Today's Outline

Review

Real-life examples of Fermat pseudoprime test

Miller-Rabin strong-pseudoprime test

Review

• Be able to distinguish *reduction modulo* m from *equality modulo* m.

• **Notation:** $\mathbf{Z}/m = \{\text{integers mod } m\}$

$$\mathbf{Z}/m = \{\overline{0}, \overline{1}, \ldots, \overline{m-1}\}$$

• Fast Modular Exponentiation algorithm.

• Fermat's Little Theorem (special case of Euler's theorem).

• Formula for square roots for prime $p = 3 \bmod 4$ with mandatory checking.

• Easy formula for e^{th} roots mod prime p for e prime, $p \neq 1 \bmod e$. (*Everything* is an e^{th} power modulo such p.)

• Less easy formula for e^{th} roots mod prime p for e prime, $p = 1 \bmod e$, $e^2 \not| (p-1)$, with mandatory checking.

• Note: There *are* algorithms for roots (if exist) modulo *all* primes. See book for general square root algorithm modulo primes.

• Note: Taking roots mod composite $n = pq$ is as hard as factoring n: can use a square root **oracle** mod n to factor n. (Below.)

• Fermat pseudoprime tests. Only moderately good, but very cheap/feasible.

Remark: To make RSA and other PK (*public-key*) things work at all, we need *many* large primes, with at least 100 decimal digits.

And they must be hard to recreate by unauthorized entities, so we must be able to find them *in any range* of integers.

That is, special methods which produce special primes are insufficient for cryptographic purposes, since the set of such things would be too small.

For example, an implementation of RSA which chooses the secret primes p, q effectively from a small set of possibilities *can be broken* by a brute force search of that space, rather than by factoring.

In this context, recall **Kerckhoff's Principle:** *Assume that the mechanism of any cipher or algorithm will become known.*

Hunting for large pseudoprimes

Choose sufficient pseudoprime test for your purposes.

To find large primes, choose a suitable starting point some *odd* N. (Large enough, and probably chosen *randomly*?!)

Test N for suitable pseudoprimality: if it passes you're done, else continue.

Test $N + 2$. If it passes you're done, else continue.

Test $N + 4$. If it passes you're done, else continue.

...

The heuristic coming from the Prime Number Theorem is that we should roughly expect to have to test only about $\frac{1}{2} \ln N$ candidates before finding a prime.

Examples using Fermat pseudoprime test base 2:

$> 10^{20}$ is $10^{20}+39$ (19 tries vs 23 predicted)

$> 10^{30}$ is $10^{30}+57$ (28 tries vs 34 predicted)

$> 10^{40}$ is $10^{40}+121$ (60 tries vs 46 predicted)

$> 10^{50}$ is $10^{50}+151$ (75 tries vs 58 predicted)

$> 10^{60}$ is $10^{60}+7$ (3 tries vs 69 predicted)

$> 10^{70}$ is $10^{70}+33$ (16 tries vs 81 predicted)

$> 10^{80}$ is $10^{80}+129$ (64 tries vs 92 predicted)

$> 10^{90}$ is $10^{90}+289$ (144 tries vs 104)

(The last computation starts to take a noticeable amount of time on a 1.44 G machine in Python.)

It turns out that these are all Fermat pseudoprimes base 3*, as well.*

First Fermat pseudoprimes above larger numbers, and predicted number of tests to find a prime:

$> 10^{100}$ is $10^{100}+267$ (133 tries vs 115)

$> 10^{120}$ is $10^{120}+79$ (39 tries vs 138)

$> 10^{140}$ is $10^{140}+13$ (6 tries vs 161)

$> 10^{160}$ is $10^{160}+303$ (151 tries vs 184)

$> 10^{180}$ is $10^{180}+313$ (156 tries vs 207)

$> 10^{200}$ is $10^{200}+357$ (178 tries vs 230)

$> 10^{300}$ is $10^{300}+331$ (165 tries vs 345)

$> 10^{400}$ is $10^{400}+69$ (34 tries vs 460)

$> 10^{500}$ is $10^{500}+961$ (480 tries vs 576)

$> 10^{1000}$ is $10^{1000}+453$ (226 tries vs 1152)

(30 minutes wait for the last. All these are also Fermat pseudoprimes base 3.)

The 51 Fermat pseudoprimes base 2 just above 10^{50} are $10^{50} + \ldots$:
151, 447, 577, 709, 889, 897, 961, 1059,
1087, 1137, 1249, 1441, 1459, 1521, 1527,
1563, 1611, 1623, 1831, 1899, 2043, 2151,
2239, 2443, 2599, 2691, 2713, 2743, 2781,
2923, 2949, 3021, 3061, 3073, 3139, 3177,
3219, 3417, 3639, 3747, 3889, 4171, 4209,
4227, 4279, 4299, 4453, 4477, 4483, 4917

An *over-interpreted* version of the Prime Number Theorem would suggest the heuristic that near 10^{50} in an interval of length 5000 there should be about

$$\frac{5000}{\ln 10^{50}} = \frac{5000}{50 \cdot \ln 10} \sim \frac{5000}{50 \cdot 2.3} \sim 43$$

primes (we found 51) with gaps

$$\ln 10^{50} \sim 115$$

The 50 Fermat pseudoprimes base 2 just above 10^{100} are $10^{100} + \ldots$:
267, 949, 1243, 1293, 1983, 2773, 2809, 2911, 2967, 3469, 3501, 3799, 4317, 4447, 4491, 5383, 5641, 5949, 6403, 6637, 6903, 7443, 8583, 8653, 9013, 9223, 9259, 9631, 10071, 10557, 10833, 10903, 11143, 11173, 11529, 11667, 11839, 12207, 12817, 13057, 13197, 13369, 13831, 13867, 14287, 15139, 15783, 16183, 16431

An *over-interpreted* version of the Prime Number Theorem would suggest the heuristic that near 10^{100} in an interval of length 16500 there should be about

$$\frac{16500}{\ln 10^{100}} = \frac{16500}{100 \cdot \ln 10} \sim \frac{16500}{100 \cdot 2.3} \sim 72$$

primes (we stopped when we found 50) with gaps

$$\ln 10^{100} \sim 230$$

The 50 Fermat pseudoprimes base 2 just above 10^{200} are $10^{200} + \ldots$:
357, 627, 799, 1849, 2569, 3381, 4143, 4603, 4731, 5263, 5541, 7317, 7357, 7851, 8269, 8383, 8833, 9073, 9277, 11269, 11421, 11619, 12091, 12769, 12897, 13761, 13909, 13981, 14727, 15313, 16407, 16671, 16687, 16699, 16737, 17773, 19069, 21783, 22093, 22711, 22957, 23277, 23317, 23433, 24621, 25329, 25749, 25951, 26737, 27723, 27979

An *over-interpreted* version of the Prime Number Theorem would suggest the heuristic that near 10^{200} in an interval of length 26700 there should be about

$$\frac{26700}{\ln 10^{200}} = \frac{26700}{200 \cdot \ln 10} \sim \frac{26700}{200 \cdot 2.3} \sim 60$$

primes (we stopped when we found 50) with gaps

$$\ln 10^{200} \sim 460$$

Examples of Fermat pseudoprimes base 2 that *fail* base 3:

Taking the first examples above successive powers of 10:

> 100 is 341

> 1000 is 1387

> 10000 is 10261

> 100000 is 113201

> 1000000 is 1004653

> 10000000 is 10004681

> 100000000 is 100302391

> 1000000000 is 1001723911

Examples of Fermat pseudoprimes base 2 and 3 that *fail* base 5:

Taking the first examples above successive powers of 10:

$>$ 100 is 2701

$>$ 1000 is 2701

$>$ 10000 is 18721

$>$ 100000 is 104653

$>$ 1000000 is 1373653

$>$ 10000000 is 10084177

$>$ 100000000 is 100017223

$>$ 1000000000 is 1002261781

Apparently there are composite numbers detected base 5 *but not base* 3*, and vice versa, etc.*

Looking for Carmichael numbers
(2.44 Gig, C++ with GMP)

In 1,000,000,000 $<$ n $<$ 2,000,000,000
1001152801, 1018928485, 1027334881,
1030401901, 1031750401, 1035608041,
1038165961, 1055384929, 1070659201,
1072570801, 1074363265, 1079556193,
1090842145, 1093916341 [**11 hrs**]
100674561, 1103145121 [**13 hrs**] 1125038377
[**15 hrs**] 1131222841, 1132988545,
1134044821, 1136739745, 1138049137 [**16.5
hrs**] 1140441121, 1150270849, 1152793621
[**20 hr**] 1162202581, 1163659861 [**21 hrs**]
1177195201, 1177800481, 1180398961,
1183104001, 1189238401, 1190790721,
1193229577, 1194866101, 1198650961,
1200456577, 1200778753, 1206057601,
1207252621, 1210178305, 1213619761,
1214703721, 1216631521, 1223475841,
1227220801, 1227280681, 1232469001
[**31 hrs**]

Miller-Rabin test, Strong Pseudoprimes

Granting fast modular exponentiation, the following algorithm runs fast. For odd integer n factor

$$n - 1 = 2^s \cdot \ell$$

with ℓ odd. Then n is a **strong pseudoprime base** b if

$$b^\ell = 1 \bmod n$$

or if for some $0 \leq r < s$

$$b^{2^r \cdot \ell} = -1 \bmod n$$

If n fails the test for some b, then n is **definitely composite**.

The heuristic is that if n *passes* base b then the **probability is at least** $3/4$ **that** n **is prime.**

(strong pseudoprime base b implies Fermat pseudoprime base b.)

In fact, there is something *provable* about the Miller-Rabin test:

Theorem: *(Miller-Rabin 1978)* For composite n, at least $3/4$ of b in the range $1 < b < n$ will detect the compositeness (via the Miller-Rabin test)

Pseudo-corollary: If n passes the Miller-Rabin test with k random bases b, then

$$\text{probability}(n \text{ is prime}) \geq 1 - \left(\frac{1}{4}\right)^k$$

Remark: Of course, an integer is either prime or it isn't, so to talk about the probability of its being prime is is misleading at best. On the other hand, operationally this is the viewpoint that is usually taken.

Detailed version of Miller-Rabin base b:

Miller-Rabin test base b:
factor $n-1 = 2^s \cdot m$ with m odd
replace b by $b^m \bmod n$
    if $b = \pm 1 \bmod n$ **stop:** n is 3/4 prime
    else continue
set $r = 1$
while $r < s$
    replace b by $b^2 \bmod n$
    if $b = -1 \bmod n$ **stop:** n is 3/4 prime
    elsif $b = +1 \bmod n$ **stop:** n is composite
    else replace r by $r + 1$ and **continue**
if we fall out of the loop, n is composite.

If n passes this test it is a
strong pseudoprime base b.

By the way, all the Fermat pseudoprimes mentioned earlier are also *strong pseudoprimes base* the 20 prime bases 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 73

... which suggests that the probability that we've reached the wrong conclusion is less than

$$1 - 4^{-20} \sim 0.0000000000001$$

Failure rate of Miller-Rabin?

The fraction of b's which detect compositeness is apparently much greater than 3/4. For $n = 21311$ the detection rate is 0.9976. For 64777 the detection rate is 0.99972. For 1112927 the detection rate is 0.9999973

For $n < 50,000$ there are only 9 non-prime strong pseudoprimes base 2, namely 2047 3277 4033 4681 8321 15841 29341 42799 49141

For $n < 500,000$ there are only 33 non-prime strong pseudoprimes base 2.

For $n < 500,000$ there are *no* non-prime strong pseudoprimes base 2 and 3

For $100,000,000 < n < 101,000,000$ there are 3 strong pseudoprimes base 2 whose compositeness is detected base 3, namely 100463443 100618933 100943201

Some big strong pseudoprimes

On a 2.44 Gig machine, in C++ using GMP: Primality testing Fermat base 2, Miller-Rabin base 2, 3, 5, to find next prime after...

('instantaneous')
First prime after 10^{21} is $10^{21} + 117$

('instantaneous')
First prime after 10^{50} is $10^{50} + 151$

('hint of time taken')
First prime after 10^{100} is $10^{100} + 267$

(3 seconds)
First prime after 10^{200} is $10^{200} + 357$

(8 seconds)
First prime after 10^{300} is $10^{300} + 331$

(97 seconds *(vs 30 minutes for Fermat test in Python)*)
First prime after 10^{1000} is $10^{1000} + 453$

Miller-Rabin test on 25 base 7

Factor $25 - 1 = 2^s \cdot m$ with m odd: here $m = 3$ and $s = 3$. The base is $b = 7$.

(by fast exponentiation) replace b by $b^m \bmod 25 = 7^3 = 18 \bmod 25$.

Since $b = 18 \neq \pm 1 \bmod 25$ we continue, entering the squaring loop. (Set $r = 1$. Since $1 = r < s = 3$ continue.)

Replace $b = 18$ by $b = 18^2 = 24 \bmod 25$.

Since $b = 24 = -1 \bmod 25$ conclude that 25 is a strong pseudoprime base $b = 7$.

Miller-Rabin test on 25 base 2

Factor $25 - 1 = 2^s \cdot m$ with m odd: here $m = 3$ and $s = 3$. The base is $b = 2$.

(by fast exponentiation) replace $b = 2$ by $b^m \bmod 25 = 2^3 = 8 \bmod 25$.

Since $b = 8 \neq \pm 1 \bmod 25$ we continue, entering the squaring loop. (Set $r = 1$. Since $1 = r < s = 3$ continue.)

Replace b by $b^2 = 8^2 = 14 \pm 1 \bmod 25$. Increment r to 2: still $1 = r < s = 3$, so continue.

Replace b by $b^2 = 14^2 = 21 \neq \pm 1 \bmod 25$. Increment r to 3.

Now $r = 3 = s$ so fall out of squaring loop: 25 is *definitely composite*.

Miller-Rabin test on 101 base 2

Factor $101 - 1 = 2^s \cdot m$ with m odd: here $m = 25$ and $s = 2$. The base is $b = 2$.

(By fast exponentiation) replace b by $b^m = 2^{25} = 10 \bmod 101$.

Since $b = 10 \neq \pm 1 \bmod 101$ we continue, entering the squaring loop. (Set $r = 1$. Since $1 = r < s = 2$ continue.)

Replace b by $b^2 = 10^2 = 100 mod 101$.

Since this is $b = -1 \bmod 101$, 101 is a strong pseudoprime base 2.