Acta Mathematica Academiae Scientiarum Hungaricae
Tomus 21 (1—2), (1970), pp. 27—33.

ON THE ENUMERATION OF SEARCH-CODES

By

A. RÉNYI (Budapest), member of the Academy

Dedicated to Professor G. ALEXITS *at the occasion of his 70th birthday*

§ 1. Definition of search-codes

Any finite sequence of nonnegative integers is called a *codeword*. If $a=(a_1, a_2, \ldots, a_s)$ is a codeword, we call s the *length* of the codeword a and put $|a|=s$. The empty sequence (of length 0) is considered as a codeword too. We denote by Z the set of all codewords. If $a=(a_1, a_2, \ldots, a_r)$ and $b=(b_1, b_2, \ldots, b_s)$ are any two codewords, we define the codeword ab as $ab=(a_1, a_2, \ldots, a_r, b_1, b_2, \ldots, b_s)$. Clearly $|ab|=|a|+|b|$, and Z is a semigroup with respect to the multiplication defined above. Denoting by e the empty codeword, e is the unit element of the semigroup Z. A codeword a is called a *prefix* of the codeword b if there exists a codeword c such that $b=ac$; in this case c is called a *suffix* of b. If $a=(a_1, \ldots, a_s)$, $a_1, \ldots, a_s$ are called the *letters* of a.

Any finite set $C=\{c_1, c_2, \ldots, c_n\}$ of different codewords $c_j=(c_{j,1}, c_{j,2}, \ldots, c_{j,l_j})$ is called a *code*. It is convenient to consider the empty set also as a code: we call it *the empty code*. The code consisting of the single codeword e, where e is the empty codeword is a nonempty code, which will be called *the trivial code*.

If C is a code and a any codeword (not necessarily in C) we denote by C_a the set of those codewords $b\in Z$ for which $ab\in C$. In other words C_a is obtained by taking all those codewords c of C of which a is a prefix and removing from these codewords the prefix a, i.e. preserving only the suffix of c following a. Evidently $C_e=C$, and $(C_a)_b=C_{ab}$. Of course for certain $a\in Z$ (in fact for all but a finite number of $a\in Z$) C_a is the empty code. If $a=c\in C$ then C_c is not empty, as it contains certainly the empty codeword e, but C_c may be the trivial code: if C_c is for every $c\in C$ the trivial code, C is called a *prefix code*. Thus a code C is a prefix code if none of its codewords is the prefix of an other codeword in C.

If C is a code, we shall denote by $N(C)$ the number of elements of C, and we shall call $N(C)$ the *size* of C.

We shall call a code C *well-branched*, if one of the following three cases takes place: 1. C is the empty code; 2. C is the trivial code; 3. C does not contain the empty codeword e and there exists an integer $b(C)$ — called *the branching number of* C — such that $b(C)\geqq 2$ and denoting by $\{k\}$ the codeword of length 1, consisting of the single letter k ($k=0, 1, \ldots$) the code $C_{\{k\}}$ is not empty if $k<b(C)$ and $C_{\{k\}}$ is empty if $k\geqq b(C)$. If C is the empty code we put $b(C)=0$ while if C is the trivial code we put $b(C)=1$; thus the branching number $b(C)$ is defined for every well-branched code.

We call a code C a *search code* if C_a is a well-branched code for every $a\in Z$. Clearly if C is a search code, then C is necessarily a prefix code. As a matter of fact if $c\in C$ the code C_c has to be a well branched code, and as it contains the empty codeword e, C_c has to be the trivial code: thus C is a prefix code. However, not every

prefix code is a search-code: for instance the code consisting of a single codeword, which is different from the empty codeword e, is a prefix code, but not a search code.

If C is a search code, we call those $a \in Z$ for which $b(C_a) \geqq 2$, the *branching points* of C. We shall denote the total number of branching points of C by $B(C)$.

A search code C is called *regular of degree* q ($q \geqq 2$), if each of its branching points has the branching number q, i.e. if for each $a \in Z$ such that $b(C_a) \geqq 2$, one has $b(C_a) = q$. The trivial code is by definition regular of every degree $q \geqq 2$.

The study of search codes is motivated by the connection of search-codes with *the mathematical theory of search* (see e.g. [1]). Let us consider a simple search process in which we want to identify an unknown element x of a finite set S. Suppose that we can not observe x directly, however we may choose some functions $f_1, f_2, \ldots, f_N$ from a given set F of nonnegative integer valued functions f defined on the set S, and observe the values $f_1(x), f_2(x), \ldots, f_N(x)$ of these functions at the unknown point x. A method for choosing the functions f_k, one after the other, from the set F (so that the choice of f_k may depend on the previously observed values $f_1(x), \ldots, f_{k-1}(x)$) so that after a finite number N of steps ($N = N(x)$ may depend on x) the unknown x is uniquely determined by the sequence $f_1(x), f_2(x), \ldots, f_N(x)$, is called *a strategy of search*. Clearly each strategy defines a code, the codewords of which are the sequences $(f_1(x), f_2(x), \ldots, f_{N(x)}(x))$ $(x \in S)$. A strategy is called *irreducible*, if none of the observations prescibed by the strategy is unnecessary. It is easy to see that the codes corresponding to irreducible strategies of search are just the search-codes defined above, provided that for every possible sequence $f_1(x), \ldots, f_{k-1}(x)$, if n is a possible value of $f_k(x)$, then so is every $m < n$, as well.

To every search-code there corresponds a rooted tree, defined as follows: the root of the tree corresponds to the empty code word, the branching points of the tree correspond to the branching points of the code, and the endpoints of the tree correspond to codewords belonging to the code. Two vertices of the tree are connected by an edge if the two corresponding codewords a and b are such that a is a prefix of b and $|b| - |a| = 1$.

In what follows we shall count search codes satisfying certain conditions. We shall use throughout the paper the method of generating functions. An other way to solve these problems is by means of the mentioned correspondence between codes and trees, making use of the methods of the theory of trees. However, this approach has the disadvantage, that to different codes there may correspond the same tree. In dealing with regular codes this does not cause any difficulty, but in the general case the counting of trees is somewhat more involved: as a matter of fact for some types of codes in the present paper, we get explicit formulae for the total number of codes, while for the number of the corresponding trees we were able to get explicit formulas only for the corresponding generating functions. In view of this we do not deal here at all with the counting of trees corresponding to the search-codes considered. However, we inted to return to these questions elsewhere.

§ 2. Enumeration of regular search-codes

Let C be a regular search code of degree $q \geqq 2$. First we prove the following

LEMMA. *If C is a regular search-code of degree q then the size $N(C)$ of the code C satisfies the following congruence relation:*

$$N(C) \equiv 1 \bmod (q-1). \tag{2. 1}$$

Conversely, if n is a natural number such that $n-1$ is divisible by $q-1$, then there exist regular search codes of degree q and size n.

PROOF. Let $E(C)$ denote the number of pairs (a, b) of code-words, such that both C_a and C_b are different from the empty code, further a is a prefix of b and $|b|-|a| = 1$. To determine $E(C)$ we may first fix a and count in how many ways b can be chosen, or conversely, fix b and consider in how many ways a can be chosen. Thus we obtain two expressions, and equating these we shall get (2. 1). Evidently a can be chosen as one of the branching points of C, i.e. in $B(C)$ ways, and if a is fixed, by definition b can be chosen in $b(C_a) = q$ ways; thus

$$E(C) = q \cdot B(C). \tag{2. 2}$$

On the other hand, b can be chosen in $B(C)+N(C)-1$ ways,* and to every b there corresponds a uniquely determined a, obtained by omitting the last letter of b. Thus

$$E(C) = B(C)+N(C)-1. \tag{2. 3}$$

Comparing (2. 2) and (2. 3) we get

$$N(C) = B(C)(q-1)+1 \tag{2. 4}$$

which proves (2. 1).

REMARK. $E(C)$ is equal to the number of edges of the tree corresponding to the code C.

The second statement of the lemma can be proved by induction as follows: If $n=1$, there exists a (unique) regular code of degree q and size $N(C)=1$, namely the trivial code. Suppose we have already constructed a regular search code of degree q and having size $N(C)=n$, where $n-1$ is divisible by $q-1$. Take any one of the codewords c belonging to the code C, omit c from C and add to C the q code-words obtained by adding to the end of c the suffix of length 1 consisting of the single letter k, where k is any one of the numbers $0, 1, \ldots, q-1$. In this way we obtain a code C' which is regular of degree q and has size $N(C') = n+q-1$. This proves Lemma 1.

We shall determine now the total number $C_q(n)$ of regular search codes of degree q and size n, where of course $n \equiv 1 \bmod (q-1)$. We shall prove the following

THEOREM 1. *For every natural number n of the form $n = k(q-1)+1$ $(k=0, 1, \ldots)$ one has*

$$C_q(n) = \frac{(kq)!}{k! \cdot n!}. \tag{2. 5}$$

* We can namely choose for b any codeword belonging to C and any branching point of C, except e.

PROOF. Clearly $C_q(1)=1$, and for every $n = k(q-1)+1$ with $k \geqq 1$ we have the recursion formula

$$C_q(n) = \sum_{n_1+n_2+\ldots+n_q=n} C_q(n_1)C_q(n_2)\ldots C_q(n_q), \tag{2. 6}$$

where the summation has to be extended only over those values of n_j for which $C_q(n_j)$ is defined i.e. for $n_j \equiv 1 \bmod (q-1)$.

Let us put

$$G_q(y) = \sum_{n \equiv 1 \bmod (q-1)} C_q(n)x^n. \tag{2. 7}$$

We get from (2. 6)

$$G_q(y) = y+(G_q(y))^q. \tag{2. 8}$$

From (2. 8) one can deduce, by the Bürmann—Lagrange formula for the power series of the inverse function of a given function, that

$$G_q(y) = \sum_{k=0}^{\infty} \frac{(kq)!}{k!(k(q-1)+1)!} y^{k(q-1)+1}, \tag{2. 9}$$

i.e. that (2. 5) holds.

REMARK. The power series expansion (2. 9) can be interpreted as the power series of that real root x of the trinomial equation $x-x^q = y$, which reduces to 0 for $y=0$. It is easy to see that the power series (2. 9) is convergent for $|y| \leqq \frac{q-1}{q^{\frac{q}{q-1}}}$ and thus (2. 9) solves the equation $x-x^q = y$ for a given value of y if $|y| \leqq \frac{q-1}{q^{\frac{q}{q-1}}} = R_q$. Notice that R_q tends increasingely to 1 for $q \to +\infty$.

We want to add some remarks on the special case $q=2$, i.e. the case of *binary search codes*. If $q=2$, then by Lemma 1 $N(C) = B(C)+1$ i.e. there exist regular binary search codes of any size $n \geqq 1$, and we get from (2. 5) that their total number is

$$C_2(n) = \frac{\binom{2n-1}{n}}{(2n-1)} = \frac{(2n-3)!!\,2^{n-1}}{n!}. \tag{2. 10}$$

This formula has been known already to A. CAYLEY (see [2] and [3]), who solved the corresponding problem for counting trees. Cayley pointed out also that $C_2(n)$ is equal to the number of possible interpretations of a „product" of n factors with respect to a nonassociative operation, i.e. $C_2(n)$ is equal to the possible bracketings of such a „product".

Thus Theorem 1 is a straightforward generalization of Cayley's classical results. It seems improbable that this generalization has not been made previously, but I did not find it in the literature.

Let us notice that for $q=2$ we get from (2. 8) the explicit formula

$$G_2(y) = \frac{1-\sqrt{1-4y}}{2}. \tag{2. 11}$$

§ 3. Enumeration of all search codes of given size

Let $D(n, k)$ denote the total number of search codes of size n and having k branching points. Evidently $k \leqq n-1$, as for a given size the number of branching points is maximal if the code is binary. It is easy to see that $D(1, 0)=1$, $D(n, 0)=0$ if $n \geqq 2$, and for $n \geqq 2$ the following recursion formula holds:

$$D(n, k) = \sum_{l=2}^{n} \sum_{\substack{n_1+n_2+\ldots+n_l=n \\ k_1+k_2+\ldots+k_l=k-1}} D(n_1, k_1) D(n_2, k_2) \ldots D(n_l, k_l). \tag{3. 1}$$

Let us put

$$\sum_{n=1}^{\infty} \sum_{k=0}^{\infty} D(n, k) x^n y^k = d(x, y). \tag{3. 2}$$

It follows that

$$d(x, y) = x + \frac{y\, d^2(x, y)}{1-d(x, y)}. \tag{3. 3}$$

Thus we get

$$d(x, y) = \frac{1+x-\sqrt{(1+x)^2-4x(y+1)}}{2(y+1)}. \tag{3. 4}$$

Let us denote now by $D(n)$ the total number of search codes of size n. It follows, putting

$$\delta(x) = \sum_{n=1}^{\infty} D(n) x^n, \tag{3. 5}$$

that

$$\delta(x) = d(x, 1) = \frac{1+x-\sqrt{1-6x+x^2}}{4}. \tag{3. 6}$$

From (3. 4) and (3. 6) we can get explicit formulae for $D(n, k)$ and $D(n)$, respectively. By some easy calculations we get

$$D(n, k) = \frac{1}{n} \binom{n-2}{k-1} \binom{n+k-1}{k} \quad (n \geqq 2,\ k \geqq 1) \tag{3. 7}$$

and thus

$$D(n) = \frac{1}{n} \sum_{k=1}^{n-1}{}' \binom{n-2}{k-1} \binom{n+k-1}{k} \quad (n \geqq 2). \tag{3. 8}$$

From (3. 6) we obtain for $D(n)$ the asymptotic formula

$$D(n) \sim \frac{\sqrt{3-2\sqrt{2}}}{4\sqrt{\pi}} \frac{(3+2\sqrt{2})^n}{n^{3/2}}. \tag{3. 9}$$

The first values of $D(n)$ are $D(2)=1$, $D(3)=3$, $D(4)=11$, $D(5)=45$, and $D(6)=197$.

We consider now the following enumeration problem: let $D_r(n)$ denote the total number of search codes of size n consisting of codewords all having length

$\leqq r$ $(r=1, 2, \ldots)$. By the same type of argument which led us to (3. 3) we obtain, putting

$$\delta_r(x) = \sum_{n=1}^{\infty} D_r(n) x^n \tag{3.10}$$

that

$$\delta_r(x) = x + \frac{\delta_{r-1}^2(x)}{1-\delta_{r-1}(x)}. \tag{3.11}$$

Thus we get successively

$$\delta_1(x) = \frac{x}{1-x},$$

$$\delta_2(x) = x + \frac{x^2}{(1-x)(1-2x)},$$

$$\delta_3(x) = x + \frac{x^2(1-2x+2x^2)^2}{(1-x)(1-2x)(1-4x+4x^2-2x^3)},$$

etc.

Evidently

$$\lim_{r \to +\infty} \delta_r(x) = \delta(x). \tag{3.12}$$

One can determine similarly the generating functions of the total number of *regular* search codes of degree q and size n, consisting of codewords all having the length $\leqq r$. If the number of such codes is denoted by $B_q(n, r)$, and we put

$$\beta_q(r, x) = \sum_{\substack{n=1 \\ n \equiv 1 \bmod (q-1)}}^{\infty} B_q(n, r) x^n \tag{3.13}$$

we obtain $\beta_q(0, x) = x$ and

$$\beta_q(r, x) = x + [\beta_q(r-1, x)]^q \qquad (r=1, 2, \ldots). \tag{3.14}$$

Especially we get for the binary case $\beta_2(0, x) = x$ and

$$\beta_2(r, x) = x + \beta_2^2(r-1, x) \quad \text{for} \quad r \geqq 1. \tag{3.15}$$

Thus we get successively the polynomials

$$\beta_2(1, x) = x + x^2,$$

$$\beta_2(2, x) = x + (x + x^2)^2,$$

$$\beta_2(3, x) = x + [x + (x + x^2)^2]^2,$$

etc.

Let $B_2(r)$ denote the total number of binary search codes, consisting of codewords all having length $\leqq r$, irrespectively of the number of codewords in the code. As clearly

$$B_2(r) = \sum_{n=1}^{\infty} B_2(n, r) = \beta_2(r, 1), \tag{3.16}$$

we get the recursion formula

$$B_2(0) = 1, \tag{3.17}$$

$$B_2(r+1) = 1 + B_2^2(r) \quad \text{for} \quad r \geqq 0$$

from which one can calculate successively the values of $B(r)$. We have $B_2(1)=2$, $B_2(2)=5$, $B_2(3)=26$, $B_2(4)=677$, $B_2(5)=458330$. (Notice that the trivial code is counted here as the unique binary code consisting of a single codeword of length 0.) If $b(r)$ denotes the total number of binary search codes such that the length of the longest codeword of the code is equal to r, then $b_2(r) = B_2(r) - B_2(r-1)$; thus we have $b_2(1)=1$, $b_2(2)=3$, $b_2(3)=21$, $b_2(4)=651$ etc.

One can get from (3.17) also asymptotic formulae for the sequences $B_2(r)$ resp. $b_2(r)$ for $r \to +\infty$.

(Received 30 April 1969)

MTA MATEMATIKAI KUTATÓ INTÉZETE,
BUDAPEST, V., REÁLTANODA U. 13—15

References

[1] A. RÉNYI, Lectures on the mathematical theory of search, *Department of Statistics of the University of North Carolina.* (In print.)

[2] A. CAYLEY, On the theory of the analytical forms called trees. I—II, *Philosophical Magazine*, **13** (1857), pp. 172—176 and **18** (1859), pp. 347—378.

[3] A. CAYLEY, A theorem on trees, *Quarterly Journal of Pure and Applied Mathematics*, **23** (1889) pp. 376—378.