Integer Factorization Is Not in P: Logical Proof

Rama Garimella

July 1, 2024

INTEGER FACTORIZATION IS NOT IN P:

LOGICAL PROOF:

Garimella Rama Murthy,

Professor, Department of Computer Science,

Mahindra University, Hyderabad, India

ABSTRACT

In this research paper, the problem of testing whether a given integer is a Highly Composite Number (HCN) is formulated. Using Ramanujan's theorem, it is reasoned that the unique factorization of Highly Composite Number cannot be performed using a polynomial time algorithm. Since the class of Highly composite numbers is countably infinite, integer factorization is not in class P. Also, determination of whether Integer Factorization is an NP complete problem is discussed.

1. INTRODUCTION:

Prime numbers stimulated the curiosity of generations of mathematicians. Euclid provided an elegant proof that primes are infinite in number (countably infinite). Eratosthenes provided the so called "sieve algorithm" to determine whether a given number is prime or not. But from the point of view of computational complexity theory (in Theoretical Computer Science), it can easily be proved that the "Eratosthenes Sieve algorithm" is NOT a polynomial time algorithm for determining whether a given number is prime number. Fermat proved an interesting theorem (so called Fermat's Little Theorem) which provides a necessary condition (but not sufficient condition) for determining whether a given number is prime. This theorem enabled researchers to design "probabilistic algorithms" which are of polynomial time complexity. Manindra Agrawal and his students successfully designed a test (so called AKS primality test) which provides a polynomial time algorithm for determining whether a given integer is prime or not [1].

Fundamental theorem of arithmetic provides a result that every natural number (integer) can be expressed as a product of powers of primes in a unique way. The problem of factorization of integers is capitalized to design secure public key cryptography algorithms. In theoretical computer science, an important problem is to determine whether there is a polynomial time algorithm for factoring a given natural number (integer) into product of powers of primes. It is still an unsolved problem. An interesting contribution to this research problem was made by Peter Shor. He designed a polynomial time algorithm for factoring integers on a quantum computer.

The author became interested in this research problem some number of years ago. The "integer factorization" problem was attempted by several researchers. Some interesting contributions are reported in [6]. It has been reasoned that the problem is NP hard. But it has not been proven to be NP complete.

This research paper is organized as follows. In Section 2, the known research literature is briefly reviewed. In Section 3, we formulate the novel problem of testing whether a given integer is a Highly Composite Number. In Section 4, our approach to address the question of finding a polynomial time algorithm for factoring integers is discussed. The research paper concludes in Section 5.

2. Review of Research Literature:

Because of its utility in many applications (particularly cryptography), integer factorization problem was subjected to intense investigation. In 2019, Fabrice Boudot, Pierrick Gaudry, Aurore Guillevic, Nadia Heninger, Emmanuel Thomé and Paul Zimmermann factored a 240-digit (795-bit) number (RSA-240) utilizing approximately 900 core-years of computing power [6]. Detailed literature information is available from the WIKIPEDIA PAGE and is avoided for brevity.

3. Testing for a Highly Composite Number:

Ramanujan formally stated the following definition and investigated the related characterization problem.

Definition: An integer is called "Highly Composite", if the number of divisors (prime divisors as well as composite divisors) of it are larger than those of any integer below it.

Note: Wikipedia states that Plato had this idea while counting the number of citizens of the city of Athens.

Ramanujan proved a necessary condition for an integer to be highly composite. We now state the Theorem of Ramanujan [4]

Theorem: An integer N is a highly composite only if its unique factorization is of the form:

$$N = 2^{a_2}\, 3^{a_3} \ldots \ldots p^{a_p} \text{ , where}$$

$$a_2 \geq a_3 \geq \cdots . a_p = 1.$$

It readily follows that there are infinitely many Highly Composite Numbers.

We now formulate a novel problem in the spirit of "PRIMALITY" testing:

Problem: Design an algorithm to determine whether a given number is "Highly Composite".

Thus, we have formulated the problem of "Highly Composite Number Testing" (HCN TESTING).

Note: We now provide an algorithm to determine whether a given number is highly composite:

ALGORITHM for HCN TESTING:

Step 1: Divide N by 2.

IF the remainder is NOT ZERO, then N is NOT a Highly Composite Number (HCN)………………….STOP the algorithm

Else,

divide N by higher powers of 2 and determine the highest exponent of 2 dividing N.

Label the highest exponent . Let c1=a2.

Step 2: Let i=2 and qi=3. Divide N by qi.

IF the remainder is NOT ZERO, then N is NOT a Highly Composite Number (HCN)……………………….STOP the algorithm

ELSE

Divide N by higher powers of qi and determine the highest exponent of qi dividing N.

Label such highest exponent as ci.

Step 3: Check if ci >= c i-1.
IF NOT, N is NOT a HCN……….STOP the algorithm

ELSE

Check IF qi > Square root (N). If YES STOP THE ALGORITHM

ELSE set i= i+1 and set qi be the HIGHER prime NEXT to qi

GO TO STEP 2.

If the integer N is HIGHLY COMPOSITE NUMBER, the INTEGER FACTORIZATION of N involves 2 with exponent a2 and other CONSECUTIVE PRIMES with exponents 'ci'.

NOTE: By the THEOREM of Ramanujan, any algorithm to determine whether N is HCN MUST ESSENTIALLY INVOLVE the above STEPS. This inference follows since the definition of HCN requires checking whether every prime factor of N below Square root (N) divides it

NOTE: The computational complexity of the algorithm depends on the number of prime divisors utilized for testing. This number is upper bounded by SQURE ROOT of N and lower bounded by { N divided by { 2 power a2 } }. The number of prime divisors will be exponential (as discussed below).

We now address the following question:

Q: Is there a polynomial time algorithm for determining the "unique factorization" of any integer ?

Our approach to providing an answer to the above problem is provided in the following section.

4. Integer Factorization is NOT in the Complexity Class P: Logical Proof:

Integer factorization of an integer involves determining its prime divisors and the exponents of prime divisors. For example, given N such that

$$N = p_1^{n_1} p_2^{n_2} \dots p_l^{n_l},$$

we need to determine $\{p_i's,\ n_i's\ \}$.

Goal: To show that integer factorization is not in P, it is sufficient to identify a class of integers for which no polynomial time algorithm exists that will provide the prime factorization.

We now reason that highly composite numbers are such class of numbers.

First we provide a logically coherent explanation of why a polynomial time algorithm cannot exist. In case "prime factorization is in P", for ANY GIVEN INTEGER, its unique factorization can be achieved in polynomial time.

Suppose the given number N is highly composite number. Its prime factorization consists of all primes starting with 2 and continuing upto a higher prime, p (less than $\sqrt{N}$). For determining the "unique factorization" of Highly composite N, we need to effectively determine the value of p (since all primes upto 'p' are divisors of N) and the exponents of all primes until 'p'. It is clear that once the value of 'p' is known, determination of exponents of all prime divisors can be carried out in polynomial time. In this effort we readily invoke the "prime number theorem" from analytic number theory. It is well known that by Prime Number Theorem, the number of primes until an integer M is of order

O(M) = { M/ log (M) }. Let M be { 2 power y } i.e. number of bits needed to represent M in the computer is exponential in { y }. Hence, using the UPPER AND LOWER BOUNDS on the NUMBER OF PRIMES IN UNIQUE FACTORIZATION of a HCN, it readily follows that the COMPUTATIONAL COMPLEXITY OF ANY ALGORITHM for HCN TESTING is EXPONENTIAL.

THUS NO POLYNOMIAL TIME ALGORITHM EXISTS FOR HCN TESTING. CONSEQUENTLY, UNIQUE FACTORIZATION OF AN INTEGER IS NOT IN "P".

NOTE: MATHEMATICALLY FORMAL PROOF WITH SUITABLE NOTATION BASED ON THE ABOVE LOGICAL PROOF READILY FOLLOWS.

- **Towards Determination of NP-Completeness of Integer Factorization Problem**:

  Suppose an algorithm provides an integer factorization solution for a given integer. We would like to verify whether the solution (provided by the algorithm) is correct or not assuming that we know the correct factorization of a given integer.

  Suppose the solution provided is given by

$$p_1^{n_1} p_2^{n_2} \ldots p_l^{n_l}.$$

  Let our input integer be given by

$$N = q_1^{m_1} q_2^{m_2} \ldots q_k^{m_k}.$$

We now reason that the verification of correctness of integer factorization can be done by a polynomial time algorithm. The algorithmic steps are provided below:

Step 1: Counting the number of Prime divisors:

Check if k = l. Based on the prime divisors provided by the algorithm, the counting step involves S (where is S = maximum { k, l }) number of additions. If $k \neq l$, then the algorithmic output is WRONG and the algorithm STOPS, ELSE (k = l) proceed to Step 2.

Step 2: Determining whether the prime divisors of provided solution and correct solution match:

Let i=1 and j=1, COUNT= 0

100: check if $p_i = q_j \ \ and \ \ n_i = m_j$

If yes, i=i+1, j=1, GO TO 100

Else if COUNT = k+1, STOP AND EXIT

Else j = j + 1, GO TO 100

Note: This step requires atmost $2\,k^2$ $operations\ (\,substractions\,)$.

THUS, THE STRAIGHT FORWARD VERIFICATION (OF FACTORIZATION SOLUTION) ALGORITHM REQUIRES POLYNOMIAL NUMBER OF OPERATIONS.

Note: The above algorithm provides a POLYNOMIAL TIME VERIFICATION FOR DETERMINING WHETHER A SOLUTION TO THE NP HARD PROBLEM (OF FACTORIZATION OF AN INTEGER) IS CORRECT OR WRONG. HENCE, THE DISCUSSION LEADS TO AN APPROACH TOWARDS RESOLUTION OF THE CONJECTURE P = NP.

5. CONCLUSION:

In this research paper, the problem of determining whether a given integer is Highly composite is formulated (HCN testing problem unlike Primality testing). It is reasoned (based on Ramanujan's Theorem) that HCN testing of an integer cannot be carried out using polynomial time algorithm. Thus, integer factorization is not in CLASS P. Also, the question of NP completeness of Integer Factorization problem (well known to be NP Hard) is discussed. Hence, the research paper provides a contribution towards resolution of the P=NP conjecture.

REFERENCES:

1. M. Agrawal, N. Kayal and N. Saxena,"Primes is in P," Annals of Mathematics, pp. 781-793, 2004
2. George Andrews,"Number Theory," 1971, Dover Publishers, New York
3. Tom M. Apostol," Introduction to Analytic Number Theory," Narosa Publishing House, New Delhi (Also available from Springer)
4. S. Ramanujan," Collected Papers of Srinivasa Ramanujan," 1927..Available from Amazon
5. George Andrews…………Private communication dealing with identification of HCN testing problem (like primality testing)….long number of years ago
6. Wikipedia page on INTEGER FACTORIZATION.