Control and Cybernetics

vol. **22** (1993) No. 1/2

Mathematical model of natural language phrase generation

by

Jadwiga Ratyńska

Technical University in Radom,
Radom,
Poland

An algorithm of a grammatical processor of declension and conjugation is presented. It is a dynamic vocabulary generating the inflected forms of the metalanguage variable. Since these forms are not remembered for a longer time, economy of space is preserved.

1. Introduction

A set of algorithms for automatic language phrase recognition makes up a grammatical processor called declension–conjugation processor (D–C processor), Laus–Mączyńska (1984), Wetulani (1990). The D–C processor generates grammatical forms on line. Its main task is to compare the testing form of a metalanguage variable $W(n)$ with the grammatical form $W(x)$ generated by the D–C processor, Ratyńska (1990). In case of compatibility of these forms the mechanism stops its work. The scheme of the algorithm identifying the grammatical structure of language phrases is presented in Fig. 1.

2. Mathematical description of grammatical unit

The declension–conjugation processor consists of grammatical units responsible for generating grammatical forms of respective metalanguage variables. An analysis of grammatical unit allows to introduce the following definitions and the theorem.

In memoriam Professor Edmund Lipiński

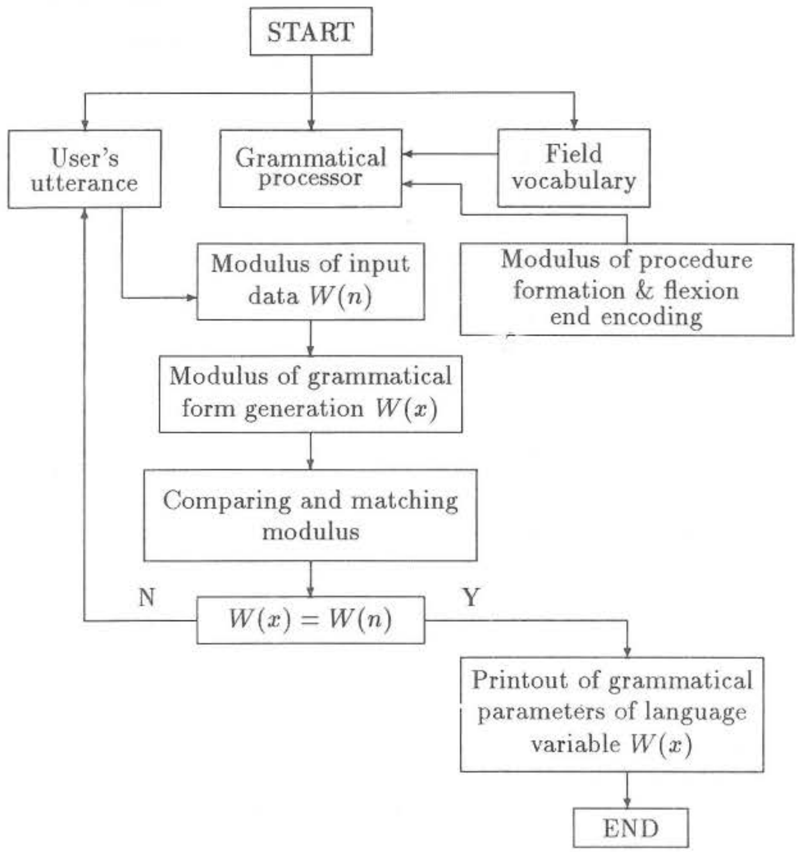

Figure 1. Scheme of grammatical identification algorithm

DEFINITION 1. *The following ordered four is called the grammatical unit :*

$$C = < S, SP, SK, H >$$

where

C	—	*grammatical unit*
S	—	*set of states*
SP	—	*set of initial states*
SK	—	*set of final states*
H	:	$S - SK \to 2^S$ — *function determining the sequence of transition to respective states* s_i

DEFINITION 2. *The transition* $l(s', s'')$ *from* s' *to* s'' *in grammatical unit* C *is called the sequence*

$$s_{l_1}, s_{l_2}, \ldots, s_{l_n} \quad \textit{such that} \quad s' = s_{l_1}, s'' = s_{l_n}$$

and for every

$$i \ \ (1 \leq i \leq n) \quad s_{l_{(i+1)}} \in H(s_{l_i})$$

DEFINITION 3. *The states* s', s'' *are assumed to be in relation of transition* l *which is determined by* $l(s', s'') \in L(C)$, *if there is a transition from* s' *to* s'' *within the grammatical unit* C.

DEFINITION 4. *The states* s_r, s_t, s_z *of grammatical unit* C *are in relation of limited sequensor* N *in the situation which can be described in the following way :*

$$(s_r, s_t, s_z) \in N(C) \qquad \textit{if there is a transition } s_{l_1}, \ldots, s_{l_n} \textit{ from } s_r \textit{ to } s_t \textit{ such that no } s_{l_i} \ (1 \leq i \leq n) \textit{ is equal } s_z.$$

The notion of predecessor can be introduced in a similar way.

DEFINITION 5. *The states* s_r, s_t, s_q *are in relation of limited predecessor* K *which can be described as follows :*

$$(s_r, s_t, s_q) \in K(C) \qquad \textit{if there is a transition } s_{l_1}, \ldots, s_{l_n} \textit{ from } s_r \textit{ to } s_t \textit{ such that no } s_{l_i} \ (1 \leq i \leq n) \textit{ is equal } s_q.$$

DEFINITION 6. *The subset of states* $N(s_q, s_z) \subset S$ *satisfying the following condition :*

$$N(s_q, s_z) = \{s^+ : (s_q, s^+, s_z) \in N(C)\}$$

is called the set of sequensors $N(s_q, s_z)$ *of state* s_q *with limitation* s_z *in a given grammatical unit* $C =< S, SP, SK, H >$, $s_q, s_z \in S$.

DEFINITION 7. *The subset of states* $K(s_q, s') \subset S$ *satisfying the following condition :*

$$K(s_q, s') = \{s^+ : (s^+, s', s_q) \in K(C)\}$$

is called the set of predecessors of state s' *with limitation* s_q *in grammatical unit* $C =< S, SP, SK, H >$, $s', s_q \in S$.

Definition 8. *The subset of states $KN(s_q, s_z) \subset S$ satisfying the following condition :*

$$KN(s_q, s_z) = \bigcup_{s' \in N(s_q, s_z)} K(s_q, s')$$

is called the set of predecessors of set elements $N(s_q, s_z)$ with limitation s_q in grammatical unit $C =< S, SP, SK, H >$.

Definition 9. *Let s_q, s_z be the states of grammatical unit C, $N(s_q, s_z)$ — the set of sequensors of state s_q with limitation s_z and*

$$Z(s_q, s_z) = \left\{ \begin{array}{ll} N(s_q, s_z) & \textit{if } (s_q, s_z) \in Z(C) \textit{ and } s_q \neq s_z \\ 0 & \textit{in other cases} \end{array} \right\}$$

If the set $Z(s_q, s_z)$ is nonempty, then the set is called the set of language production, s_q — input to the set, s_z — output from the set.

Theorem 1. *For every set Z, $Z = Z(s_q, s_z)$ in grammatical unit C, $C =< S, SP, SK, H >$, $Z \subset S$, every $s_r \in S - Z$ and every $s_t \in Z$, the following relation holds true :*

$$(s_t, s_r, s_z) \in N(C), \quad (s_r, s_t, s_q) \notin K(C)$$

Proof. It is assumed that $s_t \in Z$ and therefore $(s_q, s_t, s_z) \in N(C)$ takes place. The proof is carried out indirectly. If the first part of the thesis is negated then $(s_t, s_r, s_z) \in N(C)$. As the relation N is transitive, then $(s_q, s_r, s_z) \in N(C)$ is obtained. It means that $s_r \in Z$, which is not with agreement with assumption. Negation of the second part of the thesis results in $(s_r, s_t, s_z) \in K(C)$. Then taking into account definition 8, $s_r \in KN(s_q, s_z)$ is obtained which is not in accordance with assumption.

This completes the proof.

3. Summary

The paper presents the algorithm of grammatical processor restricted to declension–conjugation being a kind of dynamic vocabulary in which all inflected forms of analyzed metalanguage variable are generated on line without possibility to remember them for a longer time and thus without redundant computer memory loading.

The definitions present the mathematical description of proper metalanguage variable form generation.

References

LAUS-MĄCZYŃSKA K., (1984) Natural Language Processing (in Polish), PWN, Warszawa.

RATYŃSKA J., (1990) Expert System with Natural Language Communication, International Symposium on SYSTEM - MODELLING - CONTROL.

WETULANI Z., (1990) Corpus of Consultative Dialogues, Wydawnictwo Naukowe UAM, Poznań.