Fundamenta Informaticae 75 (2007) 331–334
IOS Press

Z. Pawlak, a Precursor of DNA Computing and of Picture Grammars

Solomon Marcus*
Romanian Academy
Calea Victoriei 125, Bucharest, Romania
solomon.marcus@imar.ro

41 years ago, Z. Pawlak has published in Polish language a book aimed perhaps for initiation in the field of mathematical linguistics (Pawlak 1965). Short time after this event, he attended an international Conference in Bucharest and I met him there. He offered me a copy of this book. As a matter of fact, he showed me the book and he said that he is sorry to have it in a language which is not available to me. But I told him that I would like to have the book and I will manage to follow it at least partly. Happy idea! Besides some usual introductory notions concerning the mathematical approach to grammars (the title in Polish "Gramatika i matematika" was clearly "Grammar and mathematics"), a special chapter called my attention, because it was concerned with the grammar of the genetic code. I was already introduced, at that time, in the works of Roman Jakobson and of many other authors concerning the analogy between linguistics and molecular genetics. Pawlak's approach was mainly presented in symbols, graphs and geometric pictures, while the few words in Polish were in most cases international words like codons, amino acids, nucleotides, proteins.

It is interesting to recall the period of the sixties of the past century. After a long period in which historical linguistics used ideas and metaphors of Darwinian biology, an important change took place: instead to use biological ideas and metaphors in linguistics, linguistic ideas and metaphors related to phonemic and morphemic segmentation penetrated in the study of nucleic acids, amino acids and proteins.

To this itinerary of opposite sense in respect to the previous one, Pawlak was adding the idea of a generative perspective in the study of heredity. In this aim, he proposed some mechanism operating concomitantly in two directions. On the one hand, in the direction of formal grammars, on the other hand, in the direction of what was called later picture grammars. Let us recall that both formal grammars and picture grammars were at that time at their very beginning. Formal grammars theory had to wait the

*Address for correspondence: Romanian Academy, Calea Victoriei 125, Bucharest, Romania

year 1973 for a first satisfactory rigorous presentation (Salomaa 1973), while picture grammars had to wait the year 1967 for a first systematic attempt (Shaw 1967) and two more years for the monograph by Rosenfeld (1969).

Let us recall the main ideas of Pawlak's approach. Denote by 0, 1, 2, and 3 the four types of nucleotide bases forming the alphabet on which the RNAs are defined. There are 64 modes of arrangements with repetition of them in groups of three elements forming so strings of length three (the constant length of all codons). Codons are for RNAs what morphemes are for well-formed strings in natural languages, while nucleotide bases are for RNAs what phonemes are in natural languages. The starting idea of Pawlak is to associate to each codon an equilateral triangle. Taking into account that a codon is a word of length three on the alphabet 0, 1, 2, 3, the associated triangle will have the respective symbols as labels of its edges. But, as it is well-known, the genetic code establishes a correspondence between codons and amino acids (defining in this way the move from the world of chemistry to the world of biology). There are only 20 types of amino acids relevant for heredity, so Pawlak proposes a way to select exactly 20 types of triangles among the 64 types which are possible from a purely combinatorial point of view. Let us distinguish, for any triangle, the base b, the left edge l and the right edge r, see Figure 1(a). If the codon is ijk, then we associate i to l, j to b, and k to r. Moreover, Pawlak introduces the restriction $i < j \geq k$, i.e., the symbol associated to l is strictly smaller than the symbol associated to b, which is larger than or equal to the symbol associated to r. It can be seen that the only triangles satisfying this requirements are:

$$a = 010,\ b = 011,\ c = 020,\ d = 021,\ e = 022,\ f = 120,\ g = 121,$$
$$h = 122,\ i = 030,\ j = 031,\ k = 032,\ l = 033,\ m = 130,\ n = 131,$$
$$o = 132,\ p = 133,\ q = 230,\ r = 231,\ s = 232,\ t = 233.$$

In a next step, Pawlak introduces a recursive procedure to define a generative picture grammar, whose basic bricks are the 20 types of labeled triangles. The rules of this procedure are the following:

1. Every triangle from the list $a, b, c, \ldots, r, s, t$ is a well-formed string; they are the only well-formed strings of length one.

2. All well-formed strings are words on the alphabet $\{a, b, c, \ldots, r, s, t\}$. Given a well-formed string x and adding to it a triangle A from the list 1, such that the label of its base is the same as the label of the left or right edge of an already existing triangle B in x (in other words, the base of A is the same as the left edge or the right edge of B), then the new string y of triangles so obtained is again well-formed.

3. The strings obtained by rules 1 and 2 are the only well-formed strings.

A saturated well-formed string is one from which no other longer well-formed string can be obtained. For instance, the strings of length one 010, 020, 030 are saturated, the strings $011 \cdot 010$ and $021 \cdot 010$ are saturated strings of length 2 etc. It is easy to see that there are saturated strings of any length. This fact is a consequence of the existing of some triangles that can be added to themselves. For instance, 011 is such a recursive triangle. We can add it to itself n times, then add 010 to obtained a saturated string. For instance, the saturated string of length 4 obtained in this way is: $011 \cdot 011 \cdot 011 \cdot 010$. Other examples of recursive triangles are: 022, 122, 233, 133.

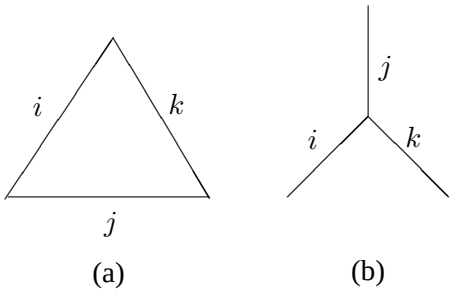

Figure 1. (a) The triangle associated to a codon ijk; (b) A different graphical representation of the triangle in (a).

Pawlak calls protein any saturated well-formed string. He defines a kind of dependency grammar, having 20 rules: to each well-formed triangle ijk Pawlak associates the rule $j \to ik$, where at left we have the label of the base, while at right we have the label of the left edge followed by the label of the right edge. From this dependency grammar Pawlak moves to a graph representation. The triangle ijk is represented by a vertical line associated to the base labeled with j, while from the inferior extremity of this line we start a segment oriented towards the south-left labeled with i and a segment oriented towards the south-right, labeled with k, see Figure 1(b). In this way, the recursive process induces a tree which can be developed as soon as it is not yet saturated.

We have shown in (Marcus 1974) that a non-deterministic propagating semi-Lindenmayer system can be defined, which is equivalent to the above defined Pawlak mechanism. But we are interested not only in the result of the generative process; we would like to know something about the structure of the language of derivations in the respective semi-Lindenmayer system. This question was left unanswered in (Marcus 1974). In the same paper we have presented a Chomsky type picture grammar which is context-free, but whose language of derivations is not context-free; it is just the Chomskian equivalent of Pawlak's dependency picture grammar.

Some advantages and some shortcomings of Pawlak's mechanism and of the corresponding Chomskian mechanism are discussed in (Marcus 1974). The whole problem deserves to be reconsidered, in the light of the new field of DNA computing, for which we send the reader to (Păun-Rozenberg-Salomaa 1998).

We conclude with some hints about the idea of a semi-Lindenmayer system. It is an ordered pair $S = \langle V, p\rangle$, where V is a finite non-empty set called alphabet, while p is a mapping associating to each element in V a language over V. If for each $v \in V$ the set $p(v)$ contains exactly one finite string over V, then S is said to be deterministic; otherwise, S is said to be non-deterministic. We say that S is propagating, if for each $v \in V$ any string in $p(v)$ is of strictly positive length; otherwise, S is non-propagating. Define now the language $L(S, M)$ generated by a semi-Lindenmayer system S with respect to a language M over V. The string y directly derives from the string x of strictly positive length if there exists a positive integer n such that $x = a(1)a(2)\dots a(n),\ y = b(1)b(2)\dots b(n)$, where each $a(i)$ ($1 \le i \le n$) belongs to V and each $b(i)$ ($1 \le i \le n$) belongs to V^*, with $b(i) \in p(a(i))$ for any $1 \le i \le n$. If p is a homomorphism, we put for any finite string w over V, $w = c(1)c(2)\dots c(n)$, $p(w) = p(c(1))p(c(2))\dots p(c(n))$. We say that the string v derives in S from the string u of strictly positive length if there exists a finite sequence of finite strings $x(1), x(2), \dots, x(q)$ over V, such that $x(1) = u$, $x(q) = v$, and $x(i+1)$ directly derives from $x(i)$ for any $1 \le i \le q-1$. The string y is

generated by S with respect to the language M over V if there exists $x \in M$ from which y derives in S. $L(S, M)$ is by definition the set of all strings generated by S with respect to M.

In (Marcus 1974), it is proved that the Pawlak dependency grammar can be expressed as a non-deterministic propagating semi-Lindenmayer system with respect to the language M consisting of four strings of length one: 0, 1, 2, 3. The mapping p is defined by $p(0) = \{0\}, p(1) = \{00, 01\}$, $p(2) = \{00, 01, 02, 10, 11, 12\}$, $p(3) = \{00, 01, 02, 03, 10, 11, 12, 13, 20, 21, 22, 23\}$. The language generated by the considered system is just the set of all saturated well-formed strings, in the sense of Pawlak, i.e., the set of proteins. What about the language of derivations in S?

A basic shortcoming of Pawlak's approach was that he did not take in consideration the Watson-Crick structure of double-helix. Our 1974 approach continuing Pawlak's work had the same shortcoming, as it was clearly mentioned in (Marcus 1974). This missing structure became just the point of departure in Tom Head's pioneering work on DNA computing (Head 1987).

References

[1] T. Head (1987), Formal language theory and DNA: an analysis of the generative capacity of specific recombinant behaviors. Bull. Math. Biology 49 (1987), 737–759.

[2] S. Marcus (1974), Linguistic structures and generative devices in molecular genetics. *Cahiers de Linguistique Théorique et Appliquée*, 11, 2, 77–104.

[3] Gh. Păun, G. Rozenberg, A. Salomaa (1998), *DNA Computing. New Computing Paradigms*. Springer, Berlin.

[4] Z. Pawlak (1965), *Gramatika i matematika*. Publishig House of the Polish Academy of Sciences, Warsaw.

[5] A. Rosenfeld (1969), *Picture Processing by Computer*. Academic Press, New York – London.

[6] A.C. Shaw (1967), *A Proposed Language for the Formal Description of Pictures*. GSG Memo 28, Stanford, California, February 1967, 32 pp.

[7] A. Salomaa (1973), *Formal Languages*. Academic Press, New York – London.