ANALYTICAL APPROACH TO DESCRIPTION OF SOME COMBINATORIAL AND NUMBER-THEORETIC COMPUTATIVE ALGORITHMS

Y. V. CHEBRAKOV and V. V. SHMAGIN
Department of Mathematics, Technical University,
Nevsky 3-11, 191186, St-Petersburg, Russia
E-mail: chebra@phdeg.hop.stu.neva.ru

We discuss the theme on translating different descriptions of computative algorithms into high-level programming languages, enumerate some advantages of analytical descriptions and demonstrate that logical functions may be used effectively to create analytical formulae available for describing a set of combinatorial and number-theoretic computative algorithms. In particular, we adduce analytical formulae to generate l-th prime numbers p_l, permutations of order m, k-th numbers of Smarandache sequences of 1st and 2nd kinds and classical Magic squares of an order n.

Key words: computative algorithms, analytical approach, logical functions, combinatorics, number theory.

1 Introduction

As well-known [1,2] verbal and diagram (graph-diagram) techniques available for describing computative algorithms are the most wide-spread at present. For instance, *Euclidean algorithm*, allowing to find the greatest common divisor (GCD) of the positive integers a and b ($a > b$) has the following verbal description [3]

1. Assign $m = a$, $n = b$;
2. Find $r = m \bmod n$;
3. If $r > 0$, then pass to 4. Otherwise, pass to 5;
4. Assign $m = n$, $n = r$ and pass to 2;
5. Answer: GCD(a, b) = n.

Since all computative algorithms are realised, as the rule, on computer at present, the main fault of the verbal description of computative algorithms is the necessity of translating this description into one of special computer-oriented languages.

The diagram form of the description of computative algorithms allows to simplify slightly the process of such translation. In particular, the diagram form of Euclidean algorithm is shown in figure 1, where squares with digits 1, 2, 4, 5 and the rhomb with the condition $r > 0$ mean respectively to points 1, 2, 4, 5 and 3 of the verbal description.

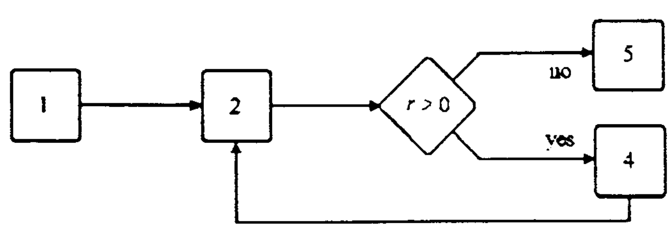

Figure 1. Diagram form of the description of Euclidean algorithm.

The logical technique[4] available for describing computative algorithms is less known than verbal and diagram descriptions, but just it gives easier way to obtain a program code. In particular, we may present the logical description of Euclidean algorithm in the form

$$A_1 \downarrow^2 A_2 \alpha_3 \uparrow_1 A_4 \beta \uparrow_2 \downarrow^1 A_5, \quad (1)$$

where A_i are elements of the vector operator $A = \{\langle m = a,\ n = b\rangle, \langle r = m \bmod n\rangle, \langle\varnothing\rangle, \langle m = n,\ n = r\rangle, \langle n\rangle\}$; α_i are elements of the vector conditional jump $\alpha = \{\langle\varnothing\rangle, \langle\varnothing\rangle, \langle r > 0\rangle, \langle\varnothing\rangle, \langle\varnothing\rangle\}$; $\langle\varnothing\rangle$ is the blank in A and α; $\uparrow_i$ and $\downarrow^i$ are arrows indicating respectively points of departures and destinations; β is the unconditional jump instruction.

We note that, generally speaking, an one-to-one correspondence exists between three foregoing techniques for describing computative algorithms. In other words these techniques are identical in substance.

One of modern techniques available for describing computative algorithms is using the built-in predicates calculus, realised, for instance, in Prolog[5]. In particular, we may represent Prolog description of Euclidean algorithm by three statements

GCD(0,V,V). (2)
GCD(NS,VS,V):–NS1 is VS mod NS, GCD(NS1,NS,V).
?–GCD(b, a, V).

Where the second statement is the direct record of the recursive computative procedure, allowing to find GCD(a, b); the first one determines the condition to finish this procedure; the third one is constructed to introduce the concrete values of numbers of Euclidean algorithm; NS, NS1, V and VS are internal variables of the procedure and V = GCD(a, b) after calculations. The main obstacle of this technique spreading is necessity of preliminary good knowledge of predicates calculus theory.

This paper is devoted to an advance of analytical approach to describing some combinatorial and number-theoretic computative algorithms. Since at present any analytical description of the computative algorithms allows to automate the process of obtaining the program code, we suppose that the discussed theme appears to be interesting.

2 Constructing analytical formulae by using logical functions

2.1 Formulae to generate n-th prime number p_n

In our view, the most impressive application of logical functions in elementary number theory is the formula[3, 6] to generate n-th prime number p_n:

$$p_n = \sum_{m=0}^{(n+1)^2+1} \mathrm{sg}\Big(n+1-\sum_{k=2}^{m}(\{(k-1)!\}^2 - k[\{(k-1)!\}^2/k])\Big), \quad (3)$$

where $p_0 = 2$, $p_1 = 3$, ...; square brackets mean the integer part; sg is a logical function: $\mathrm{sg}(x) = 1$ if $x > 0$ and $\mathrm{sg}(x) = 0$ if $x \le 0$. Let us find another analytical formula for p_n without factorials.

As well-known[3, 7], any prime number has exactly two divisors: the unit and itself. Thus, any integer number a is a prime one if it has not divisors among integer numbers from 2 to $[\sqrt{a}\,]$ or, in the language of analytical formulae, if

$$\chi_a = \prod_{j=2}^{[\sqrt{a}]} \{\mathrm{sg}(a - j[i / j])\}, \tag{4}$$

then a is a prime only when $\chi_a = 1$. It appears directly from (4) and (3) that the desirable formula for p_n has form

$$p_n = \sum_{m=0}^{(n+1)^2+1} \mathrm{sg}(n - 1 - \sum_{a=3}^{m} \chi_a), \tag{5}$$

where $p_2 = 2, p_3 = 3, p_4 = 5, \ldots$.

2.2 The analytical description of the permutations generator

As well-known[2, 3]

a) *the permutation of order m* is called any arrangement of m various objects in a series;

b) the verbal description of the simple algorithm available for constructing all the permutations from m objects, if all the permutations from $m - 1$ objects have been already constructed, has form

Enumerate $m - 1$ various objects by the numbers 1, 2, ..., $m - 1$. For each permutation of $a_1, a_2, \ldots, a_{m-1}$, containing the numbers 1, 2, ..., $m - 1$, form m other permutations by putting the number m in all the possible places. As a result obtain the permutations:

$$\begin{array}{l} m,\ a_1,\ a_2,\ \ldots,\ a_{m-1}; \\ a_1,\ m,\ a_2,\ \ldots,\ a_{m-1}; \\ \ldots\ldots\ldots\ldots\ldots\ldots; \\ a_1,\ a_2,\ \ldots,\ m,\ a_{m-1}; \\ a_1,\ a_2,\ \ldots,\ a_{m-1},\ m. \end{array} \tag{6}$$

It is evident that one can obtain all the permutations of order m by this algorithm and none of the permutations of (6) may be obtained more than once. If this verbal description we translate into one of special computer-oriented languages, for instance, into Pascal, then we obtain the program code, shown in table 1. This program works correctly at initial conditions m4 = 1; n1 = m and the array nb3 contains such numbers in the first m cells, which should be rearranged, and has following advantages over the verbal description

a) the knowledge of all permutations from $m - 1$ objects is not required for generating the permutations of order m;

b) permutations are realised with any set of numbers, contained in the first m cells of the array nb3.

Table 1. The representation of the permutations generator as Pascal program.

```
Procedure Perm(Var m4,n1,n:Integer;
Var nb3,nb4,nb5:Ten);
Label A28,A29,A30;
Var nt,k,m2:Integer;
Begin
   If m4=1 Then
   Begin
      m4:=0;n:=n1;
      For k:=2 to n do
      Begin
         nb4[k]:=0;
         nb5[k]:=1;
      End;
      Exit;
   End;
   k:=0; n:=n1;
A28:
   m2:=nb4[n]+nb5[n];nb4[n]:=m2;
   If m2=n Then
      Begin
         nb5[n]:=-1;Goto A29;
      End;
      If Abs(m2)>0 Then Goto A30;
      nb5[n]:=1;Inc(k);
A29:
   If n>2 Then
   Begin
      Dec(n);Goto A28;
   End;
   Inc(m2);m4:=1;
A30:
   m2:=m2+k;
   nt:=nb3[m2];
   nb3[m2]:=nb3[m2+1];
   nb3[m2+1]:=nt
End;
```

It should be noted that the main fault of both the verbal description and the program code is the fact that the knowledge of the previous permutation of order m is required for constructing the next permutation from m objects. To eliminate this fault one may use a set of analytical formulae [6]

$$r_j = p_j - z_1 + 1, \quad p_j = j - 1 + f(1 - c_j) + c_j\,(m - j - f), \qquad (7)$$

$$f = t_{j-1} - (m - j + 1)\,[\,t_{j-1} / (m - j + 1)],$$

$$t_j = [\,k \,/ \prod_{q=1}^{j} (m - q + 1)\,], \qquad c_j = |\,(-1)^{t_j} - 1\,| / 2,$$

$$z_1 = \mathrm{sg}(1 + p_{j-1} - p_j - z_2) + z_2, \quad z_2 = \mathrm{sg}(1 + p_{j-2} - p_j - z_3) + z_3, \;\ldots,$$

$$z_{j-1} = \mathrm{sg}(1 + p_1 - p_j),$$

where k is a number of permutation, generated of (7); r_j is a number, which j-th element of the initial sequence nb3 has in k-th permutation; the all another parameters in (7) are auxiliary.

2.3 The formula for counting the value of GCD(a, b)

We may present one of possible formulae available for counting the value of GCD(a, b) {see Sect. 1} as

$$\mathrm{GCD}(a, b) = b\{1 - \mathrm{sign}(r)\} + k\,\mathrm{sign}(r), \qquad r = a - b[a/b], \qquad (8)$$

$$k = \operatorname*{MAX}_{i=2}^{[\sqrt{b}]} \{i(1-d)\}, \quad d = \mathrm{sign}\{a - i[a/i]\} \times \mathrm{sign}\{b - i[b/i]\},$$

where the function $\mathrm{MAX}(a_1, a_2, \ldots, a_i)$ gives the greatest from numbers $a_1, a_2, \ldots, a_i$; $\mathrm{sign}(x) = |x|/x$ if $x \neq 0$ and $\mathrm{sign}(0) = 0$.

2.4 Formulae for the calculation of n-th numbers in Smarandache sequences of 1st and 2nd kinds

As we found earlier[7], the terms of six Smarandache sequences of 1st kind[8] may be computed by means of one general recurrent expression

$$a_{\varphi(n)} = \sigma(a_n 10^{\psi(a_n)} + a_n + 1), \tag{9}$$

where $\varphi(n)$ and $\psi(a_n)$ are some functions; σ is an operator. For instance,

a) if $\varphi(n) = n+1$, $\sigma = 1$ and $\psi(a_n) = [\lg(n+1)] + 1$ then the following sequence of the numbers, denoted as *S_1-sequence*, is generated by (9)

$$1,\ 12,\ 123,\ 1234,\ 12345,\ 123456, \ldots \tag{10}$$

Let each number χ_k, determined as

$$\chi_k = -1 + \sum_{j=0}^{[\lg(k+0,5)]} (k+1-10^j), \tag{11}$$

correspond to each number a_k of sequences (10), where the notation "$[\lg(y)]$" means integer part from decimal logarithm of y. Using (11) we may construct the following analytical formula for the calculation of n-th number in the S_1-sequence:

$$a_n = 10^{\chi_n} \sum_{i=1}^{n} (i/10^{\chi_i}); \tag{12}$$

b) if $\varphi(n) = n+1$; $\sigma = \gamma$ is the operator of mirror-symmetric extending the number $a_{[(n+1)/2]}$ of S_1-sequence from the right with 1-truncating the reflected number from the left, if n is the odd number, and without truncating the reflected number, if n is the even number; $\psi(a_n) = [\lg([(n+1)/2] + 1)] + 1$, then the following sequence of the numbers, denoted as *S_2-sequence*, is generated by (9)

$$1,\ 11,\ 121,\ 1221,\ 12321,\ 123321,\ 1234321, \ldots \tag{13}$$

The analytical formula for the calculation of n-th number in the S_2-sequence has the form

$$a_n = \sum_{i=1}^{[n/2]} i\, 10^{\chi_i - [\lg i]} + \sum_{i=1}^{[(n+1)/2]} i\, 10^{d}, \tag{14}$$

where $d = 1 + \chi_{[(n+1)/2]} + \chi_{[n/2]} - \chi_i$; and so on.

We find recently that the terms of Smarandache sequences of 2nd kind[8] may be computed also by the universal analytical formula {compare with formulae (12) and (5)}

$$a_n = \sum_{m=1}^{U_n} \mathrm{sg}(n+2-b-\sum_{i=b}^{m} \chi_i), \tag{15}$$

where χ_i are the characteristic numbers for Smarandache sequences of the 2nd kind; U_n is an up-estimation for the value a_n; b is a constant. For instance, if

$$U_n = n^2,\ b=2,\ \chi_i = \mathrm{sg}\,\{\sum_{k=1}^{g!} \prod_{q=1}^{[\sqrt{c}]} \mathrm{sg}\,(c-q[c/q])\,\},\ g=[\lg i]+1, \tag{16}$$

$$c = 10^g \sum_{p=1}^{g} \{\,[i/10^{g-p}] - 10\,[i/10^{g-p+1}]\,\}/10^{r_p},$$

the following sequence of Smarandache numbers of 2nd kind is generated by (15)

$$1, 2, 3, 5, 7, 11, 13, 14, 16, 17, 19, 20, 23, 29, 30, 31, 32, 34, \ldots \tag{17}$$

We note that

a) in formula (16): k is a number of the permutation, which is generated from digits of the number i; r_j is a number, which j-th digit of the number i has in k-th permutation {see (7)};

b) only such integer numbers belong to the sequence (17), which are prime numbers or can be derived to prime numbers by a permutation of digits in the initial natural numbers {the number 1 is related to prime numbers in this sequence}.

2.5 Formulae for analytical description of Magic squares constructing methods

As we discovered earlier[3, 11], logical functions may be used effectively to create analytical formulae available for describing computative algorithms on constructing Magic squares of any order n. For instance, let us consider a well-known "Method of two squares", whose verbal description has the form[3, 12]:

1. Make a drawing of two square tables of any order $n = 4k + 2$ ($k = 1, 2, \ldots$). Divide every table in four equal squares which we shall call A, B, C and D squares respectively {see figure 2(1)};

2. Place a Magic square of order $m = 2k + 1$ in the A, B, C and D squares of the first table. It is obvious {see figure 2(2)}, the first table will have the same sum of numbers in its rows, columns and main diagonals;

3. Fill the cells of the second table: all cells of A square are to have zeros; cells of D square — numbers $u = m^2$; cells of B square — numbers $2u$ and cells of C square — numbers $3u$. The obtained table {see figure 2(3)} will have the same sum of numbers only in its columns;

4. Perform such rearrangement of the numbers in the table 2(3) that the new table will have the same sum of numbers in its rows, columns and main diagonals. It can be achieved, for instance, by operations

a) underline in the square A of the second table

– k zeros, located in the extreme left positions of all rows, excepting the middle row {see figure 2(3)};

– the zero, located in the central cell of the middle row, and another $k-1$ zeros, located left of the central cell.

Exchange all marked zeros against the respective numbers of the square C {see figure 2(4)} and otherwise;

b) mark $k-1$ numbers $2u$ in the extreme right positions of every row of square B {see figure 2(3)}, and then exchange ones against corresponding numbers of the square D (see figure 2(4)) and otherwise.

5. Add (cell-wise) two auxiliary tables {the Magic square of order 10, obtained as a result of adding auxiliary squares 2(2) and 2(4) is shown in figure 2(5)}.

A	B
C	D

(1)

15	14	6	22	3	15	14	6	22	3
21	2	18	10	9	21	2	18	10	9
13	5	24	1	17	13	5	24	1	17
4	16	12	8	20	4	16	12	8	20
7	23	0	19	11	7	23	0	19	11
15	14	6	22	3	15	14	6	22	3
21	2	18	10	9	21	2	18	10	9
13	5	24	1	17	13	5	24	1	17
4	16	12	8	20	4	16	12	8	20
7	23	0	19	11	7	23	0	19	11

(2)

0	0	0	0	0	50	50	50	50	50
0	0	0	0	0	50	50	50	50	50
0	0	0	0	0	50	50	50	50	50
0	0	0	0	0	50	50	50	50	50
0	0	0	0	0	50	50	50	50	50
75	75	75	75	75	25	25	25	25	25
75	75	75	75	75	25	25	25	25	25
75	75	75	75	75	25	25	25	25	25
75	75	75	75	75	25	25	25	25	25
75	75	75	75	75	25	25	25	25	25

(3)

75	75	0	0	0	50	50	50	50	25
75	75	0	0	0	50	50	50	50	25
0	75	75	0	0	50	50	50	50	25
75	75	0	0	0	50	50	50	50	25
75	75	0	0	0	50	50	50	50	25
0	0	75	75	75	25	25	25	25	50
0	0	75	75	75	25	25	25	25	50
75	0	0	75	75	25	25	25	25	50
0	0	75	75	75	25	25	25	25	50
0	0	75	75	75	25	25	25	25	50

(4)

90	89	6	22	3	65	64	56	72	28
96	77	18	10	9	71	52	68	60	34
13	80	99	1	17	63	55	74	51	42
79	91	12	8	20	54	66	62	58	45
82	98	0	19	11	57	73	50	69	36
15	14	81	97	78	40	39	31	47	53
21	2	93	85	84	46	27	43	35	59
88	5	24	76	92	38	30	49	26	67
4	16	87	83	95	29	41	37	33	70
7	23	75	94	86	32	48	25	44	61

(5)

Figure 2. Method of two squares.

Table 2. The representation of computative algorithm "Method of two squares" as Pascal program.

```
Procedure BuildMS;
{ Build Magic square of order n=4k+2 }
Var m,i,j,n2:Byte;
  z,x,y,nn:Integer;
Begin
   z:=0;n2:=n;
   n:=n2 Shr 1;
   m:=n Shr 1;
   nn:=Sqr(n);
{ Build an auxiliary square of order n=4k+2}
   { Place a Magic square in A square }
   For z:=1 to nn do
   Begin
      x:=(fX(z)-1+n)Mod n+1;
      y:=(gY(z)-1+n)Mod n+1;
      mk[(x-1)*n2+y]:=z;
   End;
   n:=n2;n2:=n Shr 1;
   For i:=1 to n2 do For j:=1 to n2 do
   Begin
{ Place the  Magic square in B, C and D squares }
      mk[Ind(i,j+n2)]:=mk[Ind(i,j)]+nn Shl 1;
      mk[Ind(i+n2,j)]:=mk[Ind(i,j)]+nn Shl 1+nn;
      mk[Ind(i+n2,j+n2)]:=mk[Ind(i,j)]+nn;
   End;
{Modify the auxiliary square }
    { Modify A and C squares }
   For i:=1 to n2 do
   If i<>n2 Shr 1+1 Then
   Begin
      For j:=1 to m do
      Begin
         Inc(mk[Ind(i,j)],nn*3);
         Dec(mk[Ind(i+n2,j)],nn*3);
      End;
   End;
   i:=n2 Shr 1+1;
   For j:=i Downto i-m+1 do
   Begin
      Inc(mk[Ind(i,j)],nn*3);
      Dec(mk[Ind(i+n2,j)],nn*3);
   End;
    { Modify B and D squares }
   For j:=n Downto n-m+2 do
   For i:=1 to n2 do
   Begin
      Dec(mk[Ind(i,j)],nn);
      Inc(mk[Ind(i+n2,j)],nn);
   End;
End;
```

If the foregoing verbal description we translate into Pascal, we obtain the program code, shown in table 2. In this program code

a) the Magic square of order $m = 2k + 1$, located in the A, B, C and D squares of the auxiliary square, may be built for instance, by the functions [3, 11, 13]

$$f(z,\ m) \equiv (z-1) + [\,(z-1)/m\,] - [m/2], \qquad (18)$$
$$g(z,\ m) \equiv (z-1) - [\,(z-1)/m\,] + [m/2]\,,$$

where square brackets mean the integer part; a sign "≡" is the modulo m equality; z is any natural number from 1 to m^2; functions f and g afford to compute the position of any natural number z in cells of the Magic square: $x = f(z, n)$ and $y = g(z, n)$. In particular, functions (18) may be presented as following two distinct Pascal-procedures

```
Function fX(z:Integer):Integer;
Begin
   fX:=1+(z-1)+(z-1)div n - n shr 1;
End;

Function gY(z:Integer):Integer;
Begin
   gY:=1+(z-1)-(z-1)Div n + n shr 1;
End;
```

b) two procedures "Ind" and "Sign" are auxiliary and have the form

```
Function Ind(x,y:Integer):Integer;
Begin
   Ind:=(x-1)*n+y;
End;

Function Sign(n:Word):ShortInt;
Begin
   If Odd(n) Then Sign:= -1 Else Sign:=1;
End;
```

The analytical description of "Method of two squares" has the form[3]

$$x = i + c_1 m - 1; \quad y = (1 - c_3 - c_6 - c_5)(j + c_2 m) + (c_3 + c_6 + c_5)(1 + \{(j + (c_2 + 1)m - 1) \bmod n\}) - 1; \qquad (19)$$

$$u = m^2; \quad z = 1 + \{(N-1) \bmod u\}; \quad i = f(z, m) + 1;$$

$$j = g(z, m) + 1; \quad c_1 = [\{([(N-1)/u] + 1) \bmod 4\}/2];$$

$$c_2 \equiv [(N-1)/u] \bmod 2, \quad c_3 = [(\mathrm{sign}(c_1 m + i - 3k - 4) + 2)/2];$$

$$c_4 = \mathrm{asg}(j - k - 1); \quad c_6 = c_4 [(\mathrm{sign}(k - i - c_1 m) + 2)/2];$$

$$c_5 = (1 - c_4)(1 - c_1)[\{\mathrm{sign}(i-1) + 1\}/2] \times [(\mathrm{sign}(k - i + 1) + 2)/2],$$

where $n = 4k + 2$ is an order of the desirable Magic square, contained natural numbers N from 1 to n^2; $m = 2k + 1$; functions $f(z, m)$ and $g(z, m)$ are determined by the expression (18); $\mathrm{asg}(x) = 1$ if $x \neq 0$ and $\mathrm{asg}(0) = 0$ $\{\mathrm{asg}(x) = |\mathrm{sign}(x)| = \mathrm{sign}|x| = \mathrm{sg}|x|\}$.

References

1. Lyapunov A A *J. Izv. Vysh. Uch. Zav. Radiofizika* (1958) **1** (In Russian)
2. Knuth D E *Art of Computer Programming* (New York, Addison-Wesley, 1973 (1), 1981(**2**)) **1-2**
3. Chebrakov Y V *Magic squares. Number theory, algebra, combinatorial analysis* (St. Petersburg, St. Petersburg State Technical Univ. Press, 1998) (In Russian)
4. Yanov Y I *Problemy Kibernetiki* (1958) **1** (In Russian)
5. Doores J, Reiblein A R and Vadera S *Prolog – Programming for tomorrow* (Wilmslow, Sigma Press, 1987)
6. Schihanovich Y *Kvant* (1976) **5** (In Russian)
7. Vinogradov I M *Foundation of the theory of numbers* (Moscow, Nauka, 1981) (In Russian)
8. Chebrakov Y V and Shmagin V V *Some new methods for constructing analytical formulae in combinatorics* In: Proc. of Research Conf. on Algebra and Discr. Math.: Combinatorics – Algebraic, Geometric and Probabilistic Aspects (San Feliu de Guixols, 1997) 35
9. Chebrakov Y V and Shmagin V V *Smarandache Notions J.* (1997) **8** (1-2-3) 62
10. Dumitrescu C and Seleacu V *Some notions and questions in number theory* (Vail, Erhus University Press, 1995)
11. Chebrakov Y V and Shmagin V V *Different types of formulae for the analytical description of Magic squares constructing methods* In: Proc. of Conf. on Analytic and Elementary Number Theory (Vienna, 1996) 200
12. Ball W W R and Coxeter H S M *Mathematical Recreation and Essay* (Toronto, Univ. of Toronto Press, 1974)
13. Postnikov M M *Magic squares* (Moscow, Nauka, 1964) (In Russian)