In the prehistory of formal language theory: Gauss languages

Lila Kari[2], Solomon Marcus[1],
Gheorghe Paun[1], Arto Salomaa[2]

Abstract

The problem proposed by Gauss of characterizing the code of a simple crossing closed curve (SCCC, for short) can be considered a formal language question. We define three related infinite languages. Two of them are regular; the type of the third is an open problem.

1 Gauss codes

The origin of formal language theory is usually considered (see [12]) to be the Thue paper [13] but, as it is pointed out in [5], one can identify formal language theory problems even in [7]; the topic, known in combinatorics under the name of Langford strings [3], raises challenging formal language theory problems (see [6], [9]). However, the prehistory of formal language theory can be spectacularly enlarged, taking into account that also C. F. Gauss proposed and investigated [1] a problem which could be considered of syntactic nature, dealing with the formal structure of strings of abstract symbols. It concerns the so-called *Gauss code* describing a planar closed curve with simple crossing points (a point is simple if it is not a tangent point and the curve crosses itself only once at that point). Assign the numbers $1, 2, \ldots, n$ to the n crossing points of a given curve c. A sequence x_c containing exactly two occurrences of each $i, 1 \leq i \leq n$, and describing the passing of the curve through the crossing points is called a *Gauss code* (of the curve c). Example: for the curve c in Figure 1, the sequence

$$x_c = 123441562365$$

is a Gauss code.

[1]University of Bucharest, Faculty of Mathematics Str. Academiei 14, 70109 Bucuresti, ROMANIA

[2]Academy of Finland and University of Turku, Department of Mathematics, 20500 Turku,FINLAND

Fig. 1

In [1], Gauss called for the characterization of Gauss codes (of SCCC) in terms of the interlacement properties of their symbols. Moreover, Gauss himself has proved such a syntactic-like condition, which is only necessary for a string to be a Gauss code:

Denote $V_n = \{1, 2, \ldots, n\}$ and let $x \in V_n^*$ be a string such that $\mid x \mid_i = 2, 1 \leq i \leq n$. ($\mid x \mid_i$ is the number of occurrences of symbol i in x). Denote

$$V(x, i) = \{j \in V_n \mid x = x_1 i x_2 i x_3, x_1, x_2, x_3 \in V_n^*, \mid x_2 \mid_j = 1\}$$

(the set of symbols having exactly one occurrence between the two occurrences of i).

In [1] it is proved that for a Gauss code x_c, each set $V(x_c, i), 1 \leq i \leq n$, is of even cardinality.

The problem of characterizing Gauss codes was approached in a large number of papers (see [2] for a history of the problem). Most of the proposed solutions are topological and graph theoretical (even the "algebraic" one in [10] is of this type; see also [11]). There exist syntactic characterizations too; see, for instance [4], where a theorem of the following type is proved : "a word x is a Gauss code if and only if it contains no subwords of the form...".

2 Gauss languages (I)

Clearly, as it stands, the Gauss problem refers to finite strings (hence languages), therefore it is not purely of formal language nature. However, certain infinite languages can be naturally defined in this frame.

The most natural idea is to consider paths of arbitrary lengths along a SCCC. As above, describe such a path by the sequence of visited points; call such a sequence a *weak Gauss code*. Given a curve c, denote by $WG(c)$ the set of all weak Gauss codes associated to it. Clearly,

Proposition 1 *(i)$WG(c)$ is an infinite language; (ii) $WG(c) = mi(WG(c))$, for any SCCC c. (mi denotes the mirror image operation.)*

Having an infinite language, it is natural to ask which is its place in the Chomsky hierarchy; the question can easily be answered using the next formal construction of $WG(c)$.

First, some notations: for a string z over some alphabet V, denote

$$\sigma(z) = \{z_2 z_1 \mid z = z_1 z_2, z_1, z_2 \in V^*\}$$

(the *circular permutation* of z). For $L \subseteq V^*$, denote by $D(L)$ the smallest language $L' \subseteq V^*$ containing L and having the next property: if $w \in L', w = w_1 w_2, w_1, w_2 \in V^*$, and $w_2 a \in L$ for some $a \in V$, then $wa \in L'$(the right prolongation of L according to itself). Denote also by *sub(z)* the set of all subwords of $z \in V^*$.

Proposition 2 *Let c be a SCCC and x_c be a Gauss code associated to it, $x_c \in V_n^*$, for some $n \geq 1$. Then,*

$$WG(c) = L_c \cup mi(L_c)$$

where

$$L_c = \ sub(D(\sigma(x_c))).$$

Proof. The code x_c determines an orientation of the curve c; $\sigma(x_c)$ contains all Gauss codes describing the curve c in this orientation. If x'_c is another Gauss code associated to c, then either $x'_c \in \sigma(x_c)$ or $x'_c \in mi(\sigma(x_c))$; in the first case $\sigma(x'_c) = \sigma(x_c)$, in the second one $\sigma(x'_c) = mi(\sigma(x_c))$. Therefore, the choice of x_c is not important. Now, a string z in $WG(c)$, with the length larger than $\mid x_c \mid$, is obtained from a string in $\sigma(x_c)$ or in $mi(\sigma(x_c))$, depending on the orientation of z with respect to the orientation of x_c, by right prolongation. Any substring of such a string is in $WG(c)$ too. In conclusion, each weak Gauss code associated to c is either in L_c or in $mi(L_c)$ and, conversely, each string in $L_c \cup mi(L_c)$ is a weak Gauss code associated to c. □

From this representation we obtain

Proposition 3 *For any SCCC c, the language $WG(c)$ is regular.*

Proof. Indeed, $\sigma(x_c)$ is a finite language, and the family of regular languages is closed under operations D([8]), *sub*, *mi*, union. □

Given a curve c, the Gauss code x_c is uniquely determined up to a circular permutation and the mirror image (hence the language $WG(c)$ is uniquely determined).

The converse is not true, even considering topologically equivalent curves. For instance, given two points, 1, 2, we have two essentially different "codes" (modulo the circular permutation and the mirror image), namely

$$1212, 1122$$

The first one cannot describe a SCCC (apply Gauss criterion: $V(1212, 1)$ and $V(1212, 2)$ are of odd cardinality), but 1122 can describe three topologically different curves - see Figure 2.

Fig. 2

Problem 1. What topological/geometrical properties of a curve c can be (algorithmically) inferred from x_c(or from $WG(c)$), hence are common to all curves associated to a given Gauss code x_c ?

For instance, consider the number of crossing points on the edges of simple closed regions (not composed of two closed regions) determined by a curve (the external region is not taken into consideration - or it can be taken separately). Call this number the *order* of the simple closed region and call the *order of the curve* the maximum order of a simple closed region of this curve. (We can call *external order* of the curve the order of the external region.) For instance, in Figure 1 we have closed regions of order one, two, three, four: (4), (5, 6), (2, 3, 6), (1, 2, 6, 5), respectively, whereas the external order is five (1, 4, 3, 6, 5 are on the frontier).

On the other hand, in Figure 2 all curves are of order two, all contain two regions of order one and one of order two. Is the order/the external order precisely identified by the Gauss code, for any curve associated to it ? Can we deduce from examining the code whether the curve contains simple closed

regions of a given order ? Clearly, a simple closed region of order one corresponds to a substring of the form ii, and a simple closed region of order two corresponds to the existence of a substring ij appearing twice or to the pair of substrings ij, ji appearing in the Gauss code describing the curve. What about higher orders ?

Problem 2. Is it possible to represent/characterize (in a "simple" and "natural" way) the family of regular languages obtained from Gauss codes (languages $WG(c)$) and using suitable operations with languages ?

3 Gauss languages (II)

Given a SCCC c, another language can be constructed too, considering paths along c, but permitting returnings along segments, not on intersection points. Thus, we do not have a fixed orientation of the curve, but we can go freely forward and backward on it. (Of course, after passing through i, if we came

back, we have to pass again through i.)

Call such strings *double-weak Gauss codes* and denote by $DWG(c)$ the language associated in this way to c. Clearly, we have also now

Proposition 4 *(i) $DWG(c)$ includes $WG(c)$(hence $DWG(c)$ is infinite); (ii) $DWG(c) = mi(DWG(c))$, for any SCCC c.*

The language $DWG(c)$ is in general strictly larger than $WG(c)$. More exactly, we have

Proposition 5 *If c is a SCCC with at least two intersection points, then $WG(c)$ is strictly included in $DWG(c)$.*

Proof. If there is in ca simple cycle from some j to the same point (Figure 3.a), then we cannot find in $WG(c)$ substrings of the form $pj^3q, p \neq j \neq q$, but such substrings can appear in strings of $DWG(c)$.

Fig. 3

Similarly, when there is no simple cycle for a point j(Figure 3.b), then $WG(c)$ does not contain substrings of the form $pj^2q, p \neq j \neq q$, but such substrings can appear in strings of $DWG(c)$. □

Problem 3. Find a representation of $DWG(c)$ (similar to that in Proposition 2 for $WG(c)$).

A result analogous to Proposition 3 can be easily obtained for $DWG(c)$ by direct arguments.

Proposition 6 *$DWG(c)$ is regular for any SCCC c.*

Proof. Take a curve c, with intersections marked by elements of V_n and construct the right-linear grammar

$$G = (V_N, V_n, S, P)$$

with

$$V_N = \{[i, j] \mid 1 \leq i, j \leq n, i \text{ is directly linked to } j \text{ by the curve } c\} \cup \{S\},$$

$$P = \{S \to [i,j], [i,j] \to \lambda \mid [i,j] \in V_N\} \cup$$

$$\cup \{[i,j] \to j[j,k] \mid [i,j],[j,k] \in V_N\} \cup$$

$$\cup \{[i,j] \to i[k,i] \mid [i,j],[k,i] \in V_N\}.$$

The equality $L(G) = DWG(c)$ is obvious, hence $DWG(c)$ is regular. □

The language $DWG(c)$ is larger than $WG(c)$, but it is not "too large". More exactly, we have

Proposition 7 *The Gauss criterion is a necessary condition for a string to be in DWG(c).*

Proof. Consider a set $V(x,i), x \in DWG(c), i \in V_n$. If when writing $x = x_1 i x_2 i x_3$ we came from i back to i, after passing through x_2, on the same segments of c(with a returning point somewhere inside x_2), then we pass twice (at least) through some intersection point in x_2, hence such points do not appear in $V(x,i)$. Similarly, if we have returnings in x_2, the involved points do not appear in $V(x,i)$. Thus, if we return to i on another segment of c, after passing only one time through the segments in x_2(without returnings), this implies we have a closed region determined by ix_2i. The numbers of points used for *coming in* and for *coming out* this region are equal; this means the number of symbols appearing only once in x_2(each corresponds either to a coming in or to a coming out) is even. This is exactly the Gauss criterion. □

4 Gauss languages (III)

Another way for obtaining an infinite language is to allow points of multiple crossing. More exactly, given n points, consider all planar closed curves which cross arbitrarily many times in these points, in the sense that each passing through a point intersects all other passings of the curve through that point (no two curve branches are tangent in a crossing point). Denote by SG_n the set

$$sub\ \{x \mid x \text{ is a Gauss code of a curve passing arbitrarily many times through points } 1,2,\ldots,n\}$$

Please note that SG_n refers to *all* curves which pass through (some of) points $1,2,\ldots,n$. We call such strings *semi-Gauss codes.*

Proposition 8 *All languages $SG_n, n \geq 1$, are infinite.*

Proof. We shall show that $SG_1 = \{1^k \mid k \geq 2\}$(therefore it is infinite) and that $SG_n \subset SG_{n+1}, n \geq 1$.

The idea of proving the former assertion is that in Figure 4. For an odd number of passings through the crossing points (and for $k = 2$ too), we get ak-petal "flower", and for an even number of passings we get $a(k-1)$-petal "flower", provided with a "macro-petal" - see Figure 5.

Fig. 4

Fig. 5

On the other hand, each Gauss code in SG_n can be viewed as an element of SG_{n+1}(zero passings through point $n+1$). Moreover, for each string $x \in SG_n$ we can find a string $x' \in SG_{n+1}$ effectively passing through the point $n+1$. Indeed, write $x = x_1rsx_2, x_1, x_2 \in V_n^*, r, s \in V_n(r, s$ may be different or not). Then, the string $x' = x_1r(n+1)(n+1)sx_2$ is a semi-Gauss code (hence $x' \in SG_{n+1}$); Figure 6 indicates the way of constructing a curve for x', starting from a curve for x(the dotted "region" of the x-curve remains unchanged). □

Fig. 6

Clearly, SG_1 is a regular language.

Problem 4. Which is the place of languages $SG_n, n \geq 2$, in the Chomsky hierarchy? Are there "simple" characterizations of SG_n for small values of $n (n = 2$, for example) ?

5 Comparing Gauss-Thue-Langford strings

A natural "combinatorial puzzle" is now to ask whether a Gauss or a semi-Gauss code can be square- or cube-free or a Langford string. (Recall that an (m,n)-Langford string over $V_n = \{1,2,\ldots,n\}$ is a string $x \in V_n^*$ such that (a) $\mid x \mid_i = m, 1 \leq i \leq n$, and (b) for each writing $x = x_1 i x_2 i x_3, \mid x_2 \mid_i = 0$, we have $\mid x_2 \mid = i, 1 \leq i \leq n$. A string $x \in V_n^*$ fullfilling only condition (b) is called weak-Langford [6].)

First, let us point out that the strings $(12)^k 1, k \geq 2$, are in SG_2(see Figure 7), but contain subwords x^t with arbitrarily large t.

Fig. 7

On the other hand, $h^3(1)$, for the (classical) Thue morphism $h : \{1,2\}^* \to \{1,2\}^*$ defined by $h(1) = 12, h(2) = 21$, [13], is a semi-Gauss code. In Figure 8 we provide a curve the semi-Gauss code of which is

$$h^3(1) = 12212112$$

Fig. 8

A similar (but more complicated) curve can be constructed for

$$h^4(1) = 1221211221121221$$

Problem 5. We *conjecture* that all strings $h^n(1), n \geq 3$, are semi-Gauss codes (hence there are arbitrarily long cube-free strings in SG_2).

As regards the Langford case, there is no $(2, n)$-Langford string in SG_n. This follows from the necessary Gauss condition quoted above: each $(2, n)$-Langford string w contains exactly two occurrences of each i, hence if $w \in SG_n$, then w is a Gauss code; moreover, each $(2, n)$-Langford string x must contain a substring $1 \leq k \leq 1$, hence we have $V(x, 1)$ of odd cardinality, and x cannot be a Gauss code.

Problem 6. We *conjecture* that no (m, n)-Langford string, with $m \geq 2$, can be a semi-Gauss code.

On the other hand, there are weak-Langford strings in SG_3. One example is

$$x = 2312132$$

corresponding to the curve in Figure 9.

Fig. 9

The problem of finding weak-Langford strings of arbitrary length which are Gauss codes remains open.

Notes. Problems 1, 5, 6 have been approached and partially solved in [1].

We gratefully acknowledge the bibliographical help provided us by dr. Sorin Istrail, Wisconsin University, USA. The pictures with lassoing animals are by Anu Heinimki.

References

[1] J.Cassaigne, S.Schwer, P.Seebold. About Gauss codes. *Bull. EATCS*, to appear.

[2] C. F. Gauss. *Werke*. Teubner, Leipzig, 1900 (pp. 272 and 282 - 286).

[3] B. Grünbaum. Arrangements and spreads. *Conf. Board. Math. Sci. Reg. Conf. Ser. Math.* nr. **10**, Amer. Math. Soc., Providence, *RI*, 1972.

[4] C. Langford. Problem. *Math. Gazette*, **42**(1958), 228.

[5] L. Lovász, M. L. Marx. A forbidding substructure characterization of Gauss codes. *Bull. Amer. Math. Soc.* 82, **1** (1976), 121 - 122.

[6] S. Marcus. Formal languages before Axel Thue ? *Bull. EATCS*, **34** (1988), 62.

[7] S. Marcus, Gh. Paun. Langford strings, formal languages and contextual ambiguity. *Intern. J. Computer Math.*, **26** (1988), 179 - 191.

[8] E. Netto. *Lehrbuch der Combinatorik*, Leipzig, 1901.

[9] Gh. Paun. *Generative mechanisms for economic processes*, Ed. Tehnica, Bucuresti, 1980 (in Romanian).

[10] Gh. Paun. On Langford-Lyndon-Thue sequences. *Bull. EATCS*, **34** (1988), 63 - 67.

[11] P. Rosenstiehl. Solution algebrique du probléme de Gauss sur la permutation des points d'intersection d'une ou plusieurs courbes fermées du plan. *C. R. Acad. Sci. Paris*, 283, **8** (1976), 551 - 553.

[12] P. Rosenstiehl, R. J. Tarjan. Gauss codes, planar hamiltonian graphs, and stack-sortable permutations. *J. Algorithms*, **5** (1984), 391 - 407.

[13] A. Salomaa. Two-way Thue. *Bull. EATCS*, **32** (1987), 82 - 86.

[14] A. Thue. Uber unendliche Zeichenreihen. *Videns. selskapets Skrifter*, Kristiania, 1906, 1 - 22.