PRIME AND COMPOSITE NUMBERS PROBLEM IN POLYNOMIAL TIME

PRESENTATION BY:

ALI HUSSIEN

HEBA ABD-ALQADER

UNDER SUPERVISION: DR SAFIA ABBAS

CONTENTS

INTRODUCTION

The composite number problem asks if for a given positive integer $\mathcal{N}$ there exist positive integers m and n such that:

$$\mathcal{N} = m \times n$$

SIMPLE ALGORITHM

Suppose $\mathcal{N}$ is a positive Integer. $\mathcal{N}$ will be a composite number if it has a factor or factors other than 1 and $\mathcal{N}$.

In this algorithm we will divide number $\mathcal{N}$ from numbers 2 to $\mathcal{N} - 1$ one by one. If $\mathcal{N}$ is divisible by any number from 2 to $\mathcal{N} - 1$ then N will be a composite number. Or we can say if we $\mathcal{N}$ has a factor from 2 to $\mathcal{N} - 1$ then $\mathcal{N}$ is a composite number.

SIMPLE ALGORITHM

```
Step 1: Start
Step 2: [ Take Input ] Read: N
Step 3: [ Initialize Counter ] I = 2 and F = 0
Step 4: Repeat While I < N
                Check If N%I == 0 Then
                            Set: F = 1 and Break the Loop
                [ End of If Structure ]
                Compute: I = I + 1
        [ End of While Loop ]
Step 5: Check If F == 1 Then
                Print: N is a composite Number.
        Else
                Print: N is not a Composite Number.
        [ End of If Else Structure ]
Step 6: Exit
```

COMPLEXITY ANALYSIS

The complexity of the composite number problem was unknown for many years, although the problem was known to belong to (Pratt 1975, Garey and Johnson 1983). Agrawal et al. (2004) subsequently and unexpectedly discovered a polynomial-time algorithm now known as the AKS primality test.

AKS PRIMALITY ALGORITHM

The AKS primality is a deterministic primality-proving algorithm created and published by Manindra Agrawal, Neeraj Kayal, and Nitin Saxena, computer scientists at the Indian Institute of Technology Kanpur, on August 6, 2002, in a paper titled "PRIMES is in P".

The algorithm was the first to determine whether any given number is prime or composite within polynomial time. The authors received the 2006 Gödel Prize and the 2006 Fulkerson Prize for this work.

AKS PRIMALITY ALGORITHM

AKS is the first primality-proving algorithm to be simultaneously *general polynomial*, *deterministic*, and *unconditional*. Previous algorithms had been developed for centuries and achieved three of these properties at most, but not all four

- **The AKS algorithm can be used to verify the primality of any general number given. Many fast primality tests are known that work only for numbers with certain properties. For example, the Lucas–Lehmer test works only for Mersenne numbers, while Pépin's test can be applied to Fermat numbers only.**
- **The maximum running time of the algorithm can be expressed as a polynomial over the number of digits in the target number. ECPP and APR conclusively prove or disprove that a given number is prime, but are not known to have polynomial time bounds for all inputs.**
- **The algorithm is guaranteed to distinguish deterministically whether the target number is prime or composite. Randomized tests, such as Miller–Rabin and Baillie–PSW, can test any given number for primality in polynomial time, but are known to produce only a probabilistic result.**
- **The correctness of AKS is not conditional on any subsidiary unproven hypothesis. In contrast, Miller's version of the Miller-Rabin test is fully deterministic and runs in polynomial time over all inputs, but its correctness depends on the truth of the yet-unproven generalized Riemann hypothesis.**

BEFORE PROCEEDING TO THE ALGORITHM

Perfect power

In mathematics, a perfect power is a positive integer that can be expressed as an integer power of another positive integer. More formally, n is a perfect power if there exist natural numbers $m > 1$, and $k > 1$ such that $m^k = n$. In this case, n may be called a perfect kth power. If $k = 2$ or $k = 3$, then n is called a perfect square or perfect cube, respectively. Sometimes 1 is also considered a perfect power ($1^k = 1$ for any k).

BEFORE PROCEEDING TO THE ALGORITHM

Multiplicative order

In number theory, given an integer a and a positive integer n with gcd(a,n) = 1, the multiplicative order of a modulo n is the smallest positive integer k with

$$a^k \equiv 1 \pmod{n}$$

In other words, the multiplicative order of a modulo n is the order of a in the multiplicative group of the units in the ring of the integers modulo n.

The order of a modulo n is usually written ordn(a), or On(a).

BEFORE PROCEEDING TO THE ALGORITHM

Euler's totient function

In number theory, Euler's totient function counts the positive integers up to a given integer n that are relatively prime to n. It is written using the Greek letter phi as $\varphi(n)$ or $\phi(n)$, and may also be called Euler's phi function. It can be defined more formally as the number of integers k in the range $1 \leq k \leq n$ for which the greatest common divisor gcd(n, k) is equal to 1.[2][3] The integers k of this form are sometimes referred to as totatives of n.

For example, the totatives of n = 9 are the six numbers 1, 2, 4, 5, 7 and 8. They are all relatively prime to 9, but the other three numbers in this range, 3, 6, and 9 are not, because gcd(9, 3) = gcd(9, 6) = 3 and gcd(9, 9) = 9. Therefore, $\varphi(9) = 6$. As another example, $\varphi(1) = 1$ since for n = 1 the only integer in the range from 1 to n is 1 itself, and gcd(1, 1) = 1.

Euler's totient function is a multiplicative function, meaning that if two numbers m and n are relatively prime, then $\varphi(mn) = \varphi(m)\varphi(n)$.[4][5] This function gives the order of the multiplicative group of integers modulo n (the group of units of the ring $\mathbb{Z}/n\mathbb{Z}$).[6] It also plays a key role in the definition of the RSA encryption system.

BEFORE PROCEEDING TO THE ALGORITHM

coprime

In number theory, two integers *a* and *b* are said to be relatively prime, mutually prime, or coprime (also spelled co-prime)[1] if the only positive integer that divides both of them is 1. That is, the only common positive factor of the two numbers is 1. This is equivalent to their greatest common divisor being 1

THE ALGORITHM

Input: integer $n > 1$.

1. If ($n = a^b$ for $a \in N$ and $b > 1$), output COMPOSITE.
2. Find the smallest r such that $o_r(n) > \log^2 n$.
3. If $1 < (a, n) < n$ for some $a \le r$, output COMPOSITE.
4. If $n \le r$, output PRIME.
5. For $a = 1$ $\lfloor \sqrt{\phi(r)} \log n \rfloor$ do

   if $((X + a)^n \neq X^n + a \pmod{X^r - 1, n})$, output COMPOSITE;

6. Output PRIME;

REFERENCES

Simple algorithm:

http://www.bscshortnote.com/algorithm-flowchart-for-composite-number/

About composite numbers:

http://mathworld.wolfram.com/CompositeNumberProblem.html

The AKS algorithm: https://en.wikipedia.org/wiki/AKS_primality_test

Perfect power: https://en.wikipedia.org/wiki/Perfect_power

Multiplicative order: https://en.wikipedia.org/wiki/Multiplicative_order

Euler's totient function:

https://en.wikipedia.org/wiki/Euler%27s_totient_function

Co-prime intergers: https://en.wikipedia.org/wiki/Coprime_integers

THANK YOU AND ANY QUESTIONS