The Factorisation of the RSA numbers using a Factorisation by Base Algorithm (FBA), A case study

The Factorisation of the RSA numbers using a Factorisation by Base Algorithm (FBA), A code implementation

Abstract

The RSA encryption scheme uses two large numbers of similar size and the larger the numbers, the greater the cryptography strength and the more difficult it is to factorise. Methods such as Simplified Quadratic Sieve (SIQS), Lenstra and Lenstra Algorithm (LLL) and Number Field Sieve (NFS) have shown that factorising very large RSA numbers of more than 200 digits is difficult due to the large space and time complexity required. In this work, we introduce a factorisation by base algorithm (FBA) and test it on the RSA-numbers defined by the RSA challenge. We analyse the time spent on factoring to check the speed and efficiency of the algorithm.

Introduction

Integer factorization is the task of computing the divisors of natural numbers. It is a problem with a long and fascinating history, and it is certainly among the most influential in algorithmic number theory. While there is a variety of algorithms significantly faster than the brute-force search for divisors, it is still an open problem to construct a technique that efficiently factors general numbers with hundreds to thousands of digits. The hardness of this problem is fundamental for the security of widely used cryptography schemes, most prominently the RSA cryptosystem. Nevertheless, there is no proof for its hardness besides the fact that decades of efforts have failed to construct a more efficient technique. Quite regularly, there are set new records1 concerning the factorization of numbers of certain size, mostly due to improved implementations of the best available algorithms and advances in the hardware and computing power. In addition, the bound for the deterministic integer factorization problem has been improved multiple times in recent years ([11], [12], [10], [14]). On the other hand, there has only been little progress in the development of new techniques for practical integer factorization since the invention of the Number Field Sieve ([19]) in the 1990s. One of the earlier algorithms with sub-exponential runtime was by Dixon ([7]) in 1981.

Background

Let N be the number we want to factorize. We will always assume that N is odd, composite and not a perfect power of another number. We will also assume that N is made up of 2 prime factors, p and q. We will explain properties of factorisation by bases techniques using one number in base 10.

Lemma: N can be converted into a polynomial given a base.

Example: 667 in base 10 is

Now 667 is built by two prime factors which are 23 and 29. If both of them are converted into base 10 they become, and respectively.

which is different from the first form of N gotten. (A1)

From this derivation we notice that cannot be factorised to give us p and q because the discriminant was imaginary. Furthermore, we also notice that for any converted form of N we get we cannot factorise if also the discriminant is imaginary. From (A1) we noticed that a change of the form happened with h=2 and k=2 (A2) which results in, . This becomes the obtained equation.

If we say

We obtain the discriminant: which reduces to

(A3)

To solve such a problem is quite complex. We want to reduce the equation to 2 variables which can be easily solved either as Diophantine equations or through some ingenious method. There are many ways to do this.

First given Here if we insert a value for for example, 12 we get: which is a conic section. Methods of solving conic sections such as SIQS and ECM, involve the factorisation of a number to its prime factors of which this returns back to not solving the problem.

Another choice is that of given Let if we say M=34, we get We will now only be left with finding m of which by Diophantine solutions, we know that m is of a specific parametric solution. However, this method is difficult since predicting the exact values of M which will give us prime solutions is difficult and iterating all possible solutions is quite a slow method.

To solve all these problems, we present the solution below.

We propose that we can find deterministic solutions even if we use the area under a cave of f(X), as long as it is bound within some roots.

Let’s take for example, four situations in which gave us prime solutions.

(V1)

(V2)

(V3)

(V4)

From previous examples, we noticed that for most functions in which we got p,q = 1 or N, if we added or subtracted such functions from that in which we get p,q not equal to N or 1, we would derive deterministic prime solutions. Let’s now apply this for the integration, ignoring the constant. Let be the area between one of the deterministic roots against that when x is the base value.

Using V1

Meaning the area can be between or

For the area bound between we get:

For the area bound between we get:

We notice that and

And using V3

Meaning the area can be between or

For the area bound between we get:

For the area bound between we get:

We notice that and

And using V4

Meaning the area can be between or

For the area bound between we get:

For the area bound between we get:

We notice that and

Using the above scenarios, we notice that the area bound between x=base and x=root will give a prime solution which is factor of N. This can be expressed below as:

In this situation we try to solve using the simplest possible solutions. One way is to input the value of then iterate. In one situation we found for V1, hence we can assume that such a value must exist for some .

Let

and if equal to zero we obtain.

and to remove the fractions, we get.

then as for the discriminant:

of which m was found to be zero.

This implies that: of which solving it like this is more difficult as it resorts us to the first case scenario (CASE I). Hence, we combine this solution with the discriminant solution of to reach a consensus.

(T16)

(T17)

results in:

where

when

when

when

when

Using the above cases, we notice that the value of h remained constant regardless of the value of m. And when letting , we got which gave us a number of parametric solutions sets. One of them was .

In the next scenario we notice an anomaly in the sense that if we get some value of k, m and h then substitute them in we get an which does not satisfy some of the requirements we stated should happen for a full factorization, but even so, if we continue with the formulas stated we return with the correct value for p.

Example:

Given .

When

It results in such that

And such that

Also if we use we get and expanding the number, we find out that the which confirms our proposition.

THE ALGORITHM