II. Computational Linguistics II: Theoretical and Methodical Foundations
Computerlinguistik II: Theoretisch-methodische Grundlagen

4. Theoretical Framework, Aims and Concepts of Computational Linguistics

1. Introduction

The least one can say about computational linguistics (German: *Computerlinguistik,* French: *linguistique computationelle*) is that, as with other English nominal groups or compounds, the component parts of the expression have relations with each other: R. (computer, linguistics). The exact nature of this relationship is not, in the narrow sense of the word, a linguistic, in this case, morphological, matter. To specify the character of the relation one needs to know about what computers are like and what they can be used for, what linguistics is all about, what (computational) linguists are after and how they try to reach their aim(s) etc. ... Trying to answer the question of what is meant by computational linguistics therefore comes down to finding out the general framework it is to be situated in, the practice it entails, the conceptions it develops and the goals it aims at.

2. Theoretical Framework

As, by now, computers are becoming more and more part and parcel of our culture, there also is a growing knowledge among laymen about what computers are like. Any good general dictionary of a modern language reflects this situation. To quote just one of them: "A computer is a device, usually electronic, that processes data according to a set of instructions" (Collins Dictionary of the English Language, 1986 [2] , 325). With this, two important characteristics become clear: a computer is a (very general) data processing machine which can only carry out the processes meant if the latter can be properly instructed, i. e., if the latter are rule-governed, predictable, regular. In the following paragraphs we will have a closer look at the implications of this statement.

2.1. The Regularity Principle

If we take language to be a means to communicate with, two basic processes can be carried out with it, viz. speaking and understanding, or, put differently, writing and reading. In order to computerize these processes they need to be predictable or regular. The question then is what kind of rules language production/understanding is governed by. This first subsection will be restricted to production only. Language production rules can be represented as *condition—action rules:* given a set of one or more conditions (features, co(n)textual variables, constraints etc.), a set of one or more actions is undertaken (leading to the production of linguistic objects). From this point of view one can make a distinction between four types of (production) rules, viz. deterministic rules, optional rules, variable rules and statistical rules (cf. Frumkina 1963; Martin 1973, 1981).

2.1.1. Deterministic Rules

These are rules which, given a set of conditions, lead to a *fixed, predictable result* .

In German e. g. the following phonological rule is deterministic:

+ occlusive
+ voice → [— voice] / — #

Think of *Pferd* (horse) ~ *Pferde* (horses) *hab'* (have, 1 sing) ~ *haben* (have, pl.) etc. Deterministic rules are obligatory or categorial: they cannot be neglected without making a violation.

2.1.2. Optional Rules

Optional rules are rules which, given a set of conditions, can be either applied or not. This implies uncertainty as to the result obtained, not as to the correctness of this result.

Optional rules suggest a *random variation* in linguistic production. If one were free to choose to apply the T_{Adj} to "the girl with the blue eyes" the result could be either: "the blue—eyed girl" or, leaving the NP unaltered,: "the girl with the blue eyes". The homogeneity of linguistic production suggested by rules of the deterministic type is no longer (fully) guaranteed.

2.1.3. Variable Rules

As a reaction against the Chomskyan homogeneity abstraction (as expressed by rules of the first type and to a lesser degree also by rules of the second type) the concept of variable rules was introduced mainly under the influence of sociolinguists.

Variable rules are rules which capture the variable performance of the speakers of a language. Contrary to optional rules, variable rules deal with *a patterned variation:* the variable performance is seen as a function of linguistic and/or social parameters. A good example is to be found in Cedergren/Sankoff (1974) where the authors (who also present a computer program to deal with the phenomenon under discussion) illustrate the notion with examples taken from (Canadian-) French a. o. the way the *que* -deletion (as in "Au début je pense ça a été plutôt un snobisme" vs. "Au début je pense que ça a été plutôt un snobisme") should be treated.

2.1.4. Statistical Rules

Statistical rules no longer guarantee the correctness of the output or action. This is not due to our poor knowledge of the conditions under which the rule is to be applied but is taken to be an inherent characteristic of the linguistic rule system, i. e. a system where there is room for *a—systematic exceptions.* Instead of certainty statistical rules express the *probability* that the action undertaken will be correct. This does not imply that linguistic actions are totally irregular or unpredictable, but simply that they escape full determinism and are governed a. o. by preferences which can be expressed in a statistical way. In Dutch e. g. where many nouns are borrowed from English, rules should specify the gender of these loanwords (English and Dutch differ as to their gender system: English has no typical indication in the definite NP (always *the*), in Dutch there are two possibilities here: *de/het* —words). A rule such as

⟨+ English⟩
⟨+ noun⟩
⟨+ monosyllable⟩ → ⟨+ de —⟩

Then states that monosyllabic nouns which Dutch borrows from English will *by preference* be taken as *de* —words. In the case of *gag, smog, spray, hit, song, chip,* etc. this will yield a correct result (Dutch *de* — words), in the case of *team* and *script e. g.* this leads to a wrong result (in Dutch one should find het—words here). By means of this rule it is possible to state how new borrowings such as *snow* (from the drug scene), *rap* (taken from pop music) or *Aids* most probably but not necessarily will be used in Dutch (*the incurable Aids* e. g. will yield *het ongeneeslijke Aids*).

From what precedes it should have become clear that natural language processing by computer postulates regularity. Linguistic regularity however is neither an absolute nor rigid concept, but much more to be situated on a sliding scale moving between such extremes as (complete) certainty and (complete) uncertainty. Although the latter does pose serious problems to the computational linguist, linguistic usage can be said to obey certain *expectancy patterns,* entailing a lot of *redundancy* . The latter for sure will ease the task of the decoder. The encoder however may also choose to reduce this redundancy which will in turn create a second problem, viz. that of expansion.

2.2. The Expansion Problem

When we tackle language from the point of view of understanding, we notice that the computational linguist is confronted with what we will call the expansion problem.

Indeed, it is not necessary for the speaker to keep up with all the redundant features the language system offers him. Instead of doing

so, he may choose to reduce this redundancy. The consequence for the hearer will be that he has to expand the expressions he is confronted with again in order to understand them. To do so three kinds of techniques can be used: constraints, preferences and abductions.

2.2.1. Constraints

Suppose you are reading the following text: "John and Mary were walking in the street. All of a sudden *he* told her how much he loved her." It is evident that the underlined word "he" refers to *John* because of the fact that *he* can only refer to a referent which is male and not to a female one. As the text in question contains only one such referent we can infer with absolute certainty that *John* is the referent referred to by *he*. So, making use of constraints in comprehension is comparable to making use of deterministic rules in production: constraints must be obeyed, they can't be neutralized without breaking the rules.

2.2.2. Preferences

Preference is a technique which people use when they have to make a choice between several possibilities they know beforehand. In doing so the basic strategy is as follows:

— first a list of possibilities is drawn up

— then these possibilities are ordered according to preference or probability of occurrence

— finally the possibilities are tested starting with the most probable (preferred) one—this possibility is called the *default value* .

If this default solution can stand the test (no counterindications, no constraints which are violated) then it is preserved and no further searches are made. In the other case the second possibility is taken up and tested. The procedure is repeated until a solution is found (possibility without violation of constraints).

Suppose we hear a sentence as "Hans bringt Geld zur Bank" (Hans brings money to the bank/bench). We will take this to mean: "Hans brings money to a financial institution" unless we will find (syntactic, semantic, logical etc.) constraints which prove that the hypothesis taken up is false (e. g. in the context of a kidnapping affair where a ransom should be paid, it could become apparent that Hans is bringing money to a bench in the park.) This kind of *default reasoning* is comparable to the application of variable rules as described in sect. 2.1.3. Only, the constraints taken from these rules are arranged in a preferential order now and are used in a decoding context.

2.2.3. Abductions

Preferences are ideal when one element is to be chosen from a closed set of elements. If however the set we have to choose from is no longer closed but open (if we can no longer enumerate the possible solutions), we can not count on absolute certainty any more (compare to statistical rules in par. 2.1.4.). Suppose that we want to understand the following text:

"The dog ran into the room. It was wet through". When reading this text certain expectancy patterns will be activated. We will e. g. try to find an explanation for the fact that the dog is wet. The most plausible explanation could be for it to be raining outside, but this is not necessarily so. It could be that the dog has been running through a pool of water, or that it has fallen into a pond, or that the neighbour has been spraying it with a hose etc., etc. The possible explanations are not all equally probable, we could order them preferentially therefore, however, as with statistical rules, it is impossible to enumerate the set of possibilities exhaustively. We therefore will restrict ourselves to choosing the most probable solution.

The technique of abduction is often found in expert systems as a form of causal explanation:

if X is a possible cause of Y
and X is true
then X is cause of Y

(for an extensive discussion of abduction, cf. Charniak/Mc.Dermott 1985).

2.3. Interim Conclusion

If linguistics is taken to be the study of speaking and understanding of natural language then the foregoing should have been made clear that problems which are central for linguistics are even the more so for computational linguistics. In as far as the latter aims to be operational or *computable,* it indeed relies heavily on regularity and rules—rule-governed behaviour being a conditio sine qua non for the computerization of processes —. On the other hand if computational linguistics ever wants to come to grips with language understanding, it should solve the expansion

problem. As it has been stated up till now linguistic *processes* are taken to be central for computational linguistics, the computerization of these processes both leading to *new problems* and *new solutions* . It may however be worthwhile to leave this theoretical framework for a while and have a look at what in *actual practice* is done by computational linguists.

3. Computational Linguistics Practice

We will try to survey the field of activities by following a double approach. In the first instance we will describe the field extensionally, thereafter we will give an intensional description (cf. also Bátori 1977).

3.1. Extensional Approach to Computational Linguistics

One could define computational linguistics as a cover term for all work carried out by computers in solving problems set to them by linguists. This extensional approach to computational linguistics is one that is often found back at so-called CL-congresses, meetings, colloquia and the like. The result of it leading to a very heterogeneous and disparate body of research. This has mainly to do with the fact that in their practice linguists can and do use computers in quite different ways. We will discern four types of computer uses, viz. the use of computers as

— classifying machines
— calculating machines
— rule testers or control machines
— simulation machines

3.1.1. The Computer as a Classifying Machine

One of the fortes of computers is that they are very *general* machines. As a matter of fact they only work by means of programs, i. e. sets of instructions in which always new instructions can be defined making use of more primitive ones (which in the last resort are inherent to the programming language that is used). Using the computer as a classifying machine mostly means that the new instructions which one defines are, linguistically speaking, not relevant. After all, from a linguistic point of view, it does not matter whether we instruct a computer to alphabetize the words in a text either from left to right or from right to left. What matters for the linguist is to get a correct result, possibly in a rapid way. The algorithm as such is less important for him as a linguist. One could say that using the computer as a classifying machine often implies that the machine only makes use of formal (mostly graphical) linguistic knowledge and that, as a consequence, the linguist is result-, not process- oriented. The results that can be obtained this way have become routine: starting from "raw" text files the files are reordered so to get:

— indices (= alphabetical lists of words in texts + their location)
— concordances (= indices + context)
— KWIC — indices (= concordances + key-word at fixed place)
— word frequency-lists (= lists of words in texts arranged according to frequency)
— reverse word lists (= list of words in texts alphabetized from right to left)
— word length lists (= lists of words in texts arranged to number of characters)
etc., etc.

In other words, the text files are no longer static card files (representing one possible ordering of the texts), but dynamic data banks (which can lead to several orderings). However, in order for a linguist to maximally exploit this dynamism it seems important to organize the files he is working with as well as possible. This could give a new impetus to corpus linguistics on the one hand, and to the development of linguistic databases (such as lexica, thesauri, conceptual dictionaries) on the other. In corpus linguistics large bodies of texts are used as an important means to base linguistic description upon. The least one can say is that this approach could greatly profit from computerization in that it should be forced towards a twofold structuring: a text — external (texts as *factual databases*) and a text — internal one (texts as *textual databases*). The former viewpoint sees a text as an object related to other objects (texts), the latter sees texts as complex structured objects escaping traditional relational database descriptions and techniques. If this goal is to be reached the computational linguist will no longer restrict the role of the computer to that of a mere formal classifier, but use it in some other ways as are dealt with in what follows.

3.1.2. The Computer as a Calculating Machine

As computers were first used by mathematicians it seemed quite obvious that, applied to other sciences, the latter should treat analogous problems. As far as linguistics was concerned this led at first to the use of computers

by linguists as number crunchers. Mathematicians using computers were taken to be dealing with numbers and consequently that was what linguists should do also. This practice has led to counting phenomena such as graphemes and grapheme clusters (n-grams), words and wordgroups (collocations), and to applying techniques borrowed both from descriptive (such as the drawing of frequency tables and diagrams) and from inferential statistics (such as the use of significance tests, variance and factor analysis etc.).

For sure this has led to results which were otherwise difficult to obtain, but again the quantitative linguist or the scholar interested in style were far more interested in the results than in the processes underlying them.

Yet it seems quite evident that human beings also *process* numerical data: frequency about linguistic phenomena such as lexical items e. g. is not something which is innate but something which a. o. we abstract from exposure. These abstractions do not lead to a kind of exact probability figures but to such "intuitive" concepts as: very frequent, frequent, neutral (more or less frequent), less frequent, rare etc. Investigations in this direction (and the same goes for collocational studies) would move away from mere number crunching (numerical description) towards number processing (simulation of subjective processing of objective numerical data).

3.1.3. The Computer as a Control Machine

In what preceded we have demonstrated that in using the computer both as a classifying and as a calculating device the work of the linguist as linguist (or philologist, or student of style) usually, though not necessarily, starts *after* he has got the computer results.

If, in contrast with what has been said up to now, the researcher is no longer satisfied with mere observations as an output, but also wants to check them with previous knowledge or hypotheses, the computer is used as a control machine and the linguist's work as such starts *before* he gets the computer to work. Of course the kind of hypotheses tested may vary strongly but it remains a fact that the relevant features of the work to be done are not so much governed by the capabilities of the computer, but rather by the strategies followed by the researcher. To give a very simple example; Suppose we want to check a morphological rule of English which states that Adj + ly ⇒ Adv. Given a list of adjectives, this rule or statement can be easily tested. Of course it soon will become clear that the rule above must be shaded, otherwise it will generate such items as: *easyly, dailyly, comfortablely, duely, basicly, difficultly, goodly,* etc. The same could be done for the semantic contents (definition) of the linguistic objects.

In the context of a monolingual dictionary (some) run — on entries in — *ly* could be defined by implementing the following algorithm for creating definitions for adverbs derived in *-ly* (Evens/Vandendorpe/Wang 1985, 84):

IF THE MAIN ENTRY IS AN ADJECTIVE X
THEN CONSTRUCT DEF: IN AN X MANNER
ELSE IF THERE IS AN ADJECTIVE Y EARLIER IN THE LIST
OF DERIVED WORDS
THEN CONSTRUCT DEF: IN A Y MANNER
ELSE IF THE MAIN ENTRY IS A NOUN Z
THEN CONSTRUCT DEF: IN THE MANNER OF A Z
ELSE IF THE MAIN ENTRY IS A VERB W
AND THE ADVERB ENDS IN -INGLY THEN CONSTRUCT DEF: IN A WING MANNER

As one can notice the attitude of the investigator now is rather different from that outlined in paragraphs 3.1.1. and 3.1.2.: it is no longer (only) the results, the output, that interests the linguist, the algorithms used become of primary importance as well. In as far as the testing does not merely lead to putting on a par *computational linguistics* with *linguistics + implementation,* in order to put certain hypotheses to the test) we can even speak of the use of computers as simulation machines.

3.1.4. The Computer as a Simulation Machine

From the preceding paragraph it must have become clear that computers are a kind of ideal rule testers: given a set of rules one can empirically evaluate the observational adequacy of the latter by implementing them and judging the output yielded. If however one really wants to simulate human linguistic behaviour it will become evident that one cannot restrict oneself to rules which merely ex-

press declarative knowledge (as is traditionally done in most grammars): in order to process data one does not only have to *know what* objects are, one also needs to *know how* to handle and discover them. In other words, knowledge is not enough, know how, procedural knowledge, is needed as well.

To take up the example just given in the paragraph above: when looking up lexical items in a dictionary human beings know how to interpret "effusively" in the following context:

effusive (ɪ'fju:sɪv) adj. 1. extravagantly demonstrative of emotion; gushing 2. (of rock) formed by the solidification of magma.

effusively adv. — effusiveness n.

In order to extract this information from a text such as the one above one needs knowledge of different kinds such as morphological knowledge stating the relationship between adjectives and (derived) adverbs, next to syntactico-semantic knowledge specifying the role of adverbs in a sentence, the kind of constraints governing the relation between verbs and adverbs (preventing e. g. *effusively* from being interpreted in the second meaning of *effusive*), next to that one needs knowledge of the (dictionary) world stating the difference between the several kinds of objects to be found, the way information is represented etc. Above all however one needs to know how to find out about these different kinds of knowledge.

Transposed to the simple context of "effusively" being used in a sentence such as: "She thanked us effusively", in order to understand *effusively* correctly, it does not suffice to know what an adverb is but also to know how to find out that *effusively* is an adverb even if one has never heard using it before.

In other words, it is not sufficient to find in the Lexicon of a "Computational Grammar" that *hit* is a verb with the case frame [0 (A) (I)] (the verb *hit* occurs in sentences with a noun phrase in the objective case, and optionally noun phrases in agentive and instrumental cases) (cf. Harris 1985), one should also find how to find out about NP's, about their role in the sentence and, to start with, how to find the verb in a simple sentence as "The hammer hit the nail" (where, from a purely formal point of view, there are three possible candidates for the main verb function in this sentence).

One approach (cf. Martin/Platteau/Heymans 1986) could then be to assign default categories and direction pointers to lexical items so that these items, certain conditions being fulfilled, can shift to other categories. Such a system would contain e. g. the following information:

	Default Category	Direction Pointer
— hammer	n.	left
— hit	v.	right
— nail	n.	left

(meaning that items as *hammer* and *nail* e. g. are basically nouns which shift to verbs when they are put in the object or instrument slot of the verb frame)

A rule such as

```
if
        (thiscat    =    noun)
and     (thisdir    =    left)
and     (prevcat    =    det)
and     (prevdir    =    neither)
then
        thisdir:    =    neither;
```

would change the direction pointers of *hammer* and *nail* e. g. so to keep them in the class of nouns.

The process that we have briefly sketched above is known as *tagging* and it should be clear from other well-known topics such as *parsing, paraphrasing, abstracting, indexing, translating, man-machine dialogue,* etc. that the interest of the computational linguist in these cases does not only lie in the result obtained, but at least as much in the method, the procedure, to obtain this result, In this way computational linguistics becomes *process*- instead of *result* -oriented.

3.2. Intensional Approach to Computational Linguistics

The use of the computer as a simulation machine brings us on the one hand back to the beginning of this article where we dealt with the theoretical framework computational linguistics is to be situated in and on the other hand it brings us to the end of it in that we try to give an intensional definition of computational linguistics.

In such a definition it is the inherent characteristics of the object under scrutiny which play a predominant role. In the case of a scientific branch or subdiscipline one may expect that the discipline will be defined by its domain, scope or the object(s) it tackles and/or by the method(s) it uses to do so.

During the preceding, extensional, approach it must have become clear that a defi-

nition of computational linguistics as linguistics *with* a computer is of course not wrong at all, but not very illuminating as well. However in discussing the use of computers as simulation machines in linguistics, it must have become apparent too that next to linguistics *with* a computer, linguistics *for* a computer is possible also. This is particularly the case when the researcher has to formulate language rules in such a way that a computer can also "understand" them. More simply stated this means that the computer is programmed so that it can read and write natural language as human beings can, i. e. "understanding" what it is doing.

Basically computational linguistics is defined then as *that branch of linguistics that studies linguistic processes by simulating them by computer* or what Bátori (1977a) has called "die Beschreibung der Sprache als Prozeß". As we have seen at the beginning (section 2: theoretical framework) these processes ultimately come down to speaking and hearing so that one also could claim that the basic model underlying computational linguistics is that of an abstract question—answering system in which there is a linguistic interaction between man and machine in such a way that the machine can "understand" the problems it is confronted with and "offer a solution" to them (cf. Ungeheuer 1971). (As computational linguistics simulates this linguistic reality we have put the computational understanding of and solutions to linguistic problems between inverted commas, implying that the latter (the simulation) may differ from the former (reality) but nevertheless is relevant for it).

4. Final Conclusion

One of the goals of this article was to demonstrate that in order to reach the aim of computer simulation of linguistic behaviour not only a new field of problems was "discovered" (process linguistics), but new approaches to solving these problems as well (cf. the discussion of the methodological apparatus implying rules, preferences, procedures, abductions and the like).

On the other hand we also wanted to make clear that an open, dynamic definition of object and method in computational linguistics is the most realistic and fruitful one: computational linguistics is then seen as an autonomous field within linguistics with at its *core* an own object (linguistic processes) and an own methodological apparatus (rules, procedures, preferences, abductions, etc.); however, this core can be extended and move towards a periphery, these *extensions* in their turn having a *core* and an *extension* etc., etc. By defining computational linguistics in this *recursive* way one can go beyond such dichotomies as pure and applied, theory and practice, description and simulation, models and tools.

5. Literature (selected)

I. Bátori 1977a · H. Brandt Corstius 1974 · E. Charniak/D. McDermott 1985 · H. J. Cedergren/D. Sankoff 1974 · M. Evens/J. Vandendorpe/J.-C. Wang 1985 · R. M. Frumkina 1963 · P. Garvin 1962 · M. D. Harris 1985 · W. Lenders/G. Willée 1986 · W. Martin 1973 · W. Martin 1975 · W. Martin 1981 · W. Martin/F. Platteau/R. Heymans 1986 · J. A. Moyne 1977 · G. Ungeheuer 1971 · T. Winograd 1983.

Willy Martin, Amsterdam (The Netherlands)/ Antwerp (Belgium)

5. Problembereiche der Computerlinguistik: Positionierungs- und Abgrenzungsaspekte

1. Einleitung

In den letzten 35 Jahren hat sich ein Forschungs- und Wissenschaftsgebiet entwickelt, in dem als besonderes Forschungsinstrument die elektronische Datenverarbeitung zur Erforschung von Sprache eingesetzt wird: die *Computerlinguistik* (CL). Das vielfältige Spektrum und die verschiedenen For-