GENERAL PROPERTIES OF NATURAL LANGUAGE PROCESSING SYSTEMS

Luc Steels
University of Antwerp (U.I.A.)
Universiteitsplein,1
B-2610 Wilrijk-Antwerpen
Belgium

In this paper we investigate some properties of natural language processing systems from a general systems point of view. It will turn out that in order to be completely successful, such systems should have all the characteristics of OPEN SYSTEMS, this means contact with an environment, equifinality, goal-directedness, self-organization, maintenance of a steady state, continuous adaptation to the environment, etc... . We will mention some conseqeunces of these observations for the design of linguistic systems and give examples if methods for solution already exist.

In recent years, there has been a considerable effort to design systems that produce or understand (a limited set of a) natural language. Some examples can be found in Minski (1969), Simon and Siklossy (1972), Schank and Colby (1973), a.o. . In this paper we examine the general picture of a natural language processing system that is emerging from these studies. For similar reflections see Newell (1975), and the section on systems organization in Reddy (1975).
First we deal with the general outlook of an (ideal) natural language processing system, then we discuss various properties of each aspect and finally try to identify the general character of the system.

1. GENERAL OUTLOOK

It is intuitively clear that language processing systems of the type humans use, receive input from a (linguistic) environment and return output to this environment, thus constituting the environment.
It is also clear that the whole process, which is actually a mapping

from physical signals into information structures and back, can be split up in a series of subsystems, which all contribute to the main task, which is to understand or produce natural language. Examples of such subsystems (or knowledge sources) are:

1. Routines to accept input from the environment either in a graphic or acoustic form.
2. Lexical analysis/synthesis, i.e. dictionary lookup including phonemic rewriting and orthographic decoding.
3. Morphological analysis/synthesis: discovery of morphemes.
4. Syntactic analysis/synthesis: extraction/construction of syntactic structures.
5. Semantic analysis/synthesis: discovery of semantic relations, resolution of word sense ambiguity, decision of concepts to be used in production, etc... .
6. Cognitive processing: storing new information and/or retrieving information from memory, consultation of world knowledge, consultation of pragmatic knowledge of situation and speaker.
7. Means of outputting the language expressions by graphic or acoustic media.

It turns out that each subsystem consults information, which consists for example of a dictionary of the language within a system for lexical analysis, a grammar of the language within a system for syntactic analysis, a memory containing information about the world on the level of cognition, a psychological model of the speaker in a subsystem contributing pragmatic knowledge, etc... .This information is also considered to constitute a system and in this context we will call it the underlying system.

Together with an underlying system, each subsystem needs a control function specifying how the underlying system should be consulted, that is how it can be put to use. This control function largely depends on what particular task the subsystem is trying to achieve. On the level of syntactic analysis e.g. such a control function is called a parsing mechanism.

Given all these subsystems, it is clear that there should also be a general control function controlling the operation of the distinct subsystems, and a channel by which the different subsystems all communicate with each other. This channel is a medium to represent partial knowledge.

Summarizing:

(a) a linguistic whole system consists of
    (1) an overall control structure
    (2) a medium to represent partial knowledge
    (3) a set of subsystems

(b) a linguistic subsystem consists of
    (1) a control function
    (2) an underlying system

Let us now discuss in some more detail some general aspects of these various components.

2. ASPECTS

2.1. UNDERLYING SYSTEMS

As we said before the underlying systems contain the information that is being consulted by the control function of a particular subsystem. The main problem to construct underlying systems is that they should have an infinite capacity , that means they should express an infinite set of information by finite means. Intuitively it is clear that we can speak about everything we want to speak about, apply an infinite range of syntactic structures, make inferences about the information contained in the memory, etc... .

The normal way of solving this problem in linguistic practice is the following :

(i) select certain basic units and define the systems activities for them,

(ii) specify systems activities of other units in terms of previously defined units.

This 'modular' way of thinking stems from structural linguistics, a special case of it called recursion is the use of a unit itself in the definition of that unit. Recursion proved to be the main tool in our dealing with the infinite capacity of language by finite means.

As an illustration of this principle we discuss very briefly the notion of a grammar. A grammar is a quintuple $G = \langle Vn, Vt, P, S \rangle$ containing (i) a finite alphabet V, which is split into two subsets an alphabet of terminal symbols (Vt) and an alphabet of nonterminal symbols (Vn), (ii) a finite set of rules or productions $P \subseteq V^{\star} \times V^{\star}$ of the form $\gamma \rightarrow \delta$ and (iii) an axiom or start symbol $S \in Vn$.
An example of a grammar is $G = \langle \{S\}, \{a,b\}, \{S \rightarrow a\ S\ b,\ S \rightarrow a\ b\}, S \rangle$
(Note that in the first production S is defined in terms of itself)

A control structure to get a grammar at work (but not the sort of control structures used in a natural language processing system) is the generation relation $\Rightarrow$, defined as follows: $\sigma \Rightarrow \phi$ iff $\sigma = \sigma' \gamma \sigma''$, and $\phi = \sigma' \delta \sigma''$ and $\gamma \rightarrow \delta \in P$.
The relation is generalized by its reflexive and transitive closure $\overset{\star}{\Rightarrow}$.
Starting from a certain (nonterminal) element, the axiom, one can then 'generate' by consecutive application of this relation strings in a language. The output of the system are the sentences of the language defined as follows: $L(G) = \{x \mid x \in Vt^{\star}, S \overset{\star}{\Rightarrow} x\}$.
For the example given the language generated is $\{a^n b^n \mid n \geq 1\}$, and this is indeed an infinite language as the reader may verify.

It should be stressed that there are many other forms an underlying system can take. One form very much used in the field is the network (e.g. for syntax the transition networks as introduced by Woods (1970)), and there are still new developments under way.

The second problem that is to be faced with is that the underlying systems are clearly open to further extension or modification. That means any language user has the ability to define new words, add new constructs, talk about new things and about old things in new ways. In other words an underlying system should be extendable.

At some levels (e.g. dictionary) this extendibility is no great problem, at other levels however this seems a tremendous task, particularly the problem of grammatical inferences or inductive inference on a semantic level. The difficulty is partly stemming from the fact that the systems are self-organizing that means they form themselves by a continuous contact with the environment.(This viewpoint is in contrast to the Chomskyan view on language acquisition which maintains that the systems are innate)

Indeed, as we mentioned right at the beginning linguistic systems receive input from the environment; this continuous input can be considered to be the basis of the systems outlook, simply because the system should be such that it is able to process the input, else it fails. After a growth period (e.g. during childhood) in which the system adapts itself to the environment, the system can be said to maintain a steady state , that is the main character of the system remains the same, due to the fact that the input does not change. This does not mean however that we deal with a purely static or closed system. The whole system is continuously being influenced by the environment. A consequence of this should be that at the moment when a particular environment is no longer present, the systems formed in order to deal with this environment should faint away. And this is indeed the case, someone who learned a foreign language and who is not able to practise this regularly, will soon find out that his knowledge is decreasing.

These observations have a direct consequence for the learning of a (foreign) language. Instead of learning the structures of the language explicitly, one should let the language learner himself discover what the underlying systems are by bringing him into the appropriate environment and providing the necessary stimuli to be active in this environment.

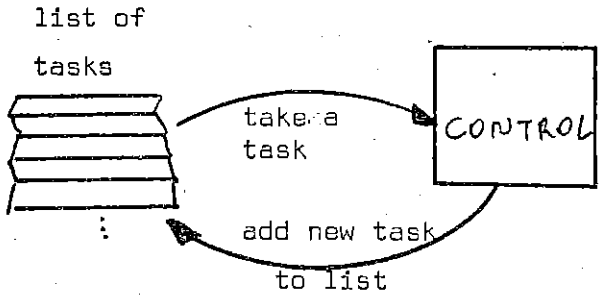
Interesting experiments to design natural language processing systems with learning behavior can be found in Vigor, Urquhart and Wilkinson (1969), Siklossy (1972), Anderson (1975).

2.2. CONTROL FUNCTION IN SUBSYSTEMS

Now we turn to the mechanisms that put the information embodied in the underlying systems to work. Of course the control function depends on what particular sort of knowledge source is involved.
The main problem with control functions is the seemingly very inefficient thing that the same datum can be used for different purposes (e.g. the same form of a word can have different meanings.). This is the principle of nondeterminism . In general a system is nondeterministic if at certain points there are several alternatives available and there is no means to see immediately what alternative should be followed. The basic cause of this is the fact that in the underlying system, a unit has been defined in more than one way.

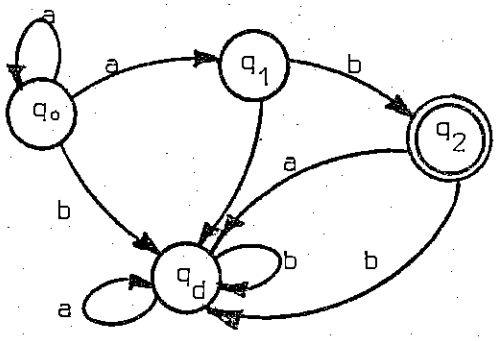
We deal with nondeterminism by means of a task-oriented control function. This goes as follows: (i) first consider primitive activities and define for each activity what particular minimum input information it must have in order to be performed. Such a sequence of minimal information is called a task or configuration.

(ii) define the control function such that the execution of a task will lead to the creation of other tasks.The execution of a series of tasks should lead to the desired result. Schematically:



We now give an example of a very simple task-oriented system which consults as data a finite automaton, input are language expressions, output is a decision whether the expression belongs to the language defined by the automaton or not.
Let $\mathcal{A} = (K, \Sigma, \delta, q_0, E)$ be a nondeterministic finite automaton where $K = \{q_0, q_1, q_2, q_d\}$ is a final set of states , $\Sigma = \{a,b\}$ is a finite alphabet, q_0 is the initial state, $E = \{q_2\}$ is the set of final states and δ is the transition function given by the following state diagram.

$\delta(q_o,a) = \{q_o, q_1\}$ $\quad \delta(q_d,a) = \{q_d\}$
$\delta(q_1,b) = \{q_2\}$ $\quad \delta(q_d,b) = \{q_d\}$
$\delta(q_1,a) = \{q_d\}$ $\quad \delta(q_o,b) = \{q_d\}$
$\delta(q_1,b) = \{q_2\}$



Clearly when we start in q_o and we go through a series of states till we reach the final state q_2, then we obtain strings of the form $a^n b$ for $n \geq 1$. Q_d is a dead state (or garbage state), when one gets there, it is no more possible to reach a final state. Nondeterminism is occurring when being in q_o and with 'a' as input.

Now we construct a task-oriented control function which consults $\mathcal{Q}$ as the underlying system. A task t_i is a pair $t_i = \langle \mu_{i,1} , \mu_{i,2} \rangle$ where $\mu_{i,2}$ is the string to be processed and $\mu_{i,1}$ is the current state at the moment of task execution.

Now the control function $\mathcal{S}$ runs as follows, t_j is the new task being produced. Let $\sigma = \mu_{i,2}$, $\sigma = a\,\sigma'$ with $\sigma' \in V^*$ and $a \in V$, then

$$\mathcal{S}(t_i)(Y) = \begin{cases} \sigma' & \text{for } Y = \mu_{j,1} \\ \delta(\mu_{i,2}, a) & \text{for } Y = \mu_{j,2} \end{cases}$$

The initial task is $\langle q_o , \sigma \rangle$ with q_o the initial task and σ the input word, a valid end task should be $t_v = \langle q_f , \lambda \rangle$ with $q_f \in E$ and λ (the null string) . For the string 'aaab' this would go as follows:

$$\langle q_o,aaab \rangle \begin{cases} \langle q_o,aab \rangle \begin{cases} \langle Q_o, ab \rangle \begin{cases} \langle q_0,b \rangle \longrightarrow \langle q_d,\lambda \rangle \\ \langle q_1,b \rangle \longrightarrow \langle q_2,\lambda \rangle \star \end{cases} \\ \langle q_1,ab \rangle \longrightarrow \langle q_d,b \rangle \longrightarrow \langle q_d,\lambda \rangle \end{cases} \\ \langle q_1,aab \rangle \rightarrow \langle q_2,ab \rangle \rightarrow \langle q_d,b \rangle \longrightarrow \langle q_d,\lambda \rangle \end{cases}$$

★This is the only valid end task, the rest is all ending up in a dead state.

Note that the order in which the tasks are executed is of no importance. If only one path of the tasks is executed till there is a break, and if the system then goes back to try other paths, we say that the control function uses backtrack.

The principle of nondeterminism gives rise to a situation where the same datum leads to different final results. The reverse is also true, we can from different data come to the same final result. Consider the fact that we are able to express the same information in so many different ways, e.g. by means of an active or passive sentence.

This observation has already been made on the level of syntax by structural linguists, and it was one of the main arguments to add a transformational component to generative grammar (see Chomsky (1957)). It turned out however that adding such a transformational component to a given grammar resulted in systems defining a class of undecidable sets (this was proved by Peters & Ritchie (1973)), an operationally more interesting solution are the transition networks of Woods (1970), which are essentially recursive transition networks with actions associated to the various transitions. A generally accepted solution to this problem has however as yet not been found.
This principle together with the principle of nondeterminism can be stated as follows: The systems reach their final states (i.e. completion of analysis) from different initial data and by a variety of paths.
This principle is commonly referred to as the principle of equifinality.

2.3. OVERALL CONTROL STRUCTURE

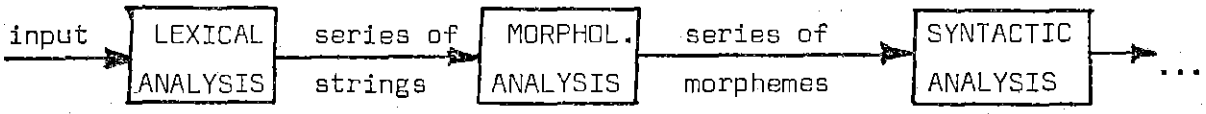
Humans have no difficulty with equifinality and nondeterminism. On the contrary, they can decipher spellings-errors, understand incomplete or syntactically incorrect sentences, start reading in the middle of a text, etc... . This aspect of human language behavior is often credited to the fact that people are intelligent, but what means being intelligent ? It seems that the main aspect of this, in the context of language, is the apparent goal-directedness of human language use.
This goal-directedness can partially be solved by the organization of the whole system in such a way that all levels interact: nondeterminism at one level can then be solved by more information about other levels, in other words alternatives can be chosen based on previous knowledge of the state of the system.

The phenomenon of goal-directedness is on the whole not understood very well, simply because we lack good knowledge of several subsystems (especially semantic ones), and therefore can not organize whole systems in complete interaction as yet.
It should be kept in mind that the main goal of a whole system is understanding or communicating, and not assigning nice structural descriptions of enormous complexity to language expressions.

Given the knowledge on underlying systems, it was thought that one could simply use on each level of analysis a system. This is illustrated by people working on machine translation :as a first step they worked from words to words (with an elaborated morphology of course), to obtain better results (that means to undue the nondeterminism at word level) a higher level of analysis, namely syntax, was taken into account and they worked from syntactic structures to syntactic structures. It turned out however that this method does not work: one cannot do a complete lexical analysis or a complete syntactic analysis without knowledge of the results of other levels of analysis. An example will illustrate this. Consider the sentence ' I saw the tree of the Zoo in Antwerp'. Normally a human person does not notice any ambiguity, but 'saw' is ambiguous, did I saw the tree or did I see the tree ? And also 'in Antwerp' is functionally ambiguous, consider 'I saw the tree of the zoo in Antwerp (on a postcard in New York).

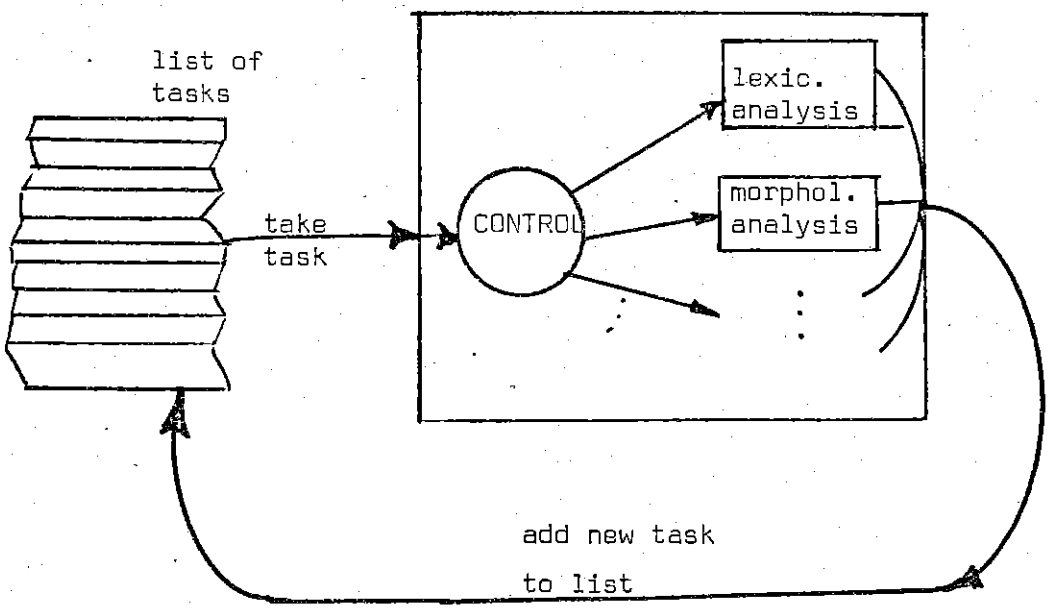
Clearly in order to be able to do analysis at a particular level, information of analyses at all other levels (including knowledge of previous text) should be available. The systems principle behind this can be stated as follows: All subsystems are in interaction.

So schematically the following interconnection of the subsystems:



where the output of the lower level of analysis is input to the higher level, will not do.

Instead we will have a task-oriented whole systemsorganization as for the subsystems themselves. The general control function examines a particular task to see of what sort it is, and sends it for execution to the different subsystems.



See for examples of task oriented systems the work of Woods (1973), and Kay (1974).

Although the principle of interaction is understood, another consequence of this principle is much harder to realize, namely the fact that ALL the parts interact, this means that we have to solve all questions (so even the semantic ones) before we can ever hope to have a good working system. When this point was realized by people working on machine translation they found this task so embarassing that they lost faith in an eventual solution to the whole problem (see e.g. Bar-Hillel(1964)). This is understandable if one considers all the different components of a language

2.4. REPRESENTATION OF INTERMEDIATE KNOWLEDGE

Another problem is how the representation of intermediate knowledge will look like. The solution to this problem is not quite clear. There have been efforts to construct task-oriented systems where the 'intermediate knowledge' was carried along by the tasks themselves, another strategy is to develop a 'neutral' representational device that is consulted by each task.The last solution has been taken e.g. in the G.P.S. see Kay(1973).
Particularly at higher levels of analysis the discovery of good representational devices is felt to be one of the major unsolved questions.

3. CONCLUSIONS

To summarize the discussion, we now give the mentioned properties of linguistic systems.

1) they receive input and produce output, Ie. they are in contact with the environment,

2) the whole subsystem is subdivided in partial systems and each partial system contains a control function and an underlying system.

3) the underlying system has a control function and an underlying system, the underlying system has a recursive organizational structure,

4) each partial system is qua organization task-oriented due to non-determinism,

5) each partial system (and as a consequence the whole system) is equifinal,

6) the whole system is goal-directed

7) the whole system is task-oriented due to the necessary interaction of all partial systems,

8) the whole system retains a steady state, this implies that it is constantly learning, i.e. adapting itself to the environment.

9) linguistic systems are self-organizing.

Contact with the environment, goal-directedness, interaction of subparts, equifinality, self-organizing, maintenance of a steady state, it is clear that natural language processing systems are OPEN system.
See for open systems in general Von Bertalanffy (1968, espec. 146-162), Katz & Kahn (1969), and others.

The insight that linguistic systems are open systems is at the same time embarassing and exciting. Embarassing because all our current efforts to construct systems for a purpose related to natural language processing will have to be even more complicated than one used to think. In the last decade it has become clear that at a semantic level world knowledge, inference mechanisms and even more sophisticated problem solving methods should be available in order to simulate understanding.

One of the main conclusions of our observations is that even if this is all available, there still remain many problems concerned with whole systems organization, especially for the self-organizing property of linguistic systems and their capability of learning.

At the same time the idea that linguistic systems are open systems is exciting, simply because all cybernetic thinking about living systems (e.g. feedback, or other control mechanisms) becomes available for solving linguistic problems.

Language systems are indeed living in the sense that they grow, reproduce themselves throughout generations of people, stay in contact and constitute an environment, etc... .

If one accepts the solution that natural language processing systems should be open system, it becomes clear that a solution to such problems as machine translation is not to be expected very soon.
Research to define the underlying systems, which has been the 'program' of structuralist schools, should be complemented with research on the respective control functions and on the whole system. The main problem in this is the simulation of goal-directedness, self-organization and adaptation to the linguistic environment.

4. REFERENCES

Anderson, J.R. (1975) Computer simulation of a Language Acquisition System. A first report, in: Solso, R.L. (ed) Information Processing and Cognition. The Loyola Symposium. Wiley, London. 1975.

Bar-Hillel, Y. (1964) A demonstration of the nonfeasibility of fully automatic high quality translation. In: Y. Bar-Hillel, Language and Information. Addison Wesley Publishing Cy. Reading, 1964.

Bertalanffy, L. Von (1968) General Systems Theory. Foundation Development Applications. Penguin Books, Harmondsworth.

Chomsky, N. (1957) Syntactic Structures. Mouton, Den Haag.

Katz, D. and R.L. Kahn (1969) Common characteristics of open systems. In: F.E. Emery (ed.) System thinking. Penguin Books, Harmondsworth.

Kay, M. (1974) Automatic morphological and syntactical analysis. Mimeo, International Summerschool for Mathematical and Computational Linguistics. Pisa.

Levelt, W.J.M. (1974) Formal grammars in linguistics and psycholinguistics . Vol II. Applications in linguistic theory. Mouton, Den Haag.

Minski, M. (ed.) (1968) Semantic Information Processing. The MIT Press, Cambridge, Mass.

Newell, A. et.al. (1973) Speech Understanding Systems. North-Holland Publishing Co. Amsterdam.

Peters, S.R. and S. Ritchie (1973) On the generative power of transformational grammars. Information Sciences. 6.

Reddy, D.R. Speech Recognition. Invited papers presented at the 1975 IEEE Symposium. New York . Academic Press.

Schank, R.C. and K.M. Colby (eds.) (1973) Computer Models of Thought and Language. W.H. Freeman and Co. San Francisco.

Simon, H.A. and L. Siklóssy (1972) Representation and Meaning. Experiments with Information Processing systems. Prentice-Hall, Inc. New Jersey.

Vigor, A., Urquhart, K. and H. Wilkinson (1969) PROSE parsing recognizer outputting sentences in English. In: Meltzer, B. et. al. (eds) Machine Intelligence 4. 271-285. University Press, Edinburgh.

Woods, W. (1970) Transition network grammars for natural language analysis. Communications of the ACM, vol 13, oct. 1970.

Woods, W. (1973) An experimental parsing system for transition network grammars. In: Rustin, R. (1973) Natural Language Processing. Algorithmics Press. N.Y..