Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file
Special pages
Permanent link
Page information
Wikidata item
Cite this page

Print/export
Create a book
Download as PDF
Printable version

Languages
العربية
Català
Deutsch
Español
فارسی
Français
Русский
ไทย
中文
Edit links

Trial division

From Wikipedia, the free encyclopedia

This article is about the mathematical algorithm. For the judicial chamber of the International Criminal Court, see Judges of the International Criminal Court.

Trial division is the most laborious but easiest to understand of the integer factorization algorithms. The essential idea behind trial division tests to see if an integer *n*, the integer to be factored, can be divided by each number in turn that is less than *n*. For example, for the integer $\mathrm{n} = 12$, the only numbers that divide it are 1,2,3,4,6,12. Selecting only the largest powers of primes in this list gives that $12 = 3 \times 4$.

Method [edit]

Given an integer *n* (throughout this article, *n* refers to "the integer to be factored"), trial division consists of systematically testing whether *n* is divisible by any smaller number. Clearly, it is only worthwhile to test candidate factors less than *n*, and in order from two upwards because an arbitrary *n* is more likely to be divisible by two than by three, and so on. With this ordering, there is no point in testing for divisibility by four if the number has already been determined not divisible by two, and so on for three and any multiple of three, etc. Therefore, effort can be reduced by selecting only prime numbers as candidate factors. Furthermore, the trial factors need go no further than $\sqrt{n}$ because, if *n* is divisible by some number *p*, then $n = p \times q$ and if *q* were smaller than *p*, *n* would have earlier been detected as being divisible by *q* or a prime factor of *q*.

A definite bound on the prime factors is possible. Suppose P_i is the *i*'th prime, so that $P_1 = 2$, $P_2 = 3$, $P_3 = 5$, etc. Then the last prime number worth testing as a possible factor of *n* is P_i where $P^2_{i+1} > n$; equality here would mean that P_{i+1} is a factor. Thus, testing with 2, 3, and 5 suffices up to *n* = 48 not just 25 because the square of the next prime is 49, and below *n* = 25 just 2 and 3 are sufficient. Should the square root of *n* be integral, then it is a factor and *n* is a perfect square.

An example of the trial division algorithm, using a prime sieve for prime number generation, is as follows (in Python):

```python
def trial_division(n):
    """Return a list of the prime factors for a natural number."""
    if n < 2:
        return []
    prime_factors = []
    for p in prime_sieve(int(n**0.5) + 1):
        if p*p > n: break
        while n % p == 0:
            prime_factors.append(p)
            n //= p
    if n > 1:
        prime_factors.append(n)
    return prime_factors
```

Trial division is guaranteed to find a factor of *n* if there is one, since it checks all possible prime factors of *n*. Thus, if the algorithm finds one factor, n, it is proof that *n* is a prime. If more than one factor is found, then *n* is a composite integer. A more computationally advantageous way of saying this is, if any prime whose square does not exceed *n* divides it without a remainder, then *n* is not prime.

Speed [edit]

In the worst case, trial division is a laborious algorithm. For a base-2 *n* digit number *a*, if it starts from two and works up to the square root of *a*, the algorithm requires

$$\pi(2^{n/2}) \approx \frac{2^{n/2}}{\left(\frac{n}{2}\right)\ln 2}$$

trial divisions, where $\pi(x)$ denotes the prime-counting function, the number of primes less than *x*. This does not take into account the overhead of primality testing to obtain the prime numbers as candidate factors. A useful table need not be large: P(3512) = 32749, the last prime that fits into a sixteen-bit signed integer and P(6542) = 65521 for unsigned sixteen-bit integers. That would suffice to test primality for numbers up to 65537^2 = 4,295,098,369. Preparing such a table (usually via the Sieve of Eratosthenes) would only be worthwhile if many numbers were to be tested. If instead a variant is used without primality testing, but simply dividing by every odd number less than the square root the base-2 *n* digit number *a*, prime or not, it can take up to about:

$$2^{n/2}$$

In both cases the required time grows exponentially with the digits of the number.

Even so, this is a quite satisfactory method, considering that even the best known algorithms have exponential time growth. For *a* chosen uniformly at random from integers of a given length, there is a 50% chance that 2 is a factor of *a*, and a 33% chance that 3 is a factor of *a*, and so on. It can be shown that 88% of all positive integers have a factor under 100, and that 92% have a factor under 1000. Thus, when confronted by an arbitrary large *a*, it is worthwhile to check for divisibility by the small primes, since for $a = 1000$, in base-2 $n = 10$.

However, many-digit numbers that do not have factors in the small primes can require days or months to factor with trial division. In such cases other methods are used such as the quadratic sieve and the general number field sieve (GNFS). Because these methods also have exponential time growth a practical limit of *n* digits is reached very quickly. For this reason, in public key cryptography, values for *a* are chosen to have large prime factors of similar size so that they cannot be factored by any publicly known method in a useful time period on any available computer system or computer cluster such as supercomputers and computer grids. The largest cryptography-grade number that has been factored is RSA-768, using the GNFS and a network of hundreds of computers. The running time was approximately 2 years.

References [edit]

- Childs, Lindsay N. (2009). *A concrete introduction to higher algebra*. Undergraduate Texts in Mathematics (3rd ed.). New York, NY: Springer-Verlag. ISBN 978-0-387-74527-5. Zbl 1165.00002.
- Crandall, Richard; Pomerance, Carl (2005). *Prime numbers. A computational perspective* (2nd ed.). New York, NY: Springer-Verlag. ISBN 0-387-25282-7. Zbl 1088.11001.

External links [edit]

- Javascript Prime Factor Calculator using trial division. Can handle numbers up to about $9×10^{15}$

v · t · e	**Number-theoretic algorithms** [hide]
Primality tests	AKS test · APR test · Baillie–PSW · ECPP test · Elliptic curve · Pocklington · Fermat · Lucas · *Lucas–Lehmer* · *Lucas–Lehmer–Riesel* · *Proth's theorem* · *Pépin's* · Quadratic Frobenius test · Solovay–Strassen · Miller–Rabin
Prime-generating	Sieve of Atkin · Sieve of Eratosthenes · Sieve of Sundaram · Wheel factorization
Integer factorization	Continued fraction (CFRAC) · Dixon's · Lenstra elliptic curve (ECM) · Euler's · Pollard's rho · *p* − 1 · *p* + 1 · Quadratic sieve (QS) · General number field sieve (GNFS) · *Special number field sieve (SNFS)* · Rational sieve · Fermat's · Shanks' square forms · **Trial division** · Shor's
Multiplication	Ancient Egyptian · Long · Karatsuba · Toom–Cook · Schönhage–Strassen · Fürer's
Discrete logarithm	Baby-step giant-step · Pollard rho · Pollard kangaroo · Pohlig–Hellman · Index calculus · Function field sieve
Greatest common divisor	Binary · Euclidean · Extended Euclidean · Lehmer's
Modular square root	Cipolla · Pocklington's · Tonelli–Shanks
	Chakravala · Cornacchia · Integer relation · Integer square root · Modular exponentiation ·

Other algorithms	Schoof's
Italics indicate that algorithm is for numbers of special forms · Smallcaps indicate a deterministic algorithm	

Categories: Integer factorization algorithms | Division (mathematics)