1/7/2012

Websites:

You want to present a professional site.

1. No videos or animations

2. No unprofessional names for variables.

Keep them up to date.

Week 5 or 6: Present your implementation of RSA encryption.

Overview:

1. Number Theory Algorithms: The basis of Cryptography. No one could run a business on the web.

2. Geometric Algorithms: Graphics and Robotics.

3. Combinatorial Algorithms: Computer Science and digital communications.

4. Graph Algorithms: Packet routing in networks, scheduling, network routing

5. Other Algorithms in widespread use: compression, error correcting codes, process simulation, FFT.

Subsets = 2^n

example: S={a,b,c,d} has 16 subsets

S=0, {a}, {b}, {c}, {d}, {a, b}, {a, c},....

Algorithm All Subsets:

S= {0, 1, 2, …., (n-1)}

input: positive integer n

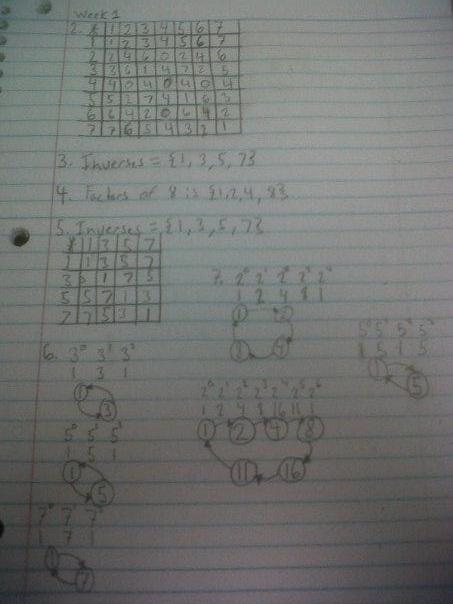
output all subset of {0, 1, 2, …., (n-1)}

method:

for count = 0 to 2^n-1

translate count to binary

print the corresponding elements of the set end.



1/9/2013

Encryption for secure communications

examples: E-Comerce, Banking, etc

Based on Number theory

RSA(Read chapter 31 for RSA): Most important encryption Algorithm

We need the following for RSA:

* Euclid’s Algorithm
* Extended Euclid’s Algorithm: allows us to comput inverses
* Modular Arithmetic: Group of Units mod n, euler phi function
* Modular Exponentiation
* Fermat’s Little theorem
* Generate Large Primes

Arithmetic mod 5(prime number) with prime Zp = {0,1,2,....p-1}

Powers of 2

1/14/2013

Euclid (n,m)

INPUT: 2 Positive Integers

OUTPUT: GCD(n,m)

METHOD:

if m=0

return n;

else return euclid(m,n mod m)

ex: Euclid (5,15)

Euclid(15, 5)

Euclid (15, 5)

return 5;

ex Euclid(20,15)

Euclid(20, 15)

Euclid(15, 5)

Euclid(5,0)

return 5;

ex Euclid(27,41)

Euclid(41, 27)

Euclid(27,14)

Euclid(14, 13)

Euclid (13,1)

Euclid(1,0)

return 1;

2. When you look at the number elements, do you notice any patterns? In Particular, what happens when n is prime? what happens when n is the product of two primes?

The groups of Units are n = ? {1,2,3,4.....n-1} which is the noticeable pattern of numbers in the number element

What happens when you have a product of two prime numbers is you with have a odd number always. like n = 3\*5 = 15 or n = 3\*7 = 21 or n = 1\*5 = 5.

1/16/13

Euler phi function:

Order of an element in the group of units mod n is the power it must be raised to get 1.

How many elements in {1,2,...,n-1}

which are coprime to n. How many elements in

1) They have inverses

2) The only common factor is one.

3) Group of units mod n.

if p is prime

if n = p\*g where p and g are prime

= (p-1)\*(g-1)

Example:

Element Order

1 1

2 4

3 4

4 2

Example:

Element Order

1 1

3 2

5 2

7 2

1/23/13

To understand RSA we need to know:

Modular arithmetic

GCD

The group of units mod n

Euler phi function

how to find the order of elements

how to find inverses

modular exponentiation

Fermat’s little theorem and Eulers Theorem

Generating large primes

Fermat’s Little Theorem:

Let p be prime, and let a {1,2,.....,p-1}

then a^(p-10) is congruent/ 1 mod p

if n is prime, all the numbers {1,2,.....,n-1} pass Fermat’s test

if n is not prime, at least ½ the numbers in {1,2,.....,n-1} fail.

Example: n=10 {1,2,3,4,5,6,7,8,9}

is 2^9 1 mod 10

is 3^9 1 mod 10

is 4^9 4 mod 10(This doesnt pass because it doesnt == 1)

1/28/13

Friday’s Assignment Presentation: Week 2: 4. Print the number of PHI for n = 2 to 1000.

Modular Exponentiation

(300 Digit Numbers)^(300 Digits) modulus (300 digits) //BigInt’s

Example:

(1234567^151) mod 561

import java.math.\*;

public class SlowExp {

public static void main (String[] args) {

BigInteger a = new BigInteger("1234567"); //Base

BigInteger product = BigInteger.ONE;

BigInteger modulus = new BigInteger("561"); //Modulus == mod

int b = 181; //exponent

for (int i = 1; i <= b; i++){

product = product.multiply(a);

product = product.mod(modulus);

System.out.println(a + "^" + i + " = " + product);

}

System.out.println("ans = " + product.mod(modulus);

}

}

Example:

7^181 mod 561 ===> 181 == in binary 10110101

7^181 = 7^10110101

7^1 == 7

7^10 == 7^2 == 49

7^101 == 49^2\*7 mod 561 == 538

7^1011 == 538^2 mod 561 == 337

7^10110 == 337^2 mod 561 == 247

7^101101 == 247^2 \* 7 mod 561 == 142

7^1011010 == 142^2 mod 561 == 529

7^10110101 == 529 ^2 \* 7 mod 561 == 436

1/30/2013

Rabin-Miller primality test

Given a Positive integer n,

1. Pick at random, an integer in {1,2,...., n-1}

2. perform the fermat test. if it passes, the probability that n is not prime is < ½.

3. repeat step 2k times. If it ever fails, n is not prime. If it passes k times in a row the probability that n is not prime is < 1/(2^k).

ex: if k = 40, and it passes 40 times in a row, the probability that n is not prime is < 1/trillion.

2/4/2013

Engma Machine:

Rivest, Shamir, and Adleman(RSA): Named after the ones who invented the algorithm. At least two people knows the key, and you have to keep the key secret.

Public Key Encryption: Publish the key pain (e,n)

n: modulus and n = p \* q 2 big primes

e: exponent where we are going to raising our message. (p-1)(q-1)=(n)

gcd(e, (n)) =1

e is in the group of units mod (n)

Unrealistic Example:

let

p = 43

g = 59

n = 2537

(n) = 2436

e = 13 (Check gcd(13,2436)=1)

Assemble blocks no bigger than the modular (our blocks are of size two letters)

message:

PUBLIC KEY CRYPTOGRAPHY

A=00, B=01,C=02....Z=25

PUBLIC == 15 20 01 11 08 02 (in plain text)

Enciphering: (Block)^13 mod 2537

EX:

(1520)^13 = 95 mod 2537

encrypted(1st 3 blocks) ⇒ 0095 1648 1410 ⇒ Send on the wire, gibberish: A? Q? OK

Deciphering: 1 == exponent \* inverse of the exponent mod (n)

To figure out (n) you’d need to factor n.

2/6/2013

Protocol:

1) Translate symbols into their ascii number. If a symbol has an ascii number with less than 3 digits, pad it in the front with zeros.

Ex. A -> 065

2) Block sizes are 300 digits. 100 symbols per block.

3) Our modulus, n = pq must have exactly 300 digits and bigger than 255255. If the modulus is too small, we destroy the message. If its too big we cant write the encrypted values into blocks of the same soze and therefore dont where the “end” of a number is.

2/08/2013

Plan:

1) Using google docs each presenting group publishes its public key.

2) Every other group will encode a message containing more than 1 block, and send it to the presenting group.

3) Presenting group displays the encrypted message and decodes it using their code.

<http://mathworld.wolfram.com/RSAEncryption.html>

<http://www.java2s.com/Code/Java/Security/SimpleRSApublickeyencryptionalgorithmimplementation.htm>

<http://introcs.cs.princeton.edu/java/78crypto/RSA.java.html>

<http://javadigest.wordpress.com/2012/08/26/rsa-encryption-example/>

2/25/2013

Wed. March 6, 2013: 10% of your grade

RSA: 15% of your grade

3/4/2013

Computational Complexity:

Time Complexity: How many computational steps required by an algorithm’s as a function of input size.

Space complexity: how much memory is required as a function of input size.

Examples Input Size:

1. sort a list of integers: n = # of integers

2. convex hull: n = # of points

3. Traveling Salesman problem: Find the shortest circuit n = # cities

4. Graph algorithms: 2 + vertices and # of edges.

a given algorithm has reads and writes, comparisons, mult, additions,.... We dont need to look at every instruction. we only need to look at “critical” instructions.

ex. for sorting, # comparisons

to figure out time complexity:

1. Identify the critical operations

2. count how many times they are used as a function of input size.

Search: algorithm segmential ⇒ [a0, a1, a2,......an-1] = n

on average we need comparision n/2

we care about the rate of growth of the algorithm

how much longer does it take if we double the input size?

3/18/2013

BST Sort: sort all elements in an array with n elements. O(nlogn)

Quick Sort:

Partition Algorithm: is an array of numbers that have to be two or more. where you have two pointers a greater than(which starts at the right) and a less than pointer( which starts at the left). you have a shift which is used to compare with the other numbers. you have repeat until everything is in order.

3/20/2013

Brute Force:All Permutations:

1) Passwords

2) Travelling Salesman

DF. The rank of a permutation the index in the list of lex ordered permutations.

1) compute the rank of a permutation

2) compute the perm. given a rank.

Psuedo Code:

permutiation rank:

input: a permutation of {1,...,n}

Output: its rank

rank =0

for i = 1 to n-1

count =0

for j = i+1 to n

if (p[j] < p[i]

count = count +1

rank = rank + count \*(n-i)!

next i