Factoring integers

Prime Factorization Theorem: For every integer $N \geq 1$, there is a unique bag of prime numbers whose product is N.

Example:

- 75 is the product of the elements in the bag $\{3, 5, 5\}$
- 126 is the product of the elements in the bag $\{2, 3, 3, 7\}$
- 23 is the product of the elements in the bag $\{23\}$

All the elements in a bag must be prime. If N is itself prime, the bag for N is just $\{N\}$.

"The problem of distinguishing prime numbers from composite numbers and of **resolving the latter into their prime factors** is known to be one of the most important and useful in arithmetic. It has engaged the industry and wisdom of ancient and modern geometers to such an extent that it would be superfluous to discuss the problem at length Further, the dignity of the science itself seems to require solution of a problem so elegant and so celebrated."

Carl Friedrich Gauss, Disquisitiones Arithmeticae, 1801

Factoring integers

Prime Factorization Theorem: For every integer $N \geq 1$, there is a unique bag of prime numbers whose product is N.

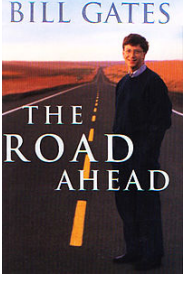
Example:

- 75 is the product of the elements in the bag $\{3, 5, 5\}$
- 126 is the product of the elements in the bag $\{2, 3, 3, 7\}$
- 23 is the product of the elements in the bag $\{23\}$

All the elements in a bag must be prime. If N is itself prime, the bag for N is just $\{N\}$.

"Because both the system's privacy and the security of digital money depend on encryption, a breakthrough in mathematics or computer science that defeats the cryptographic system could be a disaster. The obvious mathematical breakthrough would be the development of an easy way to factor large prime numbers."

(Bill Gates, The Road Ahead*, 1995).*



Secure Sockets Layer

Secure communication with websites uses HTTPS (Secure HTTP)

which is based on SSL (Secure Sockets Layer)

which is based on the RSA (Rivest-Shamir-Adelman) cryptosystem

which depends on the computational difficulty of factoring integers

Factoring integers

Testing whether a number is prime is now well-understood and easy.

Here's a one-line Python script that gives false positives when input is a Carmichael number (rare) and otherwise with probability $\frac{1}{2^{20}}$:

```
def is_prime(p, n=20): return all([pow(randint(1,p-1),p-1,p) == 1
                                   for i in range(n)])}
```

With a few more lines, can get correct answers for Carmichael numbers as well.

The hard part of factoring seems to be this: given an integer N, find any *nontrivial* divisor (divisor other than 1 and N).

If you can do that reliably, you can factor.

Factoring integers the naive way

```
def factor(N):
  for d in range(2, N-1):
    if N % d == 0: return d
```

If d is a divisor of N then so is N/d.

$\min\{d, N/d\} \leq \sqrt{N}$

This shows that it suffices to search among $2, 3, \ldots, \text{int}(\sqrt{N})$

```
def factor(N):
  for d in range(2, intsqrt(N)):
    if N % d == 0: return d
```

where `intsqrt(N)` is a procedure I provide

Useful subroutine: `gcd(m,n)`

`gcd(m,n)` return the greatest common divisor of positive integers m and n. This algorithm is attributed to Euclid, and it is very fast. Here's the code:

```
def gcd(x,y): return x if y == 0 else gcd(y, x % y)
```

Example:

- gcd(12, 16) is 4
- gcd(276534813447635747652, 333070702552660863114) is 18172055646

Using square roots to factor N

Find integers a and b such that

$$a^2 - b^2 = N$$

for then

$$(a - b)(a + b) = N$$

so $a - b$ and $a + b$ are divisors (ideally nontrivial)

How to find such integers? Naive approach...

- Choose integer a slightly more than $\sqrt{N}$
- Check if $\sqrt{a^2 - N}$ is an integer.
- If so, let $b = \sqrt{a^2 - N}$ Success! ☺
  Now $a - b$ is a divisor of N
- If not, repeat with another value for a

Example: $N = 77$

$a = 9$
$\sqrt{a^2 - N} = \sqrt{4} = 2$
so let $b = 2$
$a - b = 7$ is a divisor of N

Example: $N = 23 \cdot 41$
$a = 31 \Rightarrow a^2 - N = 18$ ☹
$a = 32 \Rightarrow a^2 - N = 81$ ☺

For large N, it takes too long to find a good integer a. ☹
We will show how **linear algebra** ☺ helps us synthesize a good integer a.

Using square roots to factor N

Find a and b such that

$$a^2 - b^2 = kN$$

for some integer k. Then

$$(a-b)(a+b) = kN$$

$$\{\text{prime factors of } a-b\} \cup \{\text{prime factors of } a+b\} = \{\text{prime factors of } k\} \cup \{\text{prime factors of } N\}$$

Suppose $\{\text{prime factors of } N\} = \{p, q\}$.

If

- p and q are both factors of $a-b$, or
- p and q are both factors of $a+b$

then $\gcd(a-b, N)$ will not find a nontrivial divisor. However, if

- p is a factor of $a-b$ and q is a factor of $a+b$, or
- p is a factor of $a+b$ and q is a factor of $a-b$

then $\gcd(a-b, N)$ **will** find a nontrivial divisor.

Example:: $N = 7 \cdot 11$
$k = 2 \cdot 3 \cdot 5 \cdot 13$

if $a-b = 2 \cdot 7 \cdot 11$ and
$a+b = 3 \cdot 5 \cdot 13$
then $\gcd(a-b, N) = N$ ☹

if $a-b = 2 \cdot 5 \cdot 11$ and
$a+b = 3 \cdot 7 \cdot 13$

then $\gcd(a-b, N) = 11$ ☺

How to find integers a, b such that $a^2 - b^2 = kN$

Idea: Start by finding the first thousand prime numbers $p_1, \ldots, p_{1000}$.

- Choose a
- Compute $a^2 - N$.
- See if $a^2 - N$ can be factored using only $p_1, \ldots, p_{1000}$
- If not, throw it away.
- If so, record a and the factorization of $a^2 - N$

Repeat a thousand and one times

a	$a * a - N$	factorization
51	182	$2 \cdot 7 \cdot 13$
52	285	$3 \cdot 5 \cdot 19$
53	390	$2 \cdot 3 \cdot 5 \cdot 13$
58	945	$3^3 \cdot 5 \cdot 7$
61	1302	$2 \cdot 3 \cdot \cdot 7 \cdot 13$
62	1425	$3 \cdot 5^2 \cdot 19$
63	1550	$2 \cdot 5^2 \cdot 31$
67	2070	$2 \cdot 3^2 \cdot 5 \cdot 23$
68	2205	$3^2 \cdot 5 \cdot 7^2$
71	2622	$2 \cdot 3 \cdot 19 \cdot 23$

Now we want to find a subset $\{a_1, \ldots, a_k\}$ such that $(a_1^2 - N) \cdots (a_k^2 - N)$ is a perfect square.

Combine $a_1 = 52$, $a_2 = 67$, $a_3 = 71$

$(a_1^2 - N)(a_2^2 - N)(a_3^2 - N) =$
$(3 \cdot 5 \cdot 19)(2 \cdot 3^2 \cdot 5 \cdot 23)(2 \cdot 3 \cdot 19 \cdot 23)$
$= 2^2 \cdot 3^4 \cdot 5^2 \cdot 19^2 \cdot 23^2 = (2 \cdot 3^2 \cdot 5 \cdot 19 \cdot 23)^2$

How to find a subset that works?

Finding a subset that works

Represent each factorization as a vector over $GF(2)$:
Represent $p_1^{a_1} p_2^{a_2} \cdots p_k^{a_k}$ by $\{p_1 : (a_1 \ \% \ 2), p_2 : (a_2 \ \% \ 2) \ldots, p_k : (a_k \ \% \ 2)\}$
Let $A =$ matrix whose rows are these vectors.
A subset of factorizations whose product is a perfect square = a subset of A's rows whose sum is the zero vector
Therefore need to find a nonzero vector in $\{\mathbf{u} \ : \ \mathbf{u} * A = \mathbf{0}\}$
If number of rows $>$ rank of matrix then there exists such a nonzero vector.

a	$a * a - N$	factorization	vector.f
51	182	$2 \cdot 7 \cdot 13$	$\{2 : \text{one}, 13 : \text{one}, 7 : \text{one}\}$
52	285	$3 \cdot 5 \cdot 19$	$\{19 : \text{one}, 3 : \text{one}, 5 : \text{one}\}$
53	390	$2 \cdot 3 \cdot 5 \cdot 13$	$\{2 : \text{one}, 3 : \text{one}, 5 : \text{one}, 13 : \text{one}\}$
58	945	$3^3 \cdot 5 \cdot 7$	$\{3 : \text{one}, 5 : \text{one}, 7 : \text{one}\}$
61	1302	$2 \cdot 3 \cdot \cdot 7 \cdot 13$	$\{31 : \text{one}, 2 : \text{one}, 3 : \text{one}, 7 : \text{one}\}$
62	1425	$3 \cdot 5^2 \cdot 19$	$\{19 : \text{one}, 3 : \text{one}, 5 : 0\}$
63	1550	$2 \cdot 5^2 \cdot 31$	$\{2 : \text{one}, 5 : 0, 31 : \text{one}\}$
67	2070	$2 \cdot 3^2 \cdot 5 \cdot 23$	$\{2 : \text{one}, 3 : 0, 5 : \text{one}, 23 : \text{one}\}$
68	2205	$3^2 \cdot 5 \cdot 7^2$	$\{3 : 0, 5 : \text{one}, 7 : 0\}$
71	2622	$2 \cdot 3 \cdot 19 \cdot 23$	$\{19 : \text{one}, 2 : \text{one}, 3 : \text{one}, 23 : \text{one}\}$

Other uses of Gaussian elimination over $GF(2)$

Simple examples of other uses of Gaussian elimination over $GF(2)$:

- Solving *Lights Out* puzzles.
- Attacking Python's pseudo-random-number generator:

```
>>> import random
>>> random.getrandbits(32)
1984256916
>>> random.getrandbits(32)
4135536776
>>> random.getrandbits(32)
```

What are the next thirty-two bits to be generated? Using Gaussian elimination, you can predict them accurately.

- Breaking simple authentication scheme (playing the role of Eve)....

Improving on the simple authentication scheme

- Password is an n-vector $\hat{\mathbf{x}}$ over $GF(2)$
- **Challenge:** Computer sends random n-vector $\mathbf{a}$
- **Response:** Human sends back $\mathbf{a} \cdot \hat{\mathbf{x}}$.

Repeated until Computer is convinced that Human knowns password $\hat{\mathbf{x}}$.

Eve eavesdrops on communication,
learns m pairs $\mathbf{a}_1, b_1, \ldots, \mathbf{a}_m, b_m$
such that b_i is right response to challenge $\mathbf{a}_i$

The password $\hat{\mathbf{x}}$ is a solution to

$$\underbrace{\begin{bmatrix} \mathbf{a}_1 \\ \hline \vdots \\ \hline \mathbf{a}_m \end{bmatrix}}_{A} \begin{bmatrix} \\ \mathbf{x} \\ \\ \end{bmatrix} = \underbrace{\begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix}}_{\mathbf{b}}$$

Once rank A reaches n, the solution is unique, and Eve can use Gaussian elimination to find it, obtaining the password.

Making the scheme more secure:
The way to make the scheme more secure is to introduce mistakes.

- In about 1/6 of the rounds, randomly, Human sends the *wrong* dot-product.
- Computer is convinced if Human gets the right answers 75% of the time.

Even if Eve knows that Human is making mistakes, she doesn't know which rounds involve mistakes.

Gaussian elimination does not find the solution when some of the right-hand side values b_i are wrong.

In fact, we don't know *any* efficient algorithm Eve can use to find the solution, even if Eve observes many, many rounds.