CS 491 CAP
Mathematics

Zhengkai Wu

University of Illinois at Urbana-Champaign

Oct 20, 2017

Today

- Number theory
- Combinatorics and Probability

illinois.edu

Number theory

- Primes
  - Sieve of Eratosthenes
- GCD/LCM
  - Euclidean Algorithm and Extension
- Fast exponentiation
- Fermat Little Theorem
  - Miller Rabin
  - Inverse element in modular group
- Chinese Remainder Theorem

Primes

- >1, only two divisors, 1 and itself.
- How to check if an integer is a prime?
- By definition, loop from 2 to N-1 to see if divisible.
- Can be optimized to check 2 to $\lceil \sqrt{n} \rceil$
- Why?
- Complexity: O($\sqrt{n}$)

Sieve of Eratosthenes

- What if we want to generate all primes <= N?
- Brute Force: Run prime check on every integer, O(n*$\sqrt{n}$)
- Let P[1..n] be the array of bools to represent each integer's primality.

Sieve of Eratosthenes

- For each prime number p, we will mark 2p, 3p, … as non-prime.
- Complexity: O(NloglogN)

```
Set P[2 .. N] to initially true
for i = 2 .. sqrt(N):
  if P[i] == true:
   for j = 2 .. N / i:
     P[i * j] = false
```

Linear Sieve

- ◇Let prime[1..k] to store all primes.
- ◇Count = 0;
- ◇For i = 2 to N do
  - If (p[i]) prime[count++] = I;
  - For j = 0 ; j<count && prime[j]*i<=N;++j
    - •P[prime[j]*i] = false;
    - •If (i mod prime[j] == 0) break;

- ◇Why linear?

All composites are eliminated by its minimal prime factor exactly once.

illinois.edu

Greatest Common Divisor (GCD)

- ◇GCD(a,b) is the greatest divisor of both a and b.
- ◇Example:
  - GCD(30,12) = 6; GCD(17,15) = 1

- ◇How to compute?
- ◇GCD(a,b) = GCD(b, a mod b)
- ◇GCD(a,0) = a
- ◇Euclidean Algorithm.
- ◇Complexity: O(log(a+b)^3)

Extended Euclidean Algorithm

- How to solve the equation: ax + by = gcd(a,b) given a,b.
- 6x + 4y = 2 => x=1 and y=-1
- Start from b = 0: gcd(a, 0) = a
- ax = a => x = 1
- Any hint from gcd(a,b) = gcd(b, a mod b) ?

Extended Euclidean Algorithm

- Suppose we can solve bx' + (a mod b)y' = gcd(b, a mod b)
- Since gcd(a,b) = gcd(b, a mod b).
- Also a mod b = $a - \left\lfloor \frac{a}{b} \right\rfloor * b$
- Therefore ay' + b(x' - $\left\lfloor \frac{a}{b} \right\rfloor$y') = gcd(a,b)

- Thus if we have x' and y'. x = y', y = x' - $\left\lfloor \frac{a}{b} \right\rfloor$y'

- Note that this won't be the unique solution. (Why?)
- What if the equation is ax + by = c?

illinois.edu

Extended Euclidean Algorithm

- If gcd(a,b) | c, then there exists a solution. Otherwise, no solution.

- In competitive programming, the most important problem/usage of Number Theory is to solve different Diophantine equation (find integer solution of polynomial equations).
- May also be true in real math. (Fermat's last theorem)

Fast exponentiation

- How to compute a^n quickly if n is large?
- Time complexity: O(log(n)*each multiplication time)
- Can be used in any operator that have associative law. (For example: Matrix exponentiation)

$$a^n = \begin{cases} 1 & n = 0 \\ a & n = 1 \\ \left(a^{n/2}\right)^2 & n \text{ is even} \\ a\left(a^{n/2}\right)^2 & n \text{ is odd} \end{cases}$$

Fermat Little Theorem

- If n is prime, then a^n = a (mod n) holds for all integer a.

- Is the converse true?
- No, counter example:
- Pseudo prime: 341, 2^341 = 2 (mod 341), but 341 is not a prime. (341 = 31*11) It is called a pseudoprime under base 2.
- Strong Pseudo prime: 561, pseudoprime under all bases.

Miller-Rabin

- If the converse is true, we can actually check an integer n by randomly choosing base a and quickly calculate a^n mod n.

- Check wiki for Miller-Rabin algorithm: https://en.wikipedia.org/wiki/Miller%E2%80%93Rabin_primality_test

- Based on Fermat Little Theorem, but changed into a probability algorithm.
- You may want to have this in your template.

Inverse element

- Sometimes we want to calculate the division under modulo.
- What is a/b mod n?

- We call c is the inverse element of b under modulo n if c*b mod n = 1.

- So /b will be equal to *c under the meaning of mod n.

Inverse element

- This means, $c*b = k*n + 1$. => $c*b – k*n = 1$, in which b and n are known and we want to solve for c.
- Use extended euclidean algorithm.
- Have solution iff $\gcd(n,b) = 1$.

- Much easier if n is a prime. (most cases in ICPC)
- Since $a^n = a \pmod n$ => $a^{(n-1)} = 1 \pmod n$
- Thus the inverse of a would be $a^{(n-2)}$.

Chinese Remainder Theorem

◇We have a set of equations:

$$\begin{aligned} x &\equiv a_1 \pmod{n_1} \\ &\vdots \\ x &\equiv a_k \pmod{n_k} \end{aligned},$$

◇How to solve it? (assuming n1,…,nk are coprime)

Chinese Remainder Theorem

- Let N = the product of all ni and Ni = N/ni.
- We want to find Mi such that
- Mi*Ni + mi*ni = 1.
- So that let x = $\sum_{\mathrm{i}=1}^{k} a_i M_i N_i$ will be the solution.

- Why? Think of what does Mi mean here.

Combinatorics

- ◇P(n,r), the number of ways of permutation of r elements out of n elements. (n elements all different)
- ◇P(n,r) = $\frac{n!}{(n-r)!}$
- ◇C(n,r), the number of ways of picking r elements out of n elements. (n elements all different)
- ◇C(n,r) = $\frac{P(n,r)}{r!} = \frac{n!}{r!(n-r)!}$

Combinatorics

- ◇ C(n,r) = C(n-1,r)+C(n-1,r-1)
- ◇ C(n,0) = 1

- ◇ Useful when solving C(n,r) for not large n (typically <= 5k, and module some prime number)

Stars and bars Method

- How many integer solutions in the equation $x1+x2+...+xm = n$? ($xi >= 0$)
- $C(n+m,m-1)$
- What if $xi>=ai$?

How to compute C(n,r) mod p?

- If p is a prime:
- N<=10^6, r<=10^6
- Preprocess the n! and inverse of n!.
- N<=10^10,r<=10^10,p<=1000
- Lucas theorem, write n and r in base p, (n1 n2 .. nk)p and (r1 r2 ... rk)p, then C(n,r)= product of c(ni,ri).

- If p is not a prime:
- N<=5k, r<=5k
- C(n,r) = C(n-1,r)+C(n-1,r-1)
- N<=10^6,r<=10^6, p<=10^5
- Find factors of p, get the count of the factors in n!, (n-r)! and r!.

Probability

- ◇Must be clear what is the base event in the problem.
- ◇Conditional probability: $P(A|B)=P(A \cap B)/P(B)$

Expected value

- ◇E[x] Expectation of random variable x.
- ◇E[x] = $\sum p_i * x_i$ discrete
- ◇ $= \int p(x) * x dx$ continuous

- ◇E[x+y] = E[x] + E[y] for any x and y.
- ◇E[x*y] = E[x] * E[y], for independent x and y.

illinois.edu

POJ 2096

- A system has n bugs and s subsystems. (n,s <= 1000)
- When you are testing the system, you can run exactly one subsystem everyday and you are finding exactly one bug (on that subsystem) everyday. (The bug you find may not be new)
- What's the expected days of finding every bugs at least one time and also finding at least one bug in each subsystem.

- Dp[i][j] = (i/n)*(j/s) * dp[i][j] + (i/n)*(s-j)/s * dp[i][j-1] + (n-i)/n*(j/s) * dp[i-1][j] + (n-i)/n*(s-j)/s * dp[i-1][j-1];

Uva 11762

- Given n, each time randomly pick an prime p that is less than n. If n is divisible by p, then let n/=p, otherwise n remains same.
- What's the expected steps needed to transform n to 1.
- N <= 10^4

- Let k = number of primes less than n.
- f[n] = 1/k * f[a0] + 1/k * f[a1] +…
- ai = n/pi if n divisible by pi
- = n otherwise.