

GENESIS OF THE PRIME NUMBERS

* Law of Genesis and Structure of the Prime Numbers

* The Algorithm "Method of the compass and the numerical range"

* Rudimentary implementation of the algorithm named "method of the compass and the numerical range" in language C/C++

ISMAELE ALONGI

RESERVED INTELLECTUAL AND LITERARY OWNERSHIP.

*ARRANGEMENT OF TRANSLATION FROM THE ORIGINAL ONE IN ITALIAN TO ENGLISH.
THE ORIGINAL VERSION IS IN ITALIAN LANGUAGE.*

GENESIS OF THE PRIME NUMBERS

Law of Genesis and Structure of the Prime Numbers

RESERVED INTELLECTUAL AND LITERARY OWNERSHIP

"The beauty, the wisdom, the power and the glory belong to God, the Most High, He Who makes becoming.

He gives to us the privilege and the joy to be able to know His eternal intention for the humanity."

Ismaele Alongi.

From an analysis of the prime numbers I have deduced the following Law of Genesis and Structure of the Prime Numbers:

In the numerical succession of the prime numbers, every new prime number is born in the first numerical passage left empty by the numerical progressions of the precedents known prime numbers.

Originally I had expressed this concept this way:

Every new prime number is produced in the first space left empty by the inferior numerical series.

The numerical succession of the prime numbers lies on a half-line that has origin in the field of the ended one and point to the endless one. Ideally I have imagined some numerical successions superimposed that they have origin from situated points on a straight line imaginary orthogonal to the half-line of the natural numbers and passing for the point zero. It is extraordinary to imagine all the prime numbers lined up on this imaginary straight line: virtually we imagine an immense and endless composed number that goes to fill the endless one... or it could be also the endless prime number... However is, there are interrogative that for the man they are inscrutable, really because the mind human work in the field of the ended one and, since the endless one cannot be filled with this method, to try to close the circle endless take us to error.

The idea of the new empty spaces that they are filled as from the new prime numbers, it springs from the observation of the special even prime number: the number two.

If in fact we observe the behavior of the progression of the number two, we immediately notice that it goes to fill $\frac{1}{2}$ of endless of the natural numbers. What would it be necessary to fill the rest of the endless one? A numerical progression in base one with reason two. It is not this however the logic of the succession of the prime numbers. The endless one is gradually filled, and since the endless one cannot be filled, it will never be filled, therefore for the human mind endless prime numbers exist. It will always stay in fact a first passage numerical void over the progressions superimposed of the known prime numbers: that first passage numerical void is one "zero", at all banal, as loved to say Bernhard Riemann, and that numerical void is occupied by a new prime number by to know, a new numerical planet with one precise cyclical trajectory of its, a frequency with a precise length of wave.

It doesn't want there so much to risk idea that exploring the prime numbers along the half-line of the natural numbers, we are exploring only half the endless one, the other half it is found to the mirror, on the virtual half-line of the negative numbers, therefore very probably we witness a mathematical mirage if we think about seeing on the virtual alignment that passes for the zero-origin an only and endless composed number, instead probably in the mirage there is an endless prime number.

I don't know if these my hypotheses go to show the Riemann's hypothesis, that almost seems to me a succeeded mathematical hazard, but this result that I have gotten it is born from another search that I am still developing: the factorization of the composed numbers.

I took the law of numerical succession and genesis of the prime numbers, that I have deduced, as starting point for developing an enough effective system to calculate the following prime numbers in a datum numerical interval, gives the precedents known prime numbers. This algorithm I have denominated it method of the numerical compass and the numerical range.

As element of base would be enough the fundamental number two, to get all the other prime numbers, in fact if we observe the progression of the number two, the first space numerical void is between two and four, therefore the next prime number is three. We observe then the progressions superimposed of two and of three: the next numerical passage is five etc.

To visualize better the concept I have realized a simple graph. My discovery in fact it was born in the moment in which I have begun to consider for the varied natural numbers that I have analyzed in succession, the least divisor - great of the unity - relatively to every number, therefore I have begun to glimpse some symmetries that they were cyclically repeated, almost as a fantastic numerical genetic sequence: the DNA of the prime numbers.

GENESIS OF THE PRIME NUMBERS

The Algorithm "Method of the compass and the numerical range"

RESERVED INTELLECTUAL AND LITERARY OWNERSHIP

Initial Datum

Sequence of known prime numbers.

We know the sequence of the prime numbers from 2 to "n."

Phase of calculation 0

Center of symmetry and ray of symmetry

The progressions produced by the varied prime numbers from 2 to "n" they create "symmetries", therefore we decide to consider the position of the last known prime number "n" as "center of the symmetry", or as center of the half-circumference traced that have as center the point "n". The value of the last known prime number we will also consider it as "ray of symmetry."

Elements of the progression of the number 2 inside the ray of symmetry.

How many elements will it have the progression with base 2 and reason 2 inside the ray of symmetry? The number of elements (following named "n_two") it will be equal to: ("ray_of_symmetry"-1) / 2.

How much the value of the last element of the 2 progression will amount (following named "last_two") inside the ray of symmetry?

"last_two" = 2 + ("n_two"-1) * 2.

This calculation on the formula is founded for the arithmetic progressions:

$$a_n = a_1 + (n - 1) * d;$$

where a_n is the last element of the progression, a_1 is the first element of the progression, n is the number of elements of the progression, d is the reason.

Phase of calculation 1

Modules of the incomplete trajectories

For every known prime number to calculate the module (or rest) of the division among "center of symmetry" and the examined prime number, therefore to memorize the module that represents, for the relative prime number to it associated, its incomplete trajectory inside the ray of symmetry.

Phase of calculation 2

Lacking trajectories

For every prime number to calculate difference between the examined prime number and the relative module: this difference is the lacking trajectory of the relative prime number examined inside the "numerical range" over the last known prime number, however before 2n.

Phase of calculation 3

Creation of the progressions of the odd prime numbers those are inferior to "n" inside the "numerical range" in base to the lacking trajectories

For every prime number to calculate the elements of its own progression inside the numerical range (that range is equal to the ray of symmetry) in base to the lacking trajectories and in base to the following rules:

- if the lacking trajectory is even, the base of the progression is equal to the lacking trajectory;
- if the lacking trajectory is odd, the base of the progression is equal to the lacking trajectory + the relative prime number;
- to calculate the progression up to that the last element of the progression is inferior or equal to the "two progression last element" that we have calculated in precedence ("last_two"), according to the following rule of calculation that allows us to jump the composed even numbers those are inside the numerical range: $k_y = k_x + 2 * p$; where k_y represents the following element of the examined numerical progression, k_x the precedent element of the progression in matter, p represents the prime number related to the progression that we are examining;
- we memorize the elements of the varied progressions calculated inside the "numerical range": these are the intersections with the half-line of the natural numbers.

Phase of calculation 4

Calculation of the zeros, positions of the new prime numbers

We calculate every 2 numerical progression element inside the numerical range up to the limit of symmetry, in base zero and reason 2, therefore in progression we verify the existence of intersections with the half-line of the natural numbers: if the considered element exists in the preceding numerical progressions, it wants to say that the intersection already exists with the half-line of the natural numbers, therefore we are in presence of a composed number; if instead the two progression element - that we are examining - it doesn't exist in the preceding progressions, it wants to say that we are in presence of a numerical void, an empty space or of one "zero" not banal, as Riemann would say. The sum of the position of the zero + the prime number "n" (or the last known prime number that we have also named "center of symmetry") it gives as resulted the following prime number in the succession of the prime numbers. To continue with the calculations up to the limit of symmetry, because we will very probably find further prime numbers inside the numerical range.

Notes

In the case in which we had to not have found new prime numbers inside the numerical range (which is equal to the ray of symmetry), to proceed with a new numerical range. In base to the Bertrand's postulate, however, it is reasonable to hypothesize rather that inside the numerical range equal to the ray of symmetry (or up to the position $2n$) we will always find some prime numbers.

I have performed a rudimentary implementation of this algorithm of calculation in language C/C++ and it is possible to perform the program to verify partly the functionalities of this method of calculation, or better clarifying ideas on as it must be formulated really the calculations.

Example of application of the denominated algorithm "method of the compass and the numerical range."

Datum: It's given the succession of the prime numbers from 2 to 19.

Phase of calculation 0

Center of symmetry and ray of symmetry.

```
n = 19; // n = element[8];
center_of_symmetry = 19; // center_of_symmetry = n;
```

Elements of the progression of the number 2 inside the ray of symmetry.

```
n_two = (19-1) / 2; // n_two = (n-1) / 2;
```

Phase of calculation 1

Modules of the incomplete trajectories

```
// module[i] = n% element[i];
```

```
module[2] = 19% 3; // module[2] = 1;
module[3] = 19% 5; // module[3] = 4;
module[4] = 19% 7; // module[4] = 5;
module[5] = 19% 11; // module[5] = 8;
module[6] = 19% 13; // module[6] = 6;
module[7] = 19% 17; // module[7] = 2;
```

Phase of calculation 2

Lacking trajectories

```
// LT[i] = element[i] - module[i];
```

```
LT[2] = 3 - module[2]; // LT[2] = 3 - 1; --> 2
LT[3] = 5 - module[3]; // LT[3] = 5 - 4; --> 1
LT[4] = 7 - module[4]; // LT[4] = 7 - 5; --> 2
LT[5] = 11 - module[5]; // LT[5] = 11 - 8; --> 3
LT[6] = 13 - module[6]; // LT[6] = 13 - 6; --> 7
LT[7] = 17 - module[7]; // LT[7] = 17 - 2; --> 15
```

Phase of calculation 3

Creation of the progressions of the odd prime numbers those are inferior to "n" inside the "numerical range" in base to the lacking trajectories

progression of the number 3, in base to the lacking trajectory:

```
base = 2; // LT[i] it is even, therefore: base=LT[i];
reason = 2 * element[i];
```

then we get 2, 8, 14, 20;

progression of the number 5, in base to the lacking trajectory:

```
base = 1 + element [i]; // LT[i] it is odd, therefore: base=LT[i]+element[i];
reason = 2 * element[i];
```

then we get 6, 16, 26;

progression of the number 7, in base to the lacking trajectory:

2, 16, 30;

progression of the number 11, in base to the lacking trajectory:

14, 36;

progression of the number 13, in base to the lacking trajectory:

20;

progression of the number 17, in base to the lacking trajectory:

32;

Phase of calculation 4

Calculation of the zeros, positions of the new prime numbers

In base to the phase of calculation 3, we have noticed that the followings intersections exist with the half-line of the natural numbers inside the numerical range:

2,8,14;

6,16;

2,16;

14.

The progression in base zero and reason two up to the limit of symmetry n, produces 9 elements:

2, 4, 6, 8, 10, 12, 14, 16, 18.

The followings numbers don't exist among the intersections, they are the first empty spaces or passages numerical denominated zeros or the positions of the new prime numbers inside the numerical range: 4, 10, 12, 18.

The formula therefore it is simple: zero = n + position;

// element[i] = n + position;

element[9] = n + 4; // 23 = 19 + 4;

element[10] = n + 10; // 29 = 19 + 10;

element[11] = n + 12; // 31 = 19 + 12;

element[12] = n + 18; // 37 = 19 + 18;

**ATTENTION!!! READ WITH CARE THE FOLLOWING CONDITIONS OF USE
BEFORE CONTINUING IN THE READING OF THE DOCUMENT.**

*ARRANGEMENT OF TRANSLATION FROM THE ORIGINAL ONE IN ITALIAN TO ENGLISH.
THE ORIGINAL VERSION IS IN ITALIAN LANGUAGE.*

**Accord of License of use to non exclusive title between the owner of
the intellectual ownership and the final consumer.**

(It make reference to the original Italian license)

The intellectual owner is Ismaele Alongi (following suitable as "the author"), been born in 1977 in Italy, author of the work object of the intellectual ownership that is titled "Genesis of the prime numbers - Rudimentary implementation of the algorithm named "method of the compass and the numerical range" in language C/C++".

The aforesaid work object of the intellectual ownership is constituted by the program for computer and the algorithm in it expressed and represented, both are been conceived, created and written from the author Ismaele Alongi (following suitable as "the following program for computer and the algorithm in it represented"), which work is found in the following pages of this document.

The final consumer is whoever decides to read, to use or to employ the following program for computer and the algorithm in it represented, accepting without reservations the conditions of this accord of license of use.

The following program for computer and the algorithm in it represented are granted in license of use to non exclusive title "so for as they are" and without any implicit or explicit guarantee, for one only consumer and for one only computer. The following program for computer and the algorithm in it represented they are not freeware, on the contrary they are simply shown to the public for better expressing and to make better comprehensible the ideas of the author with the purpose to communicate those ideas to the public.

Whoever doesn't agree with the conditions of this accord of license, is not authorized to read and/or to use in some way the following program for computer and the algorithm in it represented.

Who agrees instead with the conditions of the present accord of license, it arranges with the author that the final consumer won't ask some type of reimbursement for direct or indirect damages for the use proper and/or improper than contained and/or performed by the following program for computer and from the algorithm in it represented and for what is in it expressed and/or contained, neither the final consumer has to bring damages to other subjects through the use or application of the following program for computer and of the algorithm in it represented. The final consumer that agrees with the present accord of license, which consumer decides to read and/or to perform and/or to use and/or to employ in public the following program for computer and the algorithm in it represented, has to communicate to the people that directly or indirectly they will enjoy some application that derives of it, the conditions of the present accord of license and to put such people under the conditions to individually decide and to express his/her/their own decision, with the purpose to accept or to refuse the conditions of the present accord of license, therefore it will achieve the authorized use of it or less, in base to the express conditions in the present accord of license.

The author reserve to himself all methods and all way of economic use of his own intellectual ownership (in special way for the algorithm) and the final consumer that agrees with the present accord of license expressly consents to this limitation of use.

The reading and/or the use and/or the employment, in any way and with any method, of the followings program for computer and of the algorithm in it represented, those actions behave from the final consumer explicit acceptance and without reservations of the express conditions in the present accord of license of use to non exclusive title between the owner of the intellectual ownership and the final consumer.

GENESIS OF THE PRIME NUMBERS

Rudimentary implementation of the algorithm named "method of the compass and the numerical range" in language C/C++

RESERVED INTELLECTUAL AND LITERARY OWNERSHIP

```
// Copyright 2006 by Ismaele Alongi. All rights reserved.
// Tutti i diritti riservati in tutti i Paesi

// "The beauty, the wisdom, the power and the glory belong to God,
// the Most High, He Who makes becoming. He gives to us the joy and the
// privilege to be able to know His eternal intention for the humanity."
//
// Ismaele Alongi.

// One first rudimentary implementation of a verification of the "Law of Genesis and
// Structure of the prime numbers", hypothesized by Ismaele Alongi, and of the algorithm
// of the "compass and of the numerical range" of the author Ismaele Alongi.
// Other names to be proposed are: "Law of numerical succession of the prime numbers"
// or "order of numerical progression of the prime numbers"

//-----
// Notes on the operation of the program: the program uses as input a
// file of text with the initial prime numbers, therefore it produces as output
// a file of text with the prime numbers individualized in the following order
// to those initial.

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <memory.h>
#include <limits.h>
#include <values.h>
#include <float.h>
#include <iostream>
#include <fstream>

#define FALSE 0
#define TRUE 1

using namespace std;

void initialization_lists(int elements[]);
void verify_memorizing_existence(int ex, int existence[]);
void numeric_compass(int elements[]);

main ()
{
int initial_primes[1000]; // array of the initial prime numbers

initialization_lists(initial_primes); // initialization of the list of the primes
numeric_compass(initial_primes);
// function of calculation of the new prime numbers:
// rudimentary implementation, see details.
} // end main program
```

```

//-----
void numeric_compass(int elements[])
{
// declaration of variables
FILE *destination;

int modules[1000], // array of the modules
    LT[1000], // array of the lacking trajectories
    existence[1000]; // array of the intersections

int center_of_symmetry, i, j,
    base, k, n,
    n_two, last_two,
    position = 0, zero;

short intersection=FALSE;

existence[0]=0;

destination=fopen("new_primes.txt", "w");

// phase of calculation 0: center of symmetry
n=elements[0]; // nth prime known
center_of_symmetry=elements[n]; // it is the last known prime number

cout << "Performed calculation 0: center of symmetry\n";

// elements of the progression of the number two inside the ray of symmetry
n_two=(center_of_symmetry - 1) / 2; // number of elements of the
// two progression

last_two= 2 + (n_two - 1) * 2; // formula: An = A1 + (n - 1) * d;
// where An is the nth term looked for,
// A1 is the first element of the progression,
// n is the number of elements of the progr.,
// d is the reason.

cout << "Performed calculation elements of the progression of the number two"
    << "inside the ray of symmetry.\n";

for (i=2; i<=n; i++)
    cout << "Known prime numbers, "<< i << " = " << elements[i] << "\n";
    cout << "other "<< elements[20]<< "other " << elements[0]<< "\n";

for (i=2; i<=n; i++)
{
// phase of calculation 1: modules of the incomplete trajectories
modules[i]=center_of_symmetry%elements[i]; // esegue il module e memorizza
cout << " Performed calculation 1: modules of the incomplete trajectories"
    << " \n modules[" << i << "] = " << modules[i] << "\n";

// phase of calculation 2: lacking trajectories
LT[i]=elements[i]-modules[i];
// it calculates the lacking trajectories and it memorizes

cout << " Performed calculation 2: lacking trajectories \n"
    << "LT[" << i << "] = " << LT[i] << "\n";
}
}

```

```

// phase of calculation 3: creation of the numerical progressions;
// progressions of the remainders prime numbers,
// in base to the lacking trajectories
if (LT[i]%2==0)
    {base=LT[i];}
    // even case: base of progression = lacking trajectory
    else base = LT[i] + elements[i];
    // odd case:
    // base progr. = lacking trajectory + relative prime number
k=base;
cout << "base = " << base << "\n";
if (k == last_two)
    verify_memorizing_existence(k, existence);
if (k < last_two)
    { verify_memorizing_existence(k, existence);
      while (k < last_two)
          { k = k + 2 * elements[i]; // we jump the coincidences with
            // the composed even numbers
            if (k<=last_two)
                verify_memorizing_existence(k, existence);
            }; // end "while"
          }; // end "if (k< last_two)"
    }
cout << " Performed calculation 3: creation of the numerical progressions"
    << "\n progressions of the remainders prime numbers,in base to the"
    << "lacking trajectories\n";
cout << "index of the for " << i << "\n";
}; // end "for" of the phases of calculation 1, 2, 3
// phase 4: calculation of the zeros, positions of the new prime numbers
// - we calculate the numerical progression of the 2 up to the limit of symmetry
// - we verify the existence of the elements of the numerical progression:
// --if the number doesn't exist, this is the "zero", therefore we perform
// the sum: center_of_symmetry + position;
// - the cycle finishes when it is reached the limit of symmetry
// Noteworthy: it is necessary to keep on refining the program for when the new
// prime numbers will be rarer.... because I don't know if inside the
// range of the symmetry they are always there new prime numbers, even if
// the hypothesis could be risked that is always.
for (i=1; i<=n_two; i++)
    {
    position=position + 2; // we are calculating the elements of the
        // two progression, one to the time
    for (j=1; j<=existence[0]; j++)
        if (position==existence[j])
            { intersection=TRUE;
              j=existence[0];
            };
    if (intersection==FALSE) // once closed the nested "for" it calculates
        // the zero if there is the condition of void
        { zero=center_of_symmetry + position;
          fprintf (destination, "%ld\n", zero);
        }
    else intersection=FALSE;
    }

```

```

    cout << "Performed calculation 4: calculation of the zeros, positions of "
          << "the new prime numbers\n...index of the for " << i << "\n";

}; // end of the "for" of the two progression "as to fill"
   // inside the ray of symmetry

// Noteworthy: If new prime numbers are not found inside the ray
// of symmetry, it is necessary to continue the search with a new "numerical
// range", up to that we don't find them. This option is to implement.

fclose(destination);
} // end function "numeric_compass"

//-----
void initialization_lists(int elements[])
{
FILE *origin;

int x=0;

origin=fopen("known_primes.txt", "r");

while (fscanf(origin, "%ld", &elements[x+1]) !=EOF)
    x++;

fclose(origin);

elements[0]=x; // index of array

} // end procedure "initialization_lists"

//-----
void verify_memorizing_existence(int ex, int existence[])
{
short duplicated=FALSE;
int w;

if (existence[0]==0)
{ existence[1]=ex;
  existence[0]=1;
}
else
{ for (w=1; w<=existence[0]; w++)
    if (existence[w]==ex)
    { duplicated=TRUE;
      w=existence[0];
    }; // end "for--if"

    if (duplicated==FALSE)
    { existence[0]++;
      w=existence[0];
      existence[w]=ex;
    };
}; // end "else"
} // end procedure verify_memorizing_existence

```