**583 notes 20090410**

# Quadratic Sieve

**Problem**: Given a composite integer $N > 0$, find a factorisation $N = pq$ for $p, q \neq 1, N$.

Most difficult case is factoring a number $N = pq$ with $p, q$ both large primes.

Quadratic Sieve: useful when $p, q$ are large

Trial factorisation: for all primes $p$ up to some limit $L \leq \sqrt{N}$, check if $p \mid N$.

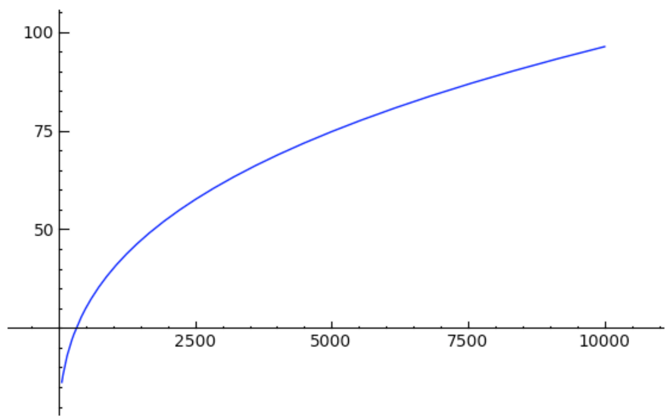
Time complexity: $O(n^{\frac{1}{2}})$.

ECM: Also for finding small factors. Asymptotics depend on size of *smallest* factor $p$:

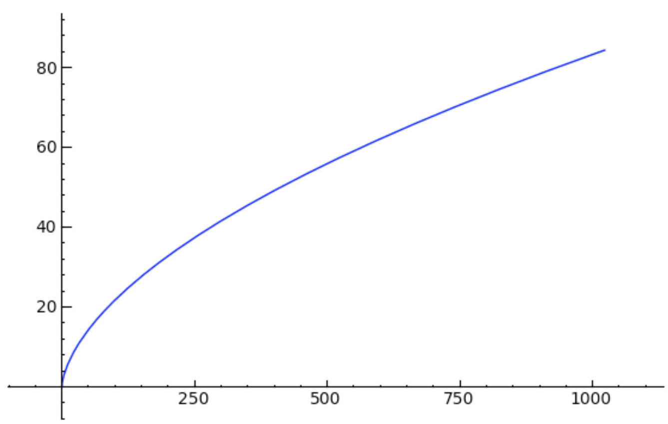
$$O(\exp((1 + o(1))\sqrt{\log(p)\log(\log(p))}))$$

**Subexponential =** always better than exponential, i.e., better than $p^k$ for any $k$, but worse than $f(\log(p))$ for $f$ any polynomial.

```
var('p')
plot((exp(1.01*sqrt(log(p)*log(log(p))))), p, (2,10000))
```




```
var('n')
plot(sqrt(n*log(n)), n, (n,1,1024))
```




If $p$ and $q$ are close together, let $p > q$ and $p, q$ odd, let $p - q = 2b$ and set $a = q + b$.

Then $p = a + b, q = a - b$ and $N = (a + b)(a - b) = a^2 - b^2$.

Fermat's algorithm: Time complexity $O(n^{1/2})$.

Let $D = \lceil\sqrt{N}\rceil, i = 0$ and...

```
>>>
do
  let s=(D+i)^2 - N
  i += 1
while s is not a square
let c = sqrt(s)
return "N = (D+i+c)*(D+i-c)"
```

Example: Factor 943

$D = \lceil\sqrt{943}\rceil = 31$

$i = 0 \quad s = 31^2 - 943 = 18$

$i = 1 \quad s = (31+1)^2 - 943 = 81 = 9^2$.

Thus 943 = (31+1+9)(31+1-9) = 41\times 23.

Lehman extended Fermat's algorithm to case where $p/q$ is close to some $m/n$ where $m, n$ small and *known* as input to the algorithm. This is just a basic extension of the algorithm.

Euler + McKee: Algorithm that is $O(n^{1/3})$ algorithm, which basically involves replacing the quadratic form $x^2 - y^2$ by a full set of reduced binary quadratic forms.

McKee: Found a variant of Fermat's algorithm that is $O(n^{1/4})$.

One-line factoring algorithm in pari that is $O(n^{1/3})$ to factor `f(k)=nextprime(10^k); N=f(k)f(k+s)`

It is:

```
h(x)=;
for(i=1,1000000000,if(issquare(ceil(sqrt(i*x))^2%x),print1("",gcd(x,floor(ceil(sqrt(i*x))-sqrt((ceil(sq
```

It doesn't work for me though...

```
%gp
h(x)=;
for(i=1,1000000000,if(issquare(ceil(sqrt(i*x))^2%x),print1("",gcd(x,floor(ceil(sqrt(i*x))-sqrt((ceil(sqrt(i*x))^2)%:
```

```
%gp
n=nextprime(10^29)*nextprime(10^31);
h(n)
```

```
    *** ceil: precision too low in truncr (precision loss in
    truncation).
```

Shanks SQUFOF (square forms factoring)

uses clas groups of totally real field $O(n^{1/4})$.

Seysen's algorithm uses class groups (asymptically subexponential)

Continued fraction algorithm (asymptically subexponential)

Valle's two-thirds method (Lensta's book)

Difference of squares --> difference of triangle numbers

Dixon's Method (1981):

An observation of Kraitchik (1920s). We only need to find

$$x^2 \equiv y^2 \pmod{N}$$

with $x \not\equiv \pm y \pmod N$ and $(xy, N) = 1$.

**Observation:** If $N$ is odd and divisible by at least two different primes, then the second condition is met at least half of the time. So if you can generate $x, y$ with $x^2 \equiv y^2 \pmod N$ ``randomly", then you'll get a factorization algorithm.

**Observation:** We search for $B$-smooth numbers of the form $f(i) = (\lceil\sqrt{N}\rceil + i)^2 - N$ for some bound $B$. We call such an $f(i)$ a relation. We call all primes $p < B$ the factor base. We call all primes $p < B$ the ***factor base*** i.e. $f(i) = \prod_{j=1}^m b_k^{e_{jk}}$ for $b_k \leq B$. This suggestion was made by Morrison and Brillhart (1975).

Find sufficiently many relations and multiply them to get a square and solve.

Dixon established the asymptotics of this algorithm.

Dixon: Use trial division or ECM to factor $f(i)$.

Pomerance: Better way to factor them all directly with*out* having to factor the $f(i)$. A ***lot of*** people don't correctly note this distinction.

***Example***: Factor 84923. We have $\lceil\sqrt{84923}\rceil = 292$. Let $B = 7$. Takes a long time to find a $B$-smooth $f(i)$. Finally get

$$513^2 mod 84923 = 2^4 \cdot 3 \cdot 5^2 \cdot 7$$

$$537^2 mod 84923 = 2^6 \cdot 3 \cdot 5^2 \cdot 7$$

$f(513 - 292)$ ---> [0,1,0,1] = r_1

$f(537 - 292)$ --> [0,1,0,1] = r_2

Linear algebra problem: r_1 + r_2 = [0,0,0,0] over $\mathbf{F}_2$.

Thus

$$513^2 \cdot 537^2 = 2^{10} \cdot 3^2 \cdot 5^4 \cdot 7^2 \pmod N$$

Reducing modulo $N$ on both sides, we get

$$20712^2 = 16800^2 \pmod N.$$

```
N = 84923
37512*3912  % N
```

```
0
```

```
gcd(37512,N)
```

```
521
```

```
gcd(3912,N)
```

```
163
```

```
N/(163*521)
```

```
1
```

```
20712-16800
```

```
3912
```

```
20712+16800
```

```
37512
```

**Complexity:**

$$L_n(1/2, s\sqrt{2}) = O(\exp(2\sqrt{2}\sqrt{\log(N)\log(\log(N))})$$

Quadratic Sieve: $L_n(1/2, 1)$.