OPEN ACCESS Freely available online

Journal of Theoretical & Computational Science

Research Article

About Efficient Algorithm for Factoring Semiprime Number

Yonatan Zilpa

Department of Mathematics, The Hebrew University of Jerusalem, Jerusalem, Israel

ABSTRACT

The complexity needed to factor large semiprime numbers is in the heart of public key cryptography. It is very important to identify cases where semiprime factorization can be done efficiently. This article introduce mathematical method for semiprime factorization. Hopefully it will help researchers to close further gaps and make public key cryptography safer [1,2].

INTRODUCTION

Let M be any semi-prime number and let $1 < p < q$ be any positive integers such that $pq = M$. For any nonzero integer n we define

$$\delta_n = q^2 - np.$$

Then

$$q^2 - n\frac{M}{q} = \delta_n$$

Multiplying both sides by q we get the following cubic equation

$$q^3 - \delta_n q - nM = 0 \tag{1.1}$$

Solving (1.1) for q we get the following vector of solutions

$$A_n = \begin{bmatrix} -\frac{\delta_n}{\sqrt[3]{-\frac{27Mn}{2}+\frac{\sqrt{729M^2n^2-108\delta_n^3}}{2}}} - \frac{\sqrt[3]{-\frac{27Mn}{2}+\frac{\sqrt{729M^2n^2-108\delta_n^3}}{2}}}{3}, \\ -\frac{\delta_n}{\left(-\frac{1}{2}-\frac{\sqrt{3}i}{2}\right)\sqrt[3]{-\frac{27Mn}{2}+\frac{\sqrt{729M^2n^2-108\delta_n^3}}{2}}} - \frac{\left(-\frac{1}{2}-\frac{\sqrt{3}i}{2}\right)\sqrt[3]{-\frac{27Mn}{2}+\frac{\sqrt{729M^2n^2-108\delta_n^3}}{2}}}{3}, \\ -\frac{\delta_n}{\left(-\frac{1}{2}+\frac{\sqrt{3}i}{2}\right)\sqrt[3]{-\frac{27Mn}{2}+\frac{\sqrt{729M^2n^2-108\delta_n^3}}{2}}} - \frac{\left(-\frac{1}{2}+\frac{\sqrt{3}i}{2}\right)\sqrt[3]{-\frac{27Mn}{2}+\frac{\sqrt{729M^2n^2-108\delta_n^3}}{2}}}{3} \end{bmatrix}$$

Substituting δ_n with x in A_n we get

$$B_n = \begin{bmatrix} -\frac{x}{\sqrt[3]{-\frac{27Mn}{2}+\frac{\sqrt{729M^2n^2-108x^3}}{2}}} - \frac{\sqrt[3]{-\frac{27Mn}{2}+\frac{\sqrt{729M^2n^2-108x^3}}{2}}}{3}, \\ -\frac{x}{\left(-\frac{1}{2}-\frac{\sqrt{3}i}{2}\right)\sqrt[3]{-\frac{27Mn}{2}+\frac{\sqrt{729M^2n^2-108x^3}}{2}}} - \frac{\left(-\frac{1}{2}-\frac{\sqrt{3}i}{2}\right)\sqrt[3]{-\frac{27Mn}{2}+\frac{\sqrt{729M^2n^2-108x^3}}{2}}}{3}, \\ -\frac{x}{\left(-\frac{1}{2}+\frac{\sqrt{3}i}{2}\right)\sqrt[3]{-\frac{27Mn}{2}+\frac{\sqrt{729M^2n^2-108x^3}}{2}}} - \frac{\left(-\frac{1}{2}+\frac{\sqrt{3}i}{2}\right)\sqrt[3]{-\frac{27Mn}{2}+\frac{\sqrt{729M^2n^2-108x^3}}{2}}}{3} \end{bmatrix}$$

Define

$$t_n(x) = \sqrt[3]{\sqrt{729n^2M^2 - 108x^3} - 27nM} \tag{1.2}$$

and define

$$u_n(x) = \sqrt{729n^2M^2 - 108x^3}. \tag{1.3}$$

Let Δ_n be the discriminant of equation (1.1), then $\Delta_n = 4\delta_n^3 - 27n^2M^2$. We assume that equation (1.1) has only one real solution, therefore

$$\Delta_n < 0$$

Correspondence to: Yonatan Zilpa, Department of Mathematics, The Hebrew University of Jerusalem, Jerusalem, Israel, Tel: 0587112358; E-mail: yz11235@gmail.com

Received: January 24, 2021; **Accepted:** August 27, 2021; **Published:** September 08, 2021

Citation: Zilpa Y (2021) About Efficient Algorithm for Factoring Semiprime Number. J Theor Comput Sci Open Access 7: p053

or equivalently

$$\delta_n < \sqrt[3]{\frac{27}{4}(nM)^2} \tag{1.4}$$

Denote $q_n(x) = B_n[0]$, then we get a function q_n of x defined by

$$q_n(x) = -\left(\frac{\sqrt[3]{2}x}{t_n(x)} + \frac{t_n(x)}{\sqrt[3]{54}}\right) \tag{1.5}$$

and since q is the solution of $x^3 - \delta_n x - nM = 0$, we get

$$q_n(\delta_n) = q. \tag{1.6}$$

Now let

$$q - p = \delta_0$$

then

$$q - \frac{M}{q} = \delta_0$$
$$q^2 - \delta_0 q - M = 0.$$

We thus get

$$q = \frac{\sqrt{\delta_0^2 + 4M} + \delta_0}{2}$$

Now we define a function q_0 such that

$$q_0(x) = \frac{\sqrt{x^2 + 4M} + x}{2}$$

similarly we define a function p_0 such that

$$p_0(x) = \frac{\sqrt{x^2 + 4M} - x}{2}$$

Consider the following system of equations:

$$\begin{cases} (q_0(x))^2 - n \cdot p_0(x) = y \\ q_n(y) - \frac{M}{q_n(y)} = x. \end{cases} \tag{1.7}$$

Clearly $(x, y) = (\delta_0, \delta_n)$ is a solution for system (1.7). In addition, substituting x, in the first equation (system 1.7) with the left side of the second equation (system 1.7) we get

$$\left(q_0\left(q_n(y) - \frac{M}{q_n(y)}\right)\right)^2 - n \cdot p_0\left(q_n(y) - \frac{M}{q_n(y)}\right) = y \tag{1.8}$$

Equation (1.8) has only one variable y, by solving this equation for y we can easily recover q. Thus the problem of factoring semiprime M is equivalent for finding the zero(s) of the function

$$f_n(x) = \left(q_0\left(q_n(x) - \frac{M}{q_n(x)}\right)\right)^2 - n \cdot p_0\left(q_n(x) - \frac{M}{q_n(x)}\right) - x. \tag{1.9}$$

Since $f_n(\delta_n) = 0$, one of the zeros of f_n must be δ_n. By finding δ_n and Plugging it into q_n (see (1.6)) we can recover q and thus factor M.

2 Choosing n

We want to choose n in such a way that f_n would be monotonic in some interval that contains δ_n. This way δ_n would be the only solution of $f_n(x) = 0$. From (1.5) we get

$$q_n'(x) = -\left(\frac{\sqrt[3]{2}t_n(x) - \sqrt[3]{2}xt_n'(x)}{t_n^2(x)} + \frac{t_n'(x)}{\sqrt[3]{54}}\right)$$

$$= \frac{-\sqrt[3]{2}t_n(x) + \sqrt[3]{2}xt_n'(x)}{t_n^2(x)} - \frac{t_n'(x)}{\sqrt[3]{54}}$$

and from (1.2) we get

$$t_n'(x) = \frac{1}{3}\left(\sqrt{729n^2M - 108x^3} - 27nM\right)^{-2/3} \cdot \frac{1}{2}\left(729n^2M - 108x^3\right)^{-1/2} \cdot (-3 \times 108)x^2$$

$$= \frac{-54x^2}{t_n^2(x)u_n(x)}.$$

Thus

$$q_n'(x) = \frac{54x^2}{\sqrt[3]{54}t_n^2(x)u_n(x)} - \frac{\sqrt[3]{2}x^3}{u_n(x)t_n^4(x)} - \frac{\sqrt[3]{2}}{t_n(x)}$$

$$= \frac{(54)^{2/3}x^2}{t_n^2(x)u_n(x)} - \frac{\sqrt[3]{x^3}}{u_n(x)t_n^4(x)} - \frac{\sqrt[3]{2}}{t_n(x)}$$

In addition

$$f_n'(x) = 2q_0\left(q_n(x) - \frac{M}{q_n(x)}\right)q_0'\left(q_n(x) - \frac{M}{q_n(x)}\right) \cdot \left(q_n'(x) + \frac{Mq_n'(x)}{q_n^2(x)}\right)$$
$$-np_0'\left(q_n(x) - \frac{M}{q_n(x)}\right) \cdot \left(q_n'(x) + \frac{Mq_n'(x)}{q_n^2(x)}\right) - 1$$

$$= q_n'(x)\left(1 + \frac{M}{q_n^2(x)}\right)\left(2q_0\left(q_n(x) - \frac{M}{q_n(x)}\right)q_0'\left(q_n(x) - \frac{M}{q_n(x)}\right) - np_0'\left(q_n(x) - \frac{M}{q_n(x)}\right)\right) - 1$$

From this we can easily see that $q'(x) < 0$ implies the monotonic decreasing of f_n.

Assuming that $u_n(x) > 0$, the following inequalities are equivalent

$$q_n'(x) < 0$$

$$\frac{54x^2}{3 \cdot \sqrt[3]{2}t_n^2u_n(x)} - \frac{\sqrt[3]{2}}{t_n(x)} - \frac{\sqrt[3]{2} \cdot 54x^3}{t_n^4(x)u_n(x)} < 0$$

$$\frac{3^2 \cdot \sqrt[3]{4}x^2}{t_n^2(x)u_n(x)} - \frac{\sqrt[3]{2}}{t_n(x)} - \frac{\sqrt[3]{2} \cdot 54x^3}{t_n^3(x)u_n(x)} < 0$$

$$3^2 \cdot \sqrt[3]{4}x^2t_n^2(x) - \sqrt[3]{2}t_n^3(x)u_n(x) - \sqrt[3]{2} \cdot 3^3 \cdot 2x^3 < 0$$

$$(\sqrt[3]{1458})x^2t_n^2(x) - t_n^3(x)u_n(x) - 54x^3 < 0$$

Thus, inequality

$$(\sqrt[3]{1458})x^2t_n^2(x) - t_n^3(x)u_n(x) - 54x^3 < 0 \tag{2.1}$$

implies the monotonic decreasing of f_n.

To find all x that satisfy inequality (2.1) we refer to the following two cases

a. $n \geq 1$ and $t_n^2(x) > x$

b. $n \geq 1$ and $t_n^2(x) \leq x$

Case (a): Assuming that $x > 0$, we get

$$(\sqrt[3]{1458})x^2t_n^2(x) - t_n^3(x)u_n(x) - 54x^3 < \sqrt[3]{1458}x^3 - t_n^3(x)u_n(x) - 54x^3$$
$$< 0.$$

Case (b): Assuming that $x > 0$ and $u_n(x) \geq \sqrt[3]{1458}t_n(x)$, then

$$\sqrt[3]{1458}x^2t_n^2(x) - t_n^3(x)u_n(x) - 54x^3 < \sqrt[3]{1458}x^2t_n^2(x) - x^2t_n(x)u_n(x) - 54x^3$$
$$< \sqrt[3]{1458}x^2t_n^2(x) - x^2t_n^2(x)\sqrt[3]{1458} - 54x$$
$$< 0.$$

In both cases inequality (2.1) is satisfied. However, in both of these cases we assumed that $x > 0$ and $u_n(x)$ is real. In the second case we also assumed that

$$u_n(x) \geq \sqrt[3]{1458}t_n(x). \tag{2.2}$$

Now $u_n(x)$ is real if

$$729n^2M^2 - 108x^3 \geq 0$$

and this equivalent to

$$x \leq \frac{3 \cdot \sqrt[3]{n^2M^2}}{\sqrt[3]{4}}$$

Cubing both sides of inequality (2.2) we get

$$u_n^3(x) \geq 1458\left(u_n(x) - 27nM\right)$$

$$u_n^3(x) - 1458u_n(x) \geq -(27 \cdot 1458)nM$$

$$u_n(x)\left(u_n^2(x) - 1458\right) \geq -39366nM \tag{2.3}$$

Since $u_n(x)$ is real, it must be nonnegtive, therefore inequality (2.3) must be satisfied. The equivalence of inequalities (2.3) and (2.2) implies that inequality (2.2) is also satisfied. So inequality (2.1) must hold true for any $x \in \left[1, 3 \cdot \sqrt[3]{\left(\frac{nM}{2}\right)^2}\right]$, therefore f_n is monotonically decreasing in

$$I_n = \left[1, \sqrt[3]{27\left(\frac{nm}{2}\right)^2}\right].$$

We also know that $\delta_n = q^2 - np$ so the larger $n \geq 1$ grows the smaller δ_n gets. We only need to find δ_n in the interval I_n. Notice that the larger n grows the larger I_n expands and, as mentioned, δ_n is getting smaller. So we can start from $n = 1$ and increase n as needed until δ_n is to be found in I_n.

Notice that the upper bound of I_n is coincide with the upper bound of δ_n that appeared in inequality (1.4).

3 Factoring Algorithm

Suppose we have chosen n and suppose that $\delta_n \in I_n = \left[1, \sqrt[3]{27\left(\frac{nM}{2}\right)^2}\right]$, then f_n is monotonically decreasing in I_n, so we can find $\delta_n = q^2 - np$ by running the following procedure.

1. Let $a = 1$
2. Let $b = \sqrt[3]{27\left(\frac{nM}{2}\right)^2}$
3. Let $I = [a, b]$
4. Let $\mu = \lfloor \frac{a+b}{2} \rfloor$.
5. If $f_n(\mu)$ is zero, then return $q_n(\mu)$ (the algorithm stops)
6. If μ is equal to a, then return $q_n(b)$ (the distance between a and b is one, and from (5) we know that δ_n is not a)
7. If $f_n(\mu) > 0$, then $b = \mu$
   Else $a = \mu$
8. Return to step (3)

In general we may start from $n = 1$ and increase n by one if needed. In this case our algorithm (written in CLRS pseudocode) would look like this:

SPF(M)

```
1  n ← 0
2  while TRUE
3        do
4           n ← n + 1
5           b ← ∛(27 (nM/2)^2)
6           a ← 1
7           c ← ⌊(a+b)/2⌋
8           while a + 1 < b
9                 do
10                   if f_n(c) = 0
11                      then return q_n(c)
12                   if f_n(c) > 0
13                      then b ← c
14                      else  a ← c
```

If n happened to be small then this semiprime factor algorithm may return answer very fast. Each increase of n by one increase the running time by

$$\lg\left(\sqrt[3]{27\left(\frac{nM}{2}\right)^2}\right).$$

We may improve this algorithm by starting from positive integer that is greater than one. Simply start the algorithm with the first positive integer n such that $\Delta n < 0$. The running time of this algorithm may vary according to the semiprime number that needs to be factor. In particular the gap between the prime factors should be chosen in such a way that n (in our algorithm) is significantly large. I hope that this will help researchers to make public key cryptography more robust and secure [3-5].

REFERENCES

1. Xingbo WA. Strategy for algorithm design in factoring RSA numbers. IOSR Journal of Computer Engineering (IOSR-JCE). 2017;19(3):1-7.
2. Li J. A parallel probabilistic approach to factorize a semiprime. American Journal of Computational Mathematics. 2018; 13:8(2): 175-83.
3. da Silva JC. Factoring semiprimes and possible implications for RSA. In2010 IEEE 26-th Convention of Electrical and Electronics Engineers in Israel 2010; 17.
4. Overmars A, Venkatraman S. Mathematical Attack of RSA by Extending the Sum of Squares of Primes to Factorize a Semi-Prime. Mathematical and Computational Applications. 2020;25(4):63.
5. Mishra M, Chaturvedi U, Pal SK. A multithreaded bound varying chaotic firefly algorithm for prime factorization. In2014 IEEE International Advance Computing Conference (IACC) 2014; 21: 1322-1325.