Part I. Basic Arithmetic

Part II. Primality Testing

Part III. Factoring Integers

Problem. Given a composite integer $n > 1$, find some nontrivial factorization of n, that is $n = uv$ where each of u and v is an integer > 1.

Note: One can be pretty confident about whether a large integer n is composite without knowing a nontrivial factorization.

Expectation. A random number n will have around $\log\log n$ prime factors.

Theorem. *If $\omega(n)$ is the number of distinct prime factors of n, then*

$$\sum_{n\le x}\big(\omega(n)-\log\log x\big)^2 \ll x\log\log x.$$

Corollary. *For almost all n, we have*

$$|\omega(n)-\log\log n| \le (\log\log n)^{2/3}. \qquad (*)$$

$$\sum_{n\le x}\big(\omega(n)-\log\log x\big)^2 \ll x\log\log x$$

Corollary. ***For almost all n, we have***

$$(*)\qquad |\omega(n)-\log\log n| \le (\log\log n)^{2/3}.$$

Explanation of Corollary:

- **Assume there are εx different $n \le x$ for which $(*)$ does not hold, where $\varepsilon > 0$ is fixed and x is large.**
- **All but $\le \sqrt{x}$ of these are $> \sqrt{x}$.**
- **For such n, we have**

$$|\log\log n-\log\log x| < 1$$
$$\implies\quad |\omega(n)-\log\log x| \ge (1/2)(\log\log x)^{2/3}.$$

- **This contradicts the theorem.**

Expectation 2. "Most" numbers n have a prime factor $> \sqrt{n}$.

$$\sum_{p \le x} \frac{1}{p} = \log\log x + A + O(1/\log x)$$

Why does the sum of the reciprocals of the primes diverge and where is this coming from (roughly)?

What does this have to do with Expectation 2?

$$\sum_{n \le x} \sum_{\substack{\sqrt{x} < p \le x \\ p \mid n}} 1$$

$$\log 2 = 0.69314718\ldots$$

Expectation 2. "Most" numbers n have a prime factor $> \sqrt{n}$.

$$\sum_{p \le x} \frac{1}{p} = \log\log x + A + O(1/\log x)$$

Why does the sum of the reciprocals of the primes diverge and where is this coming from (roughly)?

What does this have to do with Expectation 2?

$$\sum_{n \le x} \sum_{\substack{\sqrt{x} < p \le x \\ p \mid n}} 1 = \sum_{\sqrt{x} < p \le x} \sum_{\substack{n \le x \\ p \mid n}} 1 \ge (\log 2)x + O\left(\frac{x}{\log x}\right)$$

$$\log 2 = 0.69314718\ldots$$

Comment: A random number n will have small prime factors, so it is reasonable to first do a quick "sieve" to determine if this is the case.

How many integers $n \leq x$ do not have a prime factor $\leq z$?

On the order of $$\frac{x}{\log z}.$$

Pollard's $p - 1$ Factoring Algorithm

For $k \leq \min\{10^7, n-1\}$, check if $\gcd(2^{k!} - 1 \bmod n, n) > 1$.

Idea: If n has some prime factor p that is not too large and $p-1$ has fairly small prime factors, then probably $p-1$ divides some $k!$ above and hence the gcd. Further, it is likely that all primes dividing n will not simultaneously divide the first occurrence of such a k.

```
> n:=31415926535897932384626433832795028841971693993751:
> check:=0:
  for k from 1 to 25 while check=0 do
    m := 2&^(k!)-1 mod n:
    thegcd := gcd(m,n):
    if thegcd > 1 then lprint(thegcd): check:=1: fi:
  od:
1657
```

```
> n:=31415926535897932384626433832795028841971693993751:
> check:=0:
  for k from 1 to 25 while check=0 do
    m := 2&^(k!)-1 mod n:
    thegcd := gcd(m,n):
    if thegcd > 1 then lprint(thegcd): check:=1: fi:
  od:
1657
```

```
> n:=314159265358979323846264338327950288419716939937 51:
> check:=0:
  for k from 1 to 25 while check=0 do
    m := 2&^(k!)-1 mod n:
    thegcd := gcd(m,n):
    if thegcd > 1 then lprint(thegcd): check:=1: fi:
  od:
1657
> n:=n/1657;
```

$$n := 189595211441749742816091936226886112504355425 43$$

```
> check:=0:
  for k from 1 to 8000 while check=0 do
    m := 2&^(k!)-1 mod n:
    thegcd := gcd(m,n):
    if thegcd > 1 then lprint(thegcd): check:=1: fi:
  od:
2767321
```

```
> n:=3141592653589793238462643383279502884197169399375 1:
```

```
> check:=0:
  for k from 1 to 25 while check=0 do
    m := 2&^(k!)-1 mod n:
    thegcd := gcd(m,n):
    if thegcd > 1 then lprint(thegcd): check:=1: fi:
  od:
1657
```

```
> n:=n/1657;
```

$$n := 1895952114417497428160919362268861125043554 2543$$

```
> check:=0:
  for k from 1 to 8000 while check=0 do
    m := 2&^(k!)-1 mod n:
    thegcd := gcd(m,n):
    if thegcd > 1 then lprint(thegcd): check:=1: fi:
  od:
2767321
```

```
> ifactor(2767320)
```

$$(2)^3\,(3)^2\,(5)\,(7687)$$

Pollard's ρ-Algorithm

This method typically finds a prime factor p of n in about $\sqrt{p}$ steps (so $O(n^{1/4})$ steps), and small prime factors of n will usually be found first.

A couple of relevant asides:

The birthday problem and a card trick.

> $product\left(\frac{k}{365.}, k = (365 - 8) ..365\right)$

$$0.9053761649$$

> $product\left(\frac{k}{365.}, k = (365 - 8)\,..365\right)$

$$0.9053761649$$

> $product\left(\frac{k}{365.}, k = (365 - 22)\,..365\right)$

$$0.4927027640$$

> $product\left(\frac{k}{366.}, k = (366 - 22)\,..366\right)$

$$0.4936769876$$

> $product\left(\frac{k}{365.}, k = (365 - 21)\,..365\right)$

$$0.5243046907$$

Pollard's ρ-Algorithm

This method typically finds a prime factor p of n in about $\sqrt{p}$ steps (so $O(n^{1/4})$ steps), and small prime factors of n will usually be found first.

A couple of relevant asides:

The birthday problem and a card trick.

And what if birthdays are not random?

Pollard's ρ-Algorithm

More Background: Suppose we roll a fair die with "n faces" k times. If $k \geq 2\sqrt{n} + 2$, then with probability $> 1/2$ two of the numbers rolled will be the same.

$$\prod_{j=1}^{k-1} \left(\frac{n-j}{n}\right) \leq \left(1 - \frac{\sqrt{n}}{n}\right)^{\sqrt{n}} \leq \frac{1}{e}$$

Pollard's ρ-Algorithm

Idea with a hiccup:

- Take $f(x) = x^2 + 1$, and define $f^{(1)}(x) = f(x)$ and $f^{(j+1)}(x) = f(f^{(j)}(x))$ for $j \geq 1$.

- Compute $a_j = f^{(j)}(1) \mod n$ for $1 \leq j \leq k$ where $k \approx \sqrt[4]{n}$ (or less).

- Compute $\gcd(a_i - a_j, n)$ for $1 \leq i < j \leq k$ to get a likely factorization of n.

Why does this likely lead to a factorization of n?

What's the hiccup?