. It controls the whole analysis, and – according to the situation in the analyzed clause – it passes the control to the lower three functions, which are designed to analyze particular nominal, verbal, and adverbial groups.

(2) ANAL(CO) is a LISP function based on the P-rule ANAL(NG, PREPG, NGGEN). According to the formal signals coming from the analyzed clause, it recognizes which atom is to be evaluated as its argument. If a noun group of a given clause is to be analyzed, it takes NG as its argument; if it is a prepositional group, if takes PREPG; and if it is a noun group in Genitive which immediately follows some other NG, it takes NGGEN. (NGGENs are singled out because they hold a specific position in the clause structure as attributes.) ANAL(CO) tests the following collections of conditions: (a) the grammatical concord within an NG, a PREPG, or an NGGEN; (b) the grammatical case of the whole group. If the case cannot be determined unambiguously, ANAL (CO) examines the context within the given clause. (It tests the order of the groups, the parts of speech they consist of, their animateness or inanimateness, etc.) These tests reduce a large number of false analyses, though it is fair to say that they do not always exclude them completely. ANAL (CO) is built in such a way that the automatic back-up is switched on whenever the obtained results appear to be unsatisfactory. They are automatically re-examined and a new analysis is attempted without the control being returned to the main function ANALVETA (ZDROJ). This automatic back-up solves most of the cases regarded as conflicting.

(3) ANALVG( ) is a LISP function without arguments which analyzes the verb group of a clause. It is based on the P-rule ANALVG introduced above. The endeavour to find the algorithm for looking up the separate components of an analytic verb-form and their features has led to the introduction of two auxiliary LISP functions, POLOZ (meaning "put down") and PROVED (meaning "carry out"). These functions assist in overcoming the difficulties caused by the fact that Czech verb-forms frequently consist of several components appearing in various sequences, which are often discontinuous. The analysis of a typical Czech analytic

verb-form consisting of more constituents begins after its first component has been found. This component is of considerable diagnostic value, because it predicts which of the verb-constituents can be expected in the non--analyzed part of the clause. This is done by means of two auxiliary functions, MOVE and MOVE 1, which go through the non-analyzed part of the clause and look for the combinations of verb components predicted by the first component. As soon as the respective combination has been found, the function POLOZ is called. This function has three arguments, K1, K2, and SEZ. All of them are variables of lists of grammatical features. POLOZ finds whether the grammatical features on the lists K1 and K2 are also among those retrieved (together with the respective verb-constituents) from the syntactic vocabulary. If so, POLOZ only selects such features as are on the list SEZ, which is different for different combinations of verb-constituents. For instance, if the analyzed constituent of a given VG is VBL (the active past participle of být i [to be]), POLOZ may have the following arguments: (POLOZ (QUOTE PRET) NIL (TENSE)), which means that if VBL has PRET (K1) among its features and if K2 is irrelevant (NIL), the contents of SEZ in this case will be a list of one feature, namely the feature PRET. This feature becomes the resulting feature indicating the tense for the whole VG.

PROVED is a function of two arguments, CO and BEZ, where CO takes the value VG or VGB, and BEZ is a parameter taking the value TRUE or NIL. PROVED analyzes the respective verb-constiuent (by finding its grammatical features), builds the respective part of the VG subtree, and transfers the features of the analyzed verb-constituent to the list of features of the whole VG. If there are no features to be transferred, the variable BEZ takes the value NIL, and the analysis of the respective VG is finished or the next verb-constituent is retrieved. Then PROVED puts together the whole analytic verb-form, produces its tree structure with the characteristic features, and attaches it to the node VH (main clause).

(4) ANALAD( ) is a LISP function without arguments; its task is to analyze the adverbial groups in a clause. It is based on the P-rule of the same name.

The set of the auxiliary LISP functions will not be discussed here since the explanation would require the basic knowledge of both LISP 1.5 and some of the programming details. Let it suffice to say that the set of auxiliary functions over which the four basic functions are built may be characterized as a mini-language designed for building syntactic analyzers.

The testing of the last version of the SNA for Czech was finished in December 1976. By that time about 250 Czech clauses representing the most important types of Czech sentences had been successfully tested.

As an example, we take the following Czech sentence analyzed in the computer:

(1) Kdo ti zítra ráno přinese tu novou knihu o počítačových systémech?
[Who to-you tomorrow in-the-morning will-bring the new book about computer systems?]

SNA started with looking up the first word of (1) in the syntactic vocabulary. If it had not been found, an error would have been announced on the lineprinter and the analysis stopped. In our case, however, the first step was successful; the first word was found and its word category (indicating the part of speech) tested. The result determined which of the basic functions was to be activated. Since the word category of the first word was PRONQ (the interrogative pronoun) the assumption was that the first group would be an NG, and for this reason the function ANAL(CO) wad called to start its actions. After this the features of the first word were taken from the syntactic vocabulary, and the list obtained in this way was attached to the node PRONQ. The only possible grammatical case on the list was NOM. Therefore, the analyzed NG was labelled as SUBJ. Since the word category of the analyzed word was PRONQ, ANAL knew that the NG only consisted of one constituent (word) and could finish the analysis of this NG by producing its list of features and attaching it to the node VH.

Then the word category of the next word was tested. Since it was PRONPER (the personal pronoun ti [to-you]), ANAL(CO) called itself with the argument NG. Now the actions of ANAL(CO) were similar to those described in the preceding case. It was found that PRONPER forms an NG in Dative, and it was labelled as OB3. Finally, the NG with its list of features was attached to the node VH.

The next word tested was an AD (the adverb zítra [tomorrow]) and the function ANALAD( ) was called. It found that the analyzed adverb was an adverb of time. According to the program, it had to look round and test whether the next word or one of the following words would be an adverb of time again. The test was positive (the adverb ráno [in-the-morning] was found), and ANALAD( ) formed the node ADG with the feature TEMP. After the node ADG had been attached to VH, the control was passed to ANALVETA(ZDROJ), which found that the next input word was a verb (V — denoting a simple form, přinese [will-bring]), and, therefore, called the function ANALVG( ). Since the first constituent of VG was V (a finite verb-form in (formal) PRES, different from "be"), it was clear that VG would only consist of one constituent. Then the feature of V were taken from the syntactic vocabulary, and the resulting list of features for the node VG was produced. It included the features TO, SG, FUT, IND, ACT, DOK, NIL, respectively denoting the third person, singular, future tense, indicative, active voice, perfective aspect, without negation. Now the node VG was attached to VH as its daughter.

In the next step the word category of the following word was tested. It was a PROND (the demonstrative pronoun tu [the, that]), and ANAL(CO) was called with the argument NG. It started its usual actions of looking up the features, producing the node PROND with its list of features and attaching it to its parent node NG. After this, ANAL(CO) looked for the next word, assuming that it could be PRONP (a possessive pronoun) as required by the built-in grammatical rule based on the CF-rule presented above (see p. 149). Since the answer was negative, ANAL went on and looked for AD with the feature QUA (an adverb of measure), which was not found either. In the next step ANAL(CO) discovered that the input word was an A (the adjective novou [new]), and the features of this A were retrieved from the syntactic vocabulary. Then it was tested whether there was another adjective on the input, but the answer was negative. This caused ANAL(CO) to look for the next constituent of the analyzed NG. It was found that it was an N (the noun knihu [book]). Then ANAL(CO) tested whether this noun was followed by a proper noun in order to cover such cases as ten slavný spisovatel Čapek [the famous writer Čapek]. Since the test was negative, ANAL(CO) retrieved the list of features of the analyzed N, tested the concord within the whole NG and produced the resulting list for the node NG in the form (OB4 AK SG FEM NEZ), denoting accusatival object, accusative, singular, feminine, inanimate. Then the NG was attached to the node VH as its daughter, and ANAL(CO) came to the conclusion that the NG was ultimately analyzed.

Testing the word category of the next word and finding that it was a PREP (the preposition o [about]), ANAL(CO) called itself with the argument PREPG. The PREP was analyzed, its list of features was produced, and the node PREP was attached to PREPG as its daughter. The next word being an A [the adjective počítačových [computer]), the analysis went on in the same way as in the case of NG till the last constituent of the analyzed NG was found (here the noun systémech [systems]). Then the concord within the whole PREPG was tested with the result LOK (locative). As there were no other signals concerning its semantic status, the analyzed PREPG was attached to the preceding node NG as its daughter. By this, the analysis of PREPG was finished and the next input word could be tested.

In our case the following "word" was not a word but a question mark, so that the function ANALVETA took over and completed the analysis by producing the resulting list of features for the node VH, printing the labelled tree-graph of the analyzed clause and its labelled bracketing. The features produced for the node VH were the following:

(QW IND FUT NIL).

QW was placed on the list of features of VH because of the presence of PRONQ in the subject NG and the question mark at the end of the caluse. It indicates that the analyzed clause was a Wh- question. IND denotes the indicative, FUT the future tense, and NIL the absence of negation. (The distinction between VGs with negation and those without it has to be registered since it represents an important piece of information for the subsequent semantic analysis.)

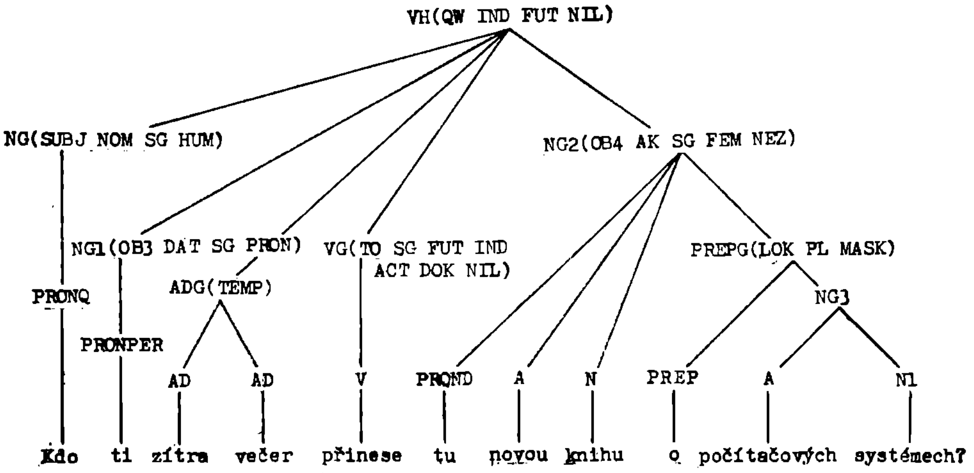


Figure 9

The syntactic tree of (1) and its labelled bracketing is diagrammed in Figures 9 and 10, respectively.

((((KDO) PRONQ) NG(SUBJ NOM SG HUM) ((TI) PRONPER) NG1(OB3 DAT SG PRON) ((ZITRA) AD (VECER) AD) ADG(TEMP ((PRINESE) V) VG(TO SG FUT IND ACT DOK NIL) ((TU) PROND (NOVOU) A (KNIHU) N ((O) PREP ((POCITACOVYCH) A (SYSTEMECH) N1) NG3(LOK PL MASK) PREPG(LOK PL MASK)) NG2 (OB4 AK SG FEM NEZ)) VH(QW IND FUT NIL))

Figure 10

As has been shown in Chapter VIII (p. 147), the output of the SNA in the form of a labelled tree-graph represents one of the two sources of input data necessary for the semantic analyzer. (The other source is represented by the semantic vocabulary, containing all the word-forms of the syntactic vocabulary, this time associated, not with lists of grammatical features, but with lists of possible type descriptions and further items of information concerning quantifiers, logical connectives and other meaning postulates.) Although the function of the semantic analyzer will be described in a separate paper, we intend to devote the last paragraphs of the present paper to a very short — and, consequently, very general — account of the semantic analyzer in order to show the leading principle of its activity as well as its close connection with the output of SNA as described above.

The semantic analysis starts with testing the nodes of the syntactic tree and the lists of features attached to them, because they contain the relevant data revealing the nature of the nodes and the internal structure of the analyzed clause. After the completion of these tests, there is a standard way of introducing the first semantic node $o(\sigma_1)(\mu)$ (the type of a t-proposition, see p. 133) and its obligatory daughters λt, λw, $o(\mu)$, o.

The next step is the analysis of VG. The semantic analyzer retrieves the first type description of the notional verb included in the syntactic node(s) of VG from the semantic vocabulary and establishes it as the node VG of the semantic tree. (If it is found that the analyzed clause contains a VGB (a verb group consisting of být i [to be] and a nominal complement), the semantic node is automatically associated with the type $o(\iota)(\sigma_1)(\mu)$ or $o(\iota,o)(\sigma_1)(\sigma_1)(\mu)$.) Then the adverbs of measure and those of manner connected with the verb are looked for. If found, they are attached to the node VG. By this, the interim analysis of VG is finished. (If in looking for the adverbs, an adverb of place is found, it is established as a new semantic node of the clause.) What follows is a preliminary analysis of NGs. It is tested whether the number of NGs constituting the daughter nodes of the syntactic node VH corresponds to the number of arguments indicated by the type associated with the semantic node VG. If the test is negative, the analysis is switched back and starts with a new analysis of VG, retrieving the next possible type description of the notional verb and proceeding in exactly the same way as described above till the number of NGs coincides with the number of arguments, i.e. till the above test is positive. It is only at this point that the semantic analysis of VG is definitely finished, and a detailed analysis of NGs can start.

The semantic analysis of NGs is based on the bottom-to-top procedure. The individual word-forms constituting the respective NG are retrieved from the semantic vocabulary together with their type descriptions. By means of the type grammar (containing the algorithm of application) the respective semantic nodes are gradually built up and are associated with the corresponding types till the resulting type of the node NG is established. This bottom-to-top procedure is subsequently applied to all the NGs constituting the daughters of the syntactic node VH. If their types match the type description associated with the semantic node VG, the results of the top-to-bottom and bottom-to-top procedures are put together in the form of the respective semantic tree. The whole semantic analysis ends with printing the tree in its linearized form, which is equal to the construction constituting the semantic representation of the analyzed clause.

REFERENCES

Chomsky, N. (1965). *Aspects of the theory of syntax* (Cambridge, Mass.).

Church, A. (1940). A formulation of the simple theory of types, *The Journal of Symbolic Logic* 5.56-68.

Church, A. (1956). *Introduction to mathematical logic I* (Princeton, New Jersey).

Čihánek, P. (1976). The semantic analyzer for Czech, *Papers at the Conference on Cybernetics* (Prague).

Firbas, J. (1964). From comparative word-order studies, *Brno Studies in English* 4.111-28 (Prague).

Firbas, J. (1966). Non-thematic subjects in contemporary English, *Travaux linguistiques de Prague* 2.239-56 (Prague).

Frege, G. (1892). Über Sinn and Bedeutung, *Zeitschrift für Philosophie und philosophische Kritik* 100. 25-50.

Kolář, J., Müller, K. (1974). *Programming language TESLA LISP 1.5*, a mimeographed manual (Brno).

Machová, S. (1976). Experiments with random generation of Czech sentences, *Papers at the Conference on Cybernetics* (Prague).

Montague, R. (1968). Pragmatics, in *Contemporary philosophy I* (ed. K. Klibansky) (Firenze, La Nuova Italia).

Montague, R. (1974). Proper treatment of quantification in ordinary English, *Formal philosophy* (ed. R. H. Thomason) 247-70, Yale University Press.

Morris, Ch. (1946). *Signs, language and behaviour* (New York).

Palová, I. (1976). The syntactic analyzer for Czech, *Papers at the Conference on Cybernetics* (Prague).

Sgall, P. at al. (1969). A functional approach to syntax, in *Generative description of language* New York, Elsevier).

Stenius, E. (1973). Syntax of symbolic logic and transformational grammar, *Synthese* 26.57-80.

Svoboda, A. et al. (1976). An ordered-triple theory of language, *Brno Studies in English* 12.159-86 (Brno).

Tichý, P. (1976). *Introduction to intensional logic*, a manuscript (University of Otago).

Winograd, T. (1972). *Understanding natural language* (Academic Press, New York and London).

Woods, A. W. (1969). *Augmented transitional networks for natural language analysis*, Report No-CS-1 (Aiken Computation Laboratory, Harvard University, Cambridge, Mass.).

Zwicky, A. M., Friedman, J., Hall, B. C., and Walker, D. E. (1965). The MITRE syntactic analysis procedure for transformational grammars, in *Proceedings of Fall Joint Computer Conference* (New York, Spartan) 317-26.

POKRAČOVÁNÍ TŘÍSLOŽKOVÉ TEORIE

Práce navazuje na stať „An Ordered-Triple Theory of Language“, která vyšla v minulém svazku tohoto sborníku, a je jejím rozvinutím i částečnou modifikací.

První oddíl je věnován převážně koncepci sémantiky. Tato koncepce je důsledně intenzionalistická, a proto se první kapitola zabývá kritikou extenzionalistického pojetí sémantiky, podle něhož jazykové výrazy označují tzv. extenze, tj. zejména individua, třídy, relace a pravdivostní hodnoty. Nutnost odlišného pojetí je zdůvodněna neintuitivními důsledky extenzionalismu (extenzionalistická analýza neumožňuje odlišit věty empirické od neempirických a porozumění od verifikace).

Druhá kapitola definuje základní pojmy intenzionální sémantiky budované nad souborem tří souborů: universa, souboru pravdivostních hodnot a souboru možných světů. Je aplikována tzv. jednoduchá teorie typů a jsou uvedeny nejtypičtější druhy intenzí. Ve třetí kapitole je zaveden pojem konstrukce (atomy, aplikace, abstrakce) a osvětlen vztah mezi jazykovým výrazem, konstrukcí a intenzí. Čtvrtá kapitola vyčleňuje definitoricky třídu tzv. jazykových konstrukcí ze souboru všech konstrukcí. Třída jazykových konstrukcí je míněna jako třída těch konstrukcí, které mohou být vyjádřeny výrazy přirozeného jazyka. V této souvislosti je možno chápat gramatiku jazyka jako soubor pravidel umožňujících odvodit ze stavby jazykových výrazů konstrukce, které tyto výrazy vyjadřují; jsou uvedeny jednoduché příklady takových pravidel.

Pátá kapitola se zabývá možností rozšířit soubor skládající se z universa, pravdivostních hodnot a možných světů o další soubory. Tak přidáme-li k uvedeným souborům soubor časových okamžiků, jsme s to provádět jemnější sémantickou analýzu zachycující i časové charakteristiky včetně gramatických časů. Podobně místní určení mohou být podrobena sémantické analýze, jestliže k uvedeným souborům přidáme soubor prostorových bodů.

Šestá kapitola je věnována rozboru úlohy indexických výrazů a stanoví vztah mezi tzv. vnější pragmatikou (tj. teorií těchto výrazů) a sémantikou. Sedmá kapitola se pokouší názorným způsobem zachytit jednak sémantické vztahy vyjadřování, označování a konstruování, jednak pragmatické vztahy demonstrování (u pragmaticky vnitřní) a determinování (u pragmatiky vnější).

Druhý oddíl představuje přechod od obecného rámce třísložkové teorie k její počítačové aplikaci. Osmá kapitola se zabývá srovnáním třísložkové teorie s funkčně generativním modelem jazyka P. Sgalla, který používá systému několika jazykových rovin. Na základě tohoto srovnání se ozřejmí místo morfologie, syntaxe i sémantiky v rámci níže navrženého systému jazykové analýzy založené na třísložkové teorii. Jsou zde též uvedeny důvody vedoucí ke konstruování počítačového modelu a zároveň je zkoumána otázka algoritmizace vztahů zavedených v třísložkové teorii.

Devátá kapitola zvažuje možnosti použití různých typů gramatik při strojové analýze češtiny. Dospívá se k závěru, že v současné době se jako nejvhodnější jeví využití bezkontextových pravidel realizovaných jako soubory procedur. Tato koncepce dala vznik procedurální gramatice češtiny, která je v kapitole stručně popsána. Tato gramatika se pojí na syntaktický slovník, koncipovaný jako soubor slovních tvarů opatřených seznamy gramatických rysů. (Výstup z tohoto slovníku je prakticky totožný s výstupem morfologického analyzátoru používaného v jiných koncepcích.)

Desátá kapitola obsahuje základní charakteristiku syntaktického analyzátoru vybudovaného na základě výše uvedené procedurální gramatiky a syntaktického slovníku. Je psán v programovacím jazyku LISP 1.5 přizpůsobeném potřebám počítače TESLA 200. Celkem jej tvoří 34 funkcí jazyka LISP. Popis funkce analyzátoru je spojen s příkladem syntaktické analýzy české věty. Výstupem analyzátoru je strukturní popis věty ve formě „labelled tree-graph“. Tento strukturní popis je připraven tak, aby mohl být zároveň vstupem jednak pro sémantický analyzátor, jednak pro zařízení stanovující postoje mluvčího v rámci vnitřní pragmatiky. V závěru je ještě zmínka o principu činnosti sémantického analyzátoru, který je ve stadiu ověřovacích testů.