EUCLIDEAN ALGORITM FOR RECURRENT SEQUENCES

Shamil Ishmukhametov, email: ishm@nextmail.ru
Andrey Mochalov, email: breathe@bk.ru
BulatMubarakov, email: mubbulat@mail.ru
Kazan Federal University

Abstract. The classical Euclidean Algorithm (EA) for calculation of greatest common divisor (gcd) of two integers have a numerous applications in the Number Theory, the Theory of Algorithms, Cryptography and many other areas of Mathematics. One of the applications of EA is a search of pseudo-prime and strong pseudoprime integers (see [1] – [4]).

Weremind that an integer n is called pseudoprime (psp) relative to base a if$n$ is composite, $(a,n) = 1$, and $a^{n-1}\, mod\ n = 1$. Respectively, integer n is called strong pseudoprime (spsp) relative to base a if$n$ is composite, $(a,n) = 1$, and, $a^d\, mod\ n = 1$, or, $a^{d2^i} mod\ n = -1$, where $n - 1 = 2^s \cdot d$, d is odd, and $0 \le i < s$.

Clearly, all spspintegers arepsp. Prime numbers satisfy to both definitions of pseudoprimality relative to all bases (due to Fermata's Theorem), so pseudoprimes of both kinds generalize the notion of primality.

Since, the number of spsp's is small especially relative to several bases so the property to satisfy the definition of spsp can be used to separate primes from composite numbers. This was made in so called the Miller and Rabin primality test (see [5], [6]).

Let n be an integer pseudoprime relative to bases a and b. Then,

$$n \in \gcd(a^t - 1, b^t - 1)\ where\ t|n - 1. \qquad (1)$$

So, if we can effectively calculate $d = \gcd(a^t - 1, b^t - 1)$, then we find easily pseudoprimes factoring d. Usually, this d is either prime, or has many small factors so the problem of factorization is not hard.

In most cases, we can limit ourselves by $a = 2$and $b = 3$. Let define $C_n = 3^n - 1, B_n = 2^n - 1$.Since, all C_n are even, so instead of C_n we consider $A_n = (3^n - 1)/2$.

In this paper we study a method of acceleration of EAfor the sequence $\{A_n, B_n\},\ n = 1,2,3, \dots$.

All required notions and definitions can be found in [7].

Keywordsandphrases. Euclidian algorithm; pseudoprime integers; strong pseudoprimes;
Computing Classification System 1998:G.3
Mathematics Subject Classification 2010:11A25, 11Y16

1. Introduction

In this section we explain the importance of the considered problem. In [6] we considered an algorithm for checking integers on primality using a function $\boldsymbol{\psi(n)}$.For any integer $\boldsymbol{n}$ the value $\boldsymbol{\psi(n)}$ means a least spsp-integer under the set of bases consisted of first $\boldsymbol{n}$ primes. For example, $\boldsymbol{\psi(0) = 2047}$ is the least composite integer, that is accepted by the one-round Miller-Rabin Primality Test as a probably prime number with base $\boldsymbol{a = 2}$. Function $\boldsymbol{\psi(n)}$ growths very quickly. Its values are counted up to $\boldsymbol{\psi(12)}$[4], but in order to continue the process,non-trivial computer resources are required.

One of possible ways to accelerate a search of sequent values of$\boldsymbol{\psi(n)}$ is learn to calculate quickly$\boldsymbol{d_n = gcd(3^n - 1, 2^n - 1)}$. We consider it as a recurrent sequence and try to use previously counted values to find subsequent ones.

2. INTRODUCING RECURRENT FORMULAS FOR $\boldsymbol{A_n, B_n}$.

In the abstract we introduced notations$A_n = (3^n - 1)/2, B_n = 2^n - 1$.Now weimply formulas connecting different A_n, B_n.
Note that $A_1 = B_1 = 1$. Then,

Lemma 1.*For all n* $A_{n+1} = 3A_n + 1, B_{n+1} = 2B_n + 1$.

Proof.Indeed,

$$A_{n+1} = \frac{3^{n+1} - 1}{2} = 3 \cdot \frac{3^n - 1}{2} + 1 = 3A_n + 1.$$

The second formula proved by analogy.

Lemma 2.*For allm and* n, $A_{m+n} = 3^n A_m + A_n, B_{m+n} = 2^n B_m + B_n$.

Proof. We have,

$$\begin{cases} A_{m+1} = 3A_m + 1, \\ A_{m+n+1} = 3A_{m+n} + 1 = 3(3^n A_m + A_n) + 1 = 3^{n+1} A_m + A_n \end{cases}$$

The second formula proved by analogy.

Lemma 3.*For allm and* n,

$$A_{m+n} = 2A_m A_n + A_m + A_n, B_{m+n} = B_m B_n + B_m + B_n.$$

Proof. *By lemma 2 we have,*

$$A_{m+n} = 3^n A_m + A_n = \frac{2A_m(3^n - 1)}{2} + A_m + A_n = 2A_m A_n + A_m + A_n$$

$$B_{m+n} = 2^n B_m + B_n = B_m(2^n - 1) + B_m + B_n = B_m B_n + B_m + B_n.$$

Corollary1.

$$A_{2n} = 2A_n(A_n + 1), \qquad B_{2n} = B_n(B_n + 2).$$

Corollary 2. *Let denote* $d_k = \gcd(A_k, B_k)$*.Then, for all n and k*

$$d\_n \mid d_{2n}, and\ in\ general, d\_n \mid d_{kn}$$

Below we collect all formulas together.

List 1.

1. $A_n = (3^n - 1)/2, B_n = 2^n - 1,$
2. $A_1 = B_1 = 1, A_{n+1} = 3A_n + 1, \qquad B_{n+1} = 2B_n + 1.$
3. $A_{m+n} = 3^n A_m + A_n, B_{m+n} = 2^n B_m + B_n.$
4. $A_{m+n} = 2A_m A_n + A_m + A_n, B_{m+n} = B_m B_n + B_m + B_n.$
5. $A_{2n} = 2A_n(A_n + 1), \qquad B_{2n} = B_n(B_n + 2).$
6. $A_{3n} = A_n(4A_n^2 + 6A_n + 3), \qquad B_{3n} = B_n(B_n^2 + 2B_n + 2) = B_n((B_n')^2 + 1),$ *where* $B_n' = B_n + 1.$
7. $A_{4n} = 4A_n(A_n + 1)(2A_n^2 + 2A_n + 1)$
8. $B_{4n} = B_n(B_n + 2)(B_n^2 + 2B_n + 2) = B_n'^4 - 1$
9. $A_{6n} = 2A_n(4A_n^2 + 6A_n + 3)(4A_n^3 + 6A_n^2 + 3A_n + 1)$
10. $B_{6n} = B_n(B_n^2 + 2B_n + 2)(B_n^3 + 2B_n^2 + 2B_n + 2).$

Lemma 4.*For any n values* A_n *and* B_n*can be counted by* $O(\log n)$ *operations.*

Proof. *In order to count* A_n *and* B_n *present* n *in the binary form, then use relations 4 and 5 from the list 1. For example, if* $= 11 = 1011_2$ *, then subsequently count* A_2, A_4, A_8 *using 5, then* $A_{10} = A_8 + A_2 + A_1$.

3 CALCULATION OF $d_n = gcd(A_n, B_n)$

As noticed in the introductory section, the search of pseudoprimes can be performed using formula (1). So, the problem is effectively count $d_n = \gcd(A_n, B_n)$.

The classical algorithm uses the known formula

$$\gcd(A_n, B_n) = \gcd(B_n, A_n \ mod \ B_n)$$

that allows to count $\gcd(A_n, B_n)$ using in average $2log_2 B_n \approx 2n$ iterations. Each iteration performs actions with integers of length n which quire $n \ log_2 \ n$elementary actions, so the common estimate of the task is

$$O(n^2 log_2 \ n) \qquad (2)$$

Below we consider ways to accelerate this calculation using results of calculation at previous steps.

4 USING CALCULATIONS OF SMALLER N TO COUNT D_N FOR LARGER N

Below we consider some partial cases ofcalculation d_k:

1. $k = 2n$. In this case we can use relation 5 from list 1. It allows us instead of calculation d_{2n} with numbers of length 2n, to perform three calculations $\gcd(A_n + 1, B_n)$, $\gcd(A_n, B_{n+2})$, and $\gcd(A_n + 1, B_n + 2)$ with integers of length n. Using estimate (2) we can evaluate that this saves not less than 25 percent's of time. Indeed, let C(n) denote the time complexity of calculation of d_n by the classical algorithm. Then,

$$C(n) = C \cdot n^2 \log n, \quad C(2n) = C \cdot 4n^2 \log 2 \, n,$$

fora constant C. The advantage of the trick is equal to

$$C(2n) - 3C(n) = C \cdot n^2 (\log n + 4) > \frac{C(2n)}{4}$$

2. Ifk= $3n$, then we can apply relation 6 of list 1 and calculationof C(3n) requires one calculation with integers of length 2n and two calculations with integers of length n, so the gain is

$$C(3n) - C(2n) - 2C(n) > 3n^2 \log n > \frac{C(3n)}{3},$$

that gives reduction of no less thana thirdpart of the time. Note that when we estimate gcd of pairs formed by numbers with different lengths, we count the complexity by the lower length, since after the first step of computation thelarger length diminished up to the lower length.

3. Let $k = 4n$. When using relations 7 and 8 from list 1 we replace the direct computation of d_{4n} by gcd- calculationof9 combinations of pairs

formed by a divisor of A_{4n} and a divisor ofB_{4n}. Four of them we already know, so the rest consists of 5 calculations with numbers of length 1 and one with length 2. The total gain of the trick is

$$C(4n) - C(2n) - 5C(n) =$$

$$= 16n^2 \log 4n - 4n^2 \log 2n - 5n^2 \log n > 7\,\mathrm{n}^2 \log n.$$

Thus, the gainforms almost half of the time.

4. Finally, we consider case $k = 6n$ and use formulas 9 and 10 from list 1. There are 9 possible combinations of pairs of divisors of A_{6n} and a divisor of B_{6n}. Four of them we already know, so the rest consists of 2 calculations with numbers of length 1, 2 calculations with numbers of length 2, and one with length 3. The total gain of the trick is

$$C(6n) - C(3n) - 2C(2n) - 2C(n) > 17n^2 \log n,$$

that gives a gain of almost a half of time.

Theorem 1. The total gain of application of this technology is about a quarter of time.

Proof. We count the average cost of 6 consecutive terms of parameter n, namely, C(n), C(2n),C(3), C(4n),C(5n), C(6n):

$$C_{total} > C(n) \cdot \left(0 + \frac{1}{4} + \frac{1}{3} + \frac{7}{16} + 0 + \frac{17}{36}\right) =$$

$$= \frac{C(n)}{144}(36 + 48 + 63 + 68) \approx 1{,}493C(n)$$

Thus, the average gain is equal to $1{,}493C(n)/6 = 0{,}248C(n)$.

Of course, if we extend the list 1, we can improve a little this theorem.

5 REDUCING TERMS A_n AND B_n BY SMALL DIVISORS

One of possible ways to accelerate the calculation is to divide $A_n$andB_n by possible small divisors. This cannot loss possible spsp's since their divisors are of form $kn + 1$ and lie in d_n.

Let p be an integer. The sequence $A_n \; mod \; p$ is finite and contains no more than $p - 1$ members. Let t be the number of different members of this sequence. Assume also that there is a k such that $A_k \; mod \; p \; = 0$.

Lemma 5. Let p, t *and* k be such as above. Then, $A_{k+mt} \equiv 0 \ mod\ p,$ for any $m \in N$. The same holds for B_n.

Example 1.Let p=11. Elements $A_n\ mod\ p$ forms a finite sequence {1, 4, 2, 7, 0} containing 5 different members, so t=4, k=5, and all members of kind $A_{5n+4}$are multiples of 5.

Example 2.For p=41, we find t=8, k=7,and members $A_7, A_{15}, A_{23}, \ldots$ are divided by 41.

Example 3. For p=11 elements $B_n\ mod\ p$form a sequence of length 10 with k=9. So all elements B_{9+10t} are multiples of 11. Uniting this with example 1 we get that all gcd$d_{9+10t}$are divided by 11.

6 CONCLUSION

In this paper we outlined the importance of the problem of finding pseudoprime and strong pseudoprime integers. This helps to find prime numbers required for needs of the theory numbers and cryptography. One of ways to find them is to exploit the relation (1) calculating the greatest common divisor $\boldsymbol{d_n}$ of pairs $\boldsymbol{A_n}, \boldsymbol{B_n}$, defined by recursion. To accelerate calculation of $\boldsymbol{d_n}$ we suggested to use information gathered at the previous stages of computation. This gives a common income by no less than 25 percent's.

REFERENCES

1. ИшмухаметовШ.Т., МубараковБ.Г. Ободномклассестрогопсевдопростыхчисел, Эвристическиеалгоритмыираспределенные вычисления, Самара, изд-о СамГУ, т.1, вып.1, с.69-78
2. C. Pomerance,C.Selfridge, S. Wagstaff. The Pseudoprimes to 25*10^9// Math. Comput. – 1980.– P. 1003-1026
3. G.Jaeschke. On Strong Pseudoprimes to Several Bases // Math. Comput. 61. – 1993. – p. 915-926
4. J .Jiang, Y. Deng.Strong pseudoprimes to the first 9 prime bases // ArXiv:1207.0063v1 [math.NT].– 2012.–P.1– 12
5. M. Rabin. Probabilistic algorithm for testing primality //J.Numb.Theory. – 12, № 1, 1980. – P. 128-138
6. S. Ishmukhametov, B.Mubarakov. On practical aspects of the Miller-Rabin Primality Test // Lobachevskii Journal of Mathematics . – v.34, №4, 2013. – P.304– 312
7. R. Crandall, C. Pomerance. The prime numbers: a computational perspective. – 2-ed., Springer-Verlag, Berlin, 2005. – 604 P.