Integer factorization: a progress report

D. J. Bernstein

Thanks to:
University of Illinois at Chicago
NSF DMS–0140542
Alfred P. Sloan Foundation

Exercise for the reader:
Find a nontrivial factor of
6366223796340423057152171586.

on:

is at Chicago

42

oundation

Exercise for the reader:

Find a nontrivial factor of

636622379634042305715217158 6.

Exercise for the re

Find a nontrivial f

63662237963404 2

Small prime factor

are easy to find.

Larger primes are

"Elliptic-curve me

scales surprisingly

(1987 Lenstra)

ECM has found a

(2005 Dodson; ra

$\approx 3 \cdot 10^{12}$ Optero

www.loria.fr/~zimmerm

Exercise for the reader:
Find a nontrivial factor of
6366223796340423057152171586.

Exercise for the reader:
Find a nontrivial factor of
6366223796340423057152171586.

Small prime factors
are easy to find.

Larger primes are harder.

"Elliptic-curve method" (ECM)
scales surprisingly well.
(1987 Lenstra)

ECM has found a prime $\approx 2^{219}$.
(2005 Dodson; rather lucky;
$\approx 3 \cdot 10^{12}$ Opteron cycles)

`www.loria.fr/~zimmerma/records/p66`

ader:

actor of

3057152171586.

Exercise for the reader:

Find a nontrivial factor of

6366223796340423057152171586.

Small prime factors
are easy to find.

Larger primes are harder.

"Elliptic-curve method" (ECM)
scales surprisingly well.
(1987 Lenstra)

ECM has found a prime $\approx 2^{219}$.
(2005 Dodson; rather lucky;
$\approx 3 \cdot 10^{12}$ Opteron cycles)

www.loria.fr/~zimmerma/records/p66

For worst-case inte
two very large prin
ECM does not sca
"number-field siev
(1988 Pollard, et a

Latest record: NF
two prime factors
of "RSA-200" cha
Bahr Boehm Fran
$\approx 5 \cdot 10^{18}$ Opteron

How much more d
is it to find prime
of an integer $n \approx$

www.loria.fr/~zimmerm

Exercise for the reader:
Find a nontrivial factor of
6366223796340423057152171586.

Small prime factors
are easy to find.

Larger primes are harder.

"Elliptic-curve method" (ECM)
scales surprisingly well.
(1987 Lenstra)

ECM has found a prime $\approx 2^{219}$.
(2005 Dodson; rather lucky;
$\approx 3 \cdot 10^{12}$ Opteron cycles)

`www.loria.fr/~zimmerma/records/p66`

For worst-case integers with
two very large prime factors,
ECM does not scale as well as
"number-field sieve" (NFS).
(1988 Pollard, et al.)

Latest record: NFS has found
two prime factors $\approx 2^{332}$
of "RSA-200" challenge. (2005
Bahr Boehm Franke Kleinjung;
$\approx 5 \cdot 10^{18}$ Opteron cycles)

How much more difficult
is it to find prime factors $\approx 2^{512}$
of an integer $n \approx 2^{1024}$?

`www.loria.fr/~zimmerma/records/rsa200`

ader:
actor of
3057152171586.

rs

harder.

thod" (ECM)
well.

prime $\approx 2^{219}$.
ther lucky;
n cycles)

na/records/p66

For worst-case integers with two very large prime factors, ECM does not scale as well as "number-field sieve" (NFS). (1988 Pollard, et al.)

Latest record: NFS has found two prime factors $\approx 2^{332}$ of "RSA-200" challenge. (2005 Bahr Boehm Franke Kleinjung; $\approx 5 \cdot 10^{18}$ Opteron cycles)

How much more difficult is it to find prime factors $\approx 2^{512}$ of an integer $n \approx 2^{1024}$?

www.loria.fr/~zimmerma/records/rsa200

NFS step 1: find a

NFS tries to facto
inspecting values o

Select integer $m \in$
find integers f_5, f_4
with $n = f_5 m^5 +$
for various integers
$(a - bm)(f_5 a^5 + f$

Practically every c
will succeed in fac
Better speed from
$(a - bm)(f_5 a^5 + f$

For worst-case integers with two very large prime factors, ECM does not scale as well as "number-field sieve" (NFS). (1988 Pollard, et al.)

Latest record: NFS has found two prime factors $\approx 2^{332}$ of "RSA-200" challenge. (2005 Bahr Boehm Franke Kleinjung; $\approx 5 \cdot 10^{18}$ Opteron cycles)

How much more difficult is it to find prime factors $\approx 2^{512}$ of an integer $n \approx 2^{1024}$?

`www.loria.fr/~zimmerma/records/rsa200`

NFS step 1: find attractive m's

NFS tries to factor n by inspecting values of a polynomial.

Select integer $m \in [n^{1/6}, n^{1/5}]$; find integers $f_5, f_4, \ldots, f_0$ with $n = f_5 m^5 + f_4 m^4 + \cdots + f_0$; for various integers a, b inspect $(a - bm)(f_5 a^5 + f_4 a^4 b + \cdots + f_0 b^5)$.

Practically every choice of m will succeed in factoring n. Better speed from smaller values $(a - bm)(f_5 a^5 + f_4 a^4 b + \cdots + f_0 b^5)$.

egers with
me factors,
le as well as
e" (NFS).
al.)

S has found
$\approx 2^{332}$
llenge. (2005
ke Kleinjung;
n cycles)

lifficult
factors $\approx 2^{512}$
2^{1024}?

na/records/rsa200

NFS step 1: find attractive m's

NFS tries to factor n by inspecting values of a polynomial.

Select integer $m \in [n^{1/6}, n^{1/5}]$;
find integers $f_5, f_4, \ldots, f_0$
with $n = f_5 m^5 + f_4 m^4 + \cdots + f_0$;
for various integers a, b inspect
$(a - bm)(f_5 a^5 + f_4 a^4 b + \cdots + f_0 b^5)$.

Practically every choice of m will succeed in factoring n.
Better speed from smaller values
$(a - bm)(f_5 a^5 + f_4 a^4 b + \cdots + f_0 b^5)$.

e.g. $n = 3141592$
Can choose $m = 1$
$f_5 = 314$, $f_4 = 15$
$f_2 = 358$, $f_1 = 97$

NFS succeeds in f
by inspecting valu
$(a - 1000b)(314a^5$
for various integer

But NFS succeeds
using $m = 1370$, i
$(a - 1370b)(65a^5$
$38a^3b^2 + 377a^2b^3$

NFS step 1: find attractive m's

NFS tries to factor n by inspecting values of a polynomial.

Select integer $m \in [n^{1/6}, n^{1/5}]$; find integers $f_5, f_4, \ldots, f_0$ with $n = f_5 m^5 + f_4 m^4 + \cdots + f_0$; for various integers a, b inspect $(a - bm)(f_5 a^5 + f_4 a^4 b + \cdots + f_0 b^5)$.

Practically every choice of m will succeed in factoring n. Better speed from smaller values $(a - bm)(f_5 a^5 + f_4 a^4 b + \cdots + f_0 b^5)$.

e.g. $n = 314159265358979323$: Can choose $m = 1000$, $f_5 = 314$, $f_4 = 159$, $f_3 = 265$, $f_2 = 358$, $f_1 = 979$, $f_0 = 323$.

NFS succeeds in factoring n by inspecting values $(a - 1000b)(314a^5 + \cdots + 323b^5)$ for various integer pairs (a, b).

But NFS succeeds more quickly using $m = 1370$, inspecting $(a - 1370b)(65a^5 + 130a^4 b + 38a^3 b^2 + 377a^2 b^3 + 127ab^4 + 33b^5)$.

attractive m's

r n by

of a polynomial.

$\in [n^{1/6}, n^{1/5}]$;

$_4, \ldots, f_0$

$f_4 m^4 + \cdots + f_0$;

s a, b inspect

$_4 a^4 b + \cdots + f_0 b^5)$.

hoice of m

toring n.

smaller values

$_4 a^4 b + \cdots + f_0 b^5)$.

e.g. $n = 314159265358979323$:
Can choose $m = 1000$,
$f_5 = 314$, $f_4 = 159$, $f_3 = 265$,
$f_2 = 358$, $f_1 = 979$, $f_0 = 323$.

NFS succeeds in factoring n
by inspecting values
$(a - 1000b)(314a^5 + \cdots + 323b^5)$
for various integer pairs (a, b).

But NFS succeeds more quickly
using $m = 1370$, inspecting
$(a - 1370b)(65a^5 + 130a^4 b +$
$38a^3 b^2 + 377a^2 b^3 + 127ab^4 + 33b^5)$.

NFS step 1: Consi
2^{45} possible choice
Quickly identify, e.
2^{25} attractive canc

Will choose one m

If $|a| \le SR$ and $|b$
$\big|(a - bm)(f_5 a^5 +$
$\mu(m, S) R^6$ where
$(mS^{-1} + S)(\big| f_5 S^5$

Attractive m, S: s
(1999 Murphy)

e.g. $n = 314159265358979323$:
Can choose $m = 1000$,
$f_5 = 314$, $f_4 = 159$, $f_3 = 265$,
$f_2 = 358$, $f_1 = 979$, $f_0 = 323$.

NFS succeeds in factoring n
by inspecting values
$(a - 1000b)(314a^5 + \cdots + 323b^5)$
for various integer pairs (a, b).

But NFS succeeds more quickly
using $m = 1370$, inspecting
$(a - 1370b)(65a^5 + 130a^4b +$
$38a^3b^2 + 377a^2b^3 + 127ab^4 + 33b^5)$.

NFS step 1: Consider, e.g.,
2^{45} possible choices of m.
Quickly identify, e.g.,
2^{25} attractive candidates.

Will choose one m in step 2.

If $|a| \le SR$ and $|b| \le S^{-1}R$ then
$\left|(a - bm)(f_5a^5 + \cdots + f_0b^5)\right| \le$
$\mu(m, S)R^6$ where $\mu(m, S) =$
$(mS^{-1} + S)(\left|f_5S^5\right| + \cdots + \left|f_0S^{-5}\right|)$.

Attractive m, S: small $\mu(m, S)$.
(1999 Murphy)

65358979323:
1000,
9, $f_3 = 265$,
79, $f_0 = 323$.

actoring n
es
$^5 + \cdots + 323b^5$)
pairs (a, b).

more quickly
nspecting
$+ 130a^4b +$
$+ 127ab^4 + 33b^5$).

NFS step 1: Consider, e.g.,
2^{45} possible choices of m.
Quickly identify, e.g.,
2^{25} attractive candidates.

Will choose one m in step 2.

If $|a| \le SR$ and $|b| \le S^{-1}R$ then
$\left|(a - bm)(f_5a^5 + \cdots + f_0b^5)\right| \le$
$\mu(m, S)R^6$ where $\mu(m, S) =$
$(mS^{-1} + S)(\left|f_5S^5\right| + \cdots + \left|f_0S^{-5}\right|)$.

Attractive m, S: small $\mu(m, S)$.
(1999 Murphy)

Choosing one typi
produces $\mu(m, 1)$

Question: How mu
need to save facto
m, S with $\mu(m, S$

This has as much
chopping $\approx 3 \lg B$

Searching for goo
takes noticeable fr
total time of optin
(If not, consider m
End up with rathe

NFS step 1: Consider, e.g.,
2^{45} possible choices of m.
Quickly identify, e.g.,
2^{25} attractive candidates.

Will choose one m in step 2.

If $|a| \le SR$ and $|b| \le S^{-1}R$ then
$\left|(a - bm)(f_5 a^5 + \cdots + f_0 b^5)\right| \le$
$\mu(m, S)R^6$ where $\mu(m, S) =$
$(mS^{-1} + S)(\left|f_5 S^5\right| + \cdots + \left|f_0 S^{-5}\right|)$.

Attractive m, S: small $\mu(m, S)$.
(1999 Murphy)

Choosing one typical $m \approx n^{1/6}$
produces $\mu(m, 1) \approx n^{2/6}$.

Question: How much time do we
need to save factor of B—to find
m, S with $\mu(m, S) \approx B^{-1} n^{2/6}$?

This has as much impact as
chopping $\approx 3 \lg B$ bits out of n.

Searching for good values of m
takes noticeable fraction of
total time of optimized NFS.
(If not, consider more m's!)
End up with rather large B.

ider, e.g.,
es of m.
.g.,
didates.

n in step 2.

$| \le S^{-1}R$ then
$\cdots + f_0 b^5)| \le$
$\mu(m, S) =$
$^5| + \cdots + |f_0 S^{-5}|)$.

small $\mu(m, S)$.

Choosing one typical $m \approx n^{1/6}$ produces $\mu(m, 1) \approx n^{2/6}$.

Question: How much time do we need to save factor of B—to find m, S with $\mu(m, S) \approx B^{-1} n^{2/6}$?

This has as much impact as chopping $\approx 3 \lg B$ bits out of n.

Searching for good values of m takes noticeable fraction of total time of optimized NFS. (If not, consider more m's!) End up with rather large B.

Four answers:

Time $B^{7.5+o(1)}$ to
$m \approx B^{0.25} n^{1/6}$
with $\mu(m, 1) \approx B$
by searching conse

Time $B^{6+o(1)}$ by s
through m's with

Time $B^{4.5+o(1)}$ to
with $\mu(m, B^{0.75})$
(1999 Murphy)

Time $B^{3.5+o(1)}$ by
controlling f_3. (20

`cr.yp.to/talks.html#2`

Choosing one typical $m \approx n^{1/6}$ produces $\mu(m, 1) \approx n^{2/6}$.

Question: How much time do we need to save factor of B—to find m, S with $\mu(m, S) \approx B^{-1}n^{2/6}$?

This has as much impact as chopping $\approx 3 \lg B$ bits out of n.

Searching for good values of m takes noticeable fraction of total time of optimized NFS. (If not, consider more m's!) End up with rather large B.

Four answers:

Time $B^{7.5+o(1)}$ to find $m \approx B^{0.25}n^{1/6}$ with $\mu(m, 1) \approx B^{-1}n^{2/6}$ by searching consecutive m's.

Time $B^{6+o(1)}$ by skipping through m's with small f_5, f_4.

Time $B^{4.5+o(1)}$ to find $m \approx B^1 n^{1/6}$ with $\mu(m, B^{0.75}) \approx B^{-1}n^{2/6}$. (1999 Murphy)

Time $B^{3.5+o(1)}$ by partly controlling f_3. (2004 Bernstein)

cal $m \approx n^{1/6}$
$\approx n^{2/6}$.

uch time do we
r of B—to find
) $\approx B^{-1}n^{2/6}$?

impact as
bits out of n.

d values of m
action of
nized NFS.
nore m's!)
r large B.

Four answers:

Time $B^{7.5+o(1)}$ to find
$m \approx B^{0.25}n^{1/6}$
with $\mu(m,1) \approx B^{-1}n^{2/6}$
by searching consecutive m's.

Time $B^{6+o(1)}$ by skipping
through m's with small f_5, f_4.

Time $B^{4.5+o(1)}$ to find $m \approx B^1 n^{1/6}$
with $\mu(m, B^{0.75}) \approx B^{-1}n^{2/6}$.
(1999 Murphy)

Time $B^{3.5+o(1)}$ by partly
controlling f_3. (2004 Bernstein)

cr.yp.to/talks.html#2004.11.15

New method uses
lattice-basis reduct
specifically integer

Many lower-level s
effectively choppin
a few more bits ou
approximate reduc
(e.g., 2004 Schnor
"PSLQ" (1999 Ba
"geometric" ideas
(2004 Nguyen Ste

www.loria.fr/~stehle/
www.loria.fr/~stehle/

Four answers:

Time $B^{7.5+o(1)}$ to find
$m \approx B^{0.25} n^{1/6}$
with $\mu(m, 1) \approx B^{-1} n^{2/6}$
by searching consecutive m's.

Time $B^{6+o(1)}$ by skipping
through m's with small f_5, f_4.

Time $B^{4.5+o(1)}$ to find $m \approx B^1 n^{1/6}$
with $\mu(m, B^{0.75}) \approx B^{-1} n^{2/6}$.
(1999 Murphy)

Time $B^{3.5+o(1)}$ by partly
controlling f_3. (2004 Bernstein)

cr.yp.to/talks.html#2004.11.15

New method uses 4-dimensional
lattice-basis reduction,
specifically integer-relation finding.

Many lower-level speedups,
effectively chopping
a few more bits out of n:
approximate reduction
(e.g., 2004 Schnorr),
"PSLQ" (1999 Bailey Ferguson),
"geometric" ideas
(2004 Nguyen Stehlé).

www.loria.fr/~stehle/LOWDIM.html
www.loria.fr/~stehle/FPLLL.html

find

$^{-1}n^{2/6}$

ecutive m's.

skipping

small f_5, f_4.

find $m \approx B^1 n^{1/6}$

$\approx B^{-1} n^{2/6}$.

partly

004 Bernstein)

2004.11.15

New method uses 4-dimensional
lattice-basis reduction,
specifically integer-relation finding.

Many lower-level speedups,
effectively chopping
a few more bits out of n:
approximate reduction
(e.g., 2004 Schnorr),
"PSLQ" (1999 Bailey Ferguson),
"geometric" ideas
(2004 Nguyen Stehlé).

`www.loria.fr/~stehle/LOWDIM.html`
`www.loria.fr/~stehle/FPLLL.html`

NFS step 2: choos

Previous step insp
Kept the attractive
as measured by μ

NFS step 2: Evalu
of each attractive
Choose highest-me
for factoring n.

Merit evaluation is
but is applied to fe
More accurate tha
so selects better n

New method uses 4-dimensional lattice-basis reduction, specifically integer-relation finding.

Many lower-level speedups, effectively chopping a few more bits out of n: approximate reduction (e.g., 2004 Schnorr), "PSLQ" (1999 Bailey Ferguson), "geometric" ideas (2004 Nguyen Stehlé).

`www.loria.fr/~stehle/LOWDIM.html`
`www.loria.fr/~stehle/FPLLL.html`

NFS step 2: choose one m

Previous step inspected many m's. Kept the attractive m's, as measured by μ values.

NFS step 2: Evaluate merit of each attractive m. Choose highest-merit m for factoring n.

Merit evaluation is slower than μ but is applied to fewer m's. More accurate than μ so selects better m.

4-dimensional
tion,
-relation finding.

speedups,
g
ut of n:
tion
r),
iley Ferguson),

hlé).

/LOWDIM.html
/FPLLL.html

NFS step 2: choose one m

Previous step inspected many m's.
Kept the attractive m's,
as measured by μ values.

NFS step 2: Evaluate merit
of each attractive m.
Choose highest-merit m
for factoring n.

Merit evaluation is slower than μ
but is applied to fewer m's.
More accurate than μ
so selects better m.

Given $H, m, f_5, \ldots$
Consider integer p
with $b > 0$ and gc
How many values
$(a - bm)(f_5 a^5 + \cdot$
are in $[-H, H]$?

μ bound is quite c

Instead enumerate
count a's for each
(Silverman, Contir

NFS step 2: choose one m

Previous step inspected many m's.
Kept the attractive m's,
as measured by μ values.

NFS step 2: Evaluate merit
of each attractive m.
Choose highest-merit m
for factoring n.

Merit evaluation is slower than μ
but is applied to fewer m's.
More accurate than μ
so selects better m.

Given $H, m, f_5, \ldots, f_0$:
Consider integer pairs (a, b)
with $b > 0$ and $\gcd\{a, b\} = 1$.
How many values
$(a - bm)(f_5 a^5 + \cdots + f_0 b^5)$
are in $[-H, H]$?

μ bound is quite crude.

Instead enumerate b's,
count a's for each b.
(Silverman, Contini, Lenstra)

se one m

ected many m's.
e m's,
values.

uate merit
m.
erit m

s slower than μ
ewer m's.
n μ
n.

Given $H, m, f_5, \ldots, f_0$:
Consider integer pairs (a, b)
with $b > 0$ and $\gcd\{a, b\} = 1$.
How many values
$(a - bm)(f_5 a^5 + \cdots + f_0 b^5)$
are in $[-H, H]$?

μ bound is quite crude.

Instead enumerate b's,
count a's for each b.
(Silverman, Contini, Lenstra)

Faster (2004 Bern
Numerically appro
the area of
$\{(a, b) \in \mathbf{R} \times \mathbf{R} : \cdot$

Number of qualify
is extremely close
$(3/\pi^2) H^{2/6} \int_{-\infty}^{\infty} d$
where
$f(x) = (x - m)(f$

Evaluate superelli
by standard techni
partition, use serie

Given $H, m, f_5, \ldots, f_0$:
Consider integer pairs (a, b)
with $b > 0$ and $\gcd\{a, b\} = 1$.
How many values
$(a - bm)(f_5 a^5 + \cdots + f_0 b^5)$
are in $[-H, H]$?

μ bound is quite crude.

Instead enumerate b's,
count a's for each b.
(Silverman, Contini, Lenstra)

Faster (2004 Bernstein):
Numerically approximate
the area of
$\{(a, b) \in \mathbf{R} \times \mathbf{R} : \cdots \in [-H, H]\}$.

Number of qualifying pairs
is extremely close to
$(3/\pi^2) H^{2/6} \int_{-\infty}^{\infty} dx/(f(x)^2)^{1/6}$
where
$f(x) = (x - m)(f_5 x^5 + \cdots + f_0)$.

Evaluate superelliptic integral
by standard techniques:
partition, use series expansions.

. . , f_0:
airs (a, b)
d$\{a, b\} = 1$.

$\cdots + f_0 b^5)$

crude.

b's,
b.
ni, Lenstra)

Faster (2004 Bernstein):
Numerically approximate
the area of
$\{(a, b) \in \mathbf{R} \times \mathbf{R} : \cdots \in [-H, H]\}$.

Number of qualifying pairs
is extremely close to
$(3/\pi^2) H^{2/6} \int_{-\infty}^{\infty} dx/(f(x)^2)^{1/6}$
where
$f(x) = (x - m)(f_5 x^5 + \cdots + f_0)$.

Evaluate superelliptic integral
by standard techniques:
partition, use series expansions.

`cr.yp.to/talks.html#2004.11.15`

Will see that NFS
fully factored valu
$(a - bm)(f_5 a^5 + \cdot$

Won't be able to u
with unknown prin

Merit of $m \approx$ cha
$(a - bm)(f_5 a^5 + \cdot$
will be fully factor

Simplified definitio
“fully factored”: ‘
i.e., no prime divis

Faster (2004 Bernstein):
Numerically approximate
the area of
$\{(a, b) \in \mathbf{R} \times \mathbf{R} : \cdots \in [-H, H]\}$.

Number of qualifying pairs
is extremely close to
$(3/\pi^2)H^{2/6} \int_{-\infty}^{\infty} dx/(f(x)^2)^{1/6}$
where
$f(x) = (x - m)(f_5 x^5 + \cdots + f_0)$.

Evaluate superelliptic integral
by standard techniques:
partition, use series expansions.

cr.yp.to/talks.html#2004.11.15

Will see that NFS needs
fully factored values
$(a - bm)(f_5 a^5 + \cdots + f_0 b^5)$.

Won't be able to use values
with unknown prime divisors.

Merit of $m \approx$ chance that
$(a - bm)(f_5 a^5 + \cdots + f_0 b^5)$
will be fully factored.

Simplified definition of
"fully factored": "2^{40}-smooth,"
i.e., no prime divisors $> 2^{40}$.

stein):
ximate

$\cdots \in [-H, H]\}$.

ing pairs
to
$dx/(f(x)^2)^{1/6}$

$f_5 x^5 + \cdots + f_0)$.

otic integral
iques:
es expansions.

2004.11.15

Will see that NFS needs
fully factored values
$(a - bm)(f_5 a^5 + \cdots + f_0 b^5)$.

Won't be able to use values
with unknown prime divisors.

Merit of $m \approx$ chance that
$(a - bm)(f_5 a^5 + \cdots + f_0 b^5)$
will be fully factored.

Simplified definition of
"fully factored": "2^{40}-smooth,"
i.e., no prime divisors $> 2^{40}$.

What is chance th
$(a - bm)(f_5 a^5 + \cdots$
will be fully factor
given that it is in

Try to account for
roots modulo sma
(Schroeppel, Murp

Can do this accura
(2002 Bernstein)

cr.yp.to/papers.html#
cr.yp.to/psibound.htm

Will see that NFS needs
fully factored values
$(a - bm)(f_5 a^5 + \cdots + f_0 b^5)$.

Won't be able to use values
with unknown prime divisors.

Merit of $m \approx$ chance that
$(a - bm)(f_5 a^5 + \cdots + f_0 b^5)$
will be fully factored.

Simplified definition of
"fully factored": "2^{40}-smooth,"
i.e., no prime divisors $> 2^{40}$.

What is chance that
$(a - bm)(f_5 a^5 + \cdots + f_0 b^5)$
will be fully factored,
given that it is in $[-H, H]$?

Try to account for
roots modulo small primes.
(Schroeppel, Murphy, et al.)

Can do this accurately.
(2002 Bernstein)

`cr.yp.to/papers.html#psi`
`cr.yp.to/psibound.html`

needs
es
$\cdots + f_0b^5)$.

use values
me divisors.

nce that
$\cdots + f_0b^5)$
ed.

on of
“2^{40}-smooth,”
sors $> 2^{40}$.

What is chance that
$(a - bm)(f_5a^5 + \cdots + f_0b^5)$
will be fully factored,
given that it is in $[-H, H]$?

Try to account for
roots modulo small primes.
(Schroeppel, Murphy, et al.)

Can do this accurately.
(2002 Bernstein)

`cr.yp.to/papers.html#psi`
`cr.yp.to/psibound.html`

NFS step 3: find s

Have integer m, p
$f(x) = (x - m)(f$

Consider values b^6
$(a - bm)(f_5a^5 + \cdot$

NFS step 3: Choo
For each pair $(a, b$
with $b^6 f(a/b) \in [$
find small prime di
of $b^6 f(a/b)$.

What is chance that
$(a - bm)(f_5 a^5 + \cdots + f_0 b^5)$
will be fully factored,
given that it is in $[-H, H]$?

Try to account for
roots modulo small primes.
(Schroeppel, Murphy, et al.)

Can do this accurately.
(2002 Bernstein)

`cr.yp.to/papers.html#psi`
`cr.yp.to/psibound.html`

NFS step 3: find small primes

Have integer m, polynomial
$f(x) = (x - m)(f_5 x^5 + \cdots + f_0)$.

Consider values $b^6 f(a/b) =$
$(a - bm)(f_5 a^5 + \cdots + f_0 b^5)$.

NFS step 3: Choose H.
For each pair (a, b)
with $b^6 f(a/b) \in [-H, H]$,
find small prime divisors
of $b^6 f(a/b)$.

at

$\cdots + f_0 b^5)$

ed,

$[-H, H]$?

,

ll primes.

phy, et al.)

ately.

#psi

nl

NFS step 3: find small primes

Have integer m, polynomial $f(x) = (x - m)(f_5 x^5 + \cdots + f_0)$.

Consider values $b^6 f(a/b) = (a - bm)(f_5 a^5 + \cdots + f_0 b^5)$.

NFS step 3: Choose H. For each pair (a, b) with $b^6 f(a/b) \in [-H, H]$, find small prime divisors of $b^6 f(a/b)$.

Simplified definitic $\leq 2^{12}$.

(Serious misconcep

"Sieving": Conside array of 2^{15} conse For each small pri Mark a's with $b^6 f(a/b)$ divisible

Can jump quickly these a's: they lie arithmetic progres

NFS step 3: find small primes

Have integer m, polynomial $f(x) = (x - m)(f_5 x^5 + \cdots + f_0)$.

Consider values $b^6 f(a/b) = (a - bm)(f_5 a^5 + \cdots + f_0 b^5)$.

NFS step 3: Choose H.
For each pair (a, b)
with $b^6 f(a/b) \in [-H, H]$,
find small prime divisors
of $b^6 f(a/b)$.

Simplified definition of "small": $\le 2^{12}$.

(Serious misconception: $\le 2^{40}$.)

"Sieving": Consider one b,
array of 2^{15} consecutive a's.
For each small prime p:
Mark a's with
$b^6 f(a/b)$ divisible by p.

Can jump quickly through
these a's: they lie in a few
arithmetic progressions mod p.

small primes

olynomial

$f_5 x^5 + \cdots + f_0$).

$f(a/b) =$

$\cdots + f_0 b^5$).

se H.

)

$-H, H$],

ivisors

Simplified definition of "small": $\leq 2^{12}$.

(Serious misconception: $\leq 2^{40}$.)

"Sieving": Consider one b, array of 2^{15} consecutive a's. For each small prime p: Mark a's with $b^6 f(a/b)$ divisible by p.

Can jump quickly through these a's: they lie in a few arithmetic progressions mod p.

Dramatically impro

adapting to CPU a

Example:

For primes $p \in [2^1$

each progression h

8 or 9 array entrie

Always mark 9 ent

often overflowing a

to eliminate branc

Simplified definition of "small": $\le 2^{12}$.

(Serious misconception: $\le 2^{40}$.)

"Sieving": Consider one b, array of 2^{15} consecutive a's.
For each small prime p:
Mark a's with $b^6 f(a/b)$ divisible by p.

Can jump quickly through these a's: they lie in a few arithmetic progressions mod p.

Dramatically improve speed by adapting to CPU architecture.

Example:
For primes $p \in [2^{15}/9, 2^{15}/8]$, each progression has 8 or 9 array entries.
Always mark 9 entries, often overflowing array, to eliminate branch mispredictions.

on of "small":

ption: $\leq 2^{40}$.)

er one b,
cutive a's.
me p:

by p.

through
in a few
sions mod p.

Dramatically improve speed by adapting to CPU architecture.

Example:
For primes $p \in [2^{15}/9, 2^{15}/8]$,
each progression has
8 or 9 array entries.
Always mark 9 entries,
often overflowing array,
to eliminate branch mispredictions.

Generalize $b^6 f(a/$
NFS can use $b^6 f($
for (a, b) in a dete
(1984 Davis Holdr
1993 Pollard)

Number of $b^6 f(a/$
is proportional to
Can choose surpris
and compensate b
(1995 Bernstein)

cr.yp.to/papers.html#

Dramatically improve speed by adapting to CPU architecture.

Example:
For primes $p \in [2^{15}/9, 2^{15}/8]$,
each progression has
8 or 9 array entries.
Always mark 9 entries,
often overflowing array,
to eliminate branch mispredictions.

Generalize $b^6 f(a/b)$:
NFS can use $b^6 f(a/b)/q$
for (a, b) in a determinant-q lattice.
(1984 Davis Holdridge,
1993 Pollard)

Number of $b^6 f(a/b)/q$ in $[-H, H]$
is proportional to $q^{-2/3}$.
Can choose surprisingly small H
and compensate by using many q's.
(1995 Bernstein)

`cr.yp.to/papers.html#mlnfs`

ove speed by
architecture.

${}^{5}/9, 2^{15}/8]$,
as
s.
tries,
array,
h mispredictions.

Generalize $b^6 f(a/b)$:
NFS can use $b^6 f(a/b)/q$
for (a, b) in a determinant-q lattice.
(1984 Davis Holdridge,
1993 Pollard)

Number of $b^6 f(a/b)/q$ in $[-H, H]$
is proportional to $q^{-2/3}$.
Can choose surprisingly small H
and compensate by using many q's.
(1995 Bernstein)

`cr.yp.to/papers.html#mlnfs`

NFS step 4: early

Have many pairs (
For each $b^6 f(a/b)$
small prime diviso
and not-yet-factor

NFS step 4: Choo
Discard all values
not-yet-factored p

How to choose L?
Balance time for s
with time for step

Generalize $b^6 f(a/b)$:
NFS can use $b^6 f(a/b)/q$
for (a, b) in a determinant-q lattice.
(1984 Davis Holdridge,
1993 Pollard)

Number of $b^6 f(a/b)/q$ in $[-H, H]$
is proportional to $q^{-2/3}$.
Can choose surprisingly small H
and compensate by using many q's.
(1995 Bernstein)

NFS step 4: early abort

Have many pairs (a, b).
For each $b^6 f(a/b)$, know
small prime divisors
and not-yet-factored part.

NFS step 4: Choose L.
Discard all values $b^6 f(a/b)$ with
not-yet-factored parts above L.

How to choose L? Answer:
Balance time for step 5
with time for step 3.

b):

$a/b)/q$

erminant-q lattice.

idge,

$/b)/q$ in $[-H, H]$

$q^{-2/3}$.

singly small H

y using many q's.

#mlnfs

NFS step 4: early abort

Have many pairs (a, b).
For each $b^6 f(a/b)$, know
small prime divisors
and not-yet-factored part.

NFS step 4: Choose L.
Discard all values $b^6 f(a/b)$ with
not-yet-factored parts above L.

How to choose L? Answer:
Balance time for step 5
with time for step 3.

NFS step 5: fully

Have some pairs (
For each value $b^6 f$
know small prime
not-yet-factored pa

NFS step 5: Ident
$b^6 f(a/b)$ that are

Should replace "2
with slightly differ
not discussed in th
(e.g. 1993 Copper

NFS step 4: early abort

Have many pairs (a, b).
For each $b^6 f(a/b)$, know small prime divisors and not-yet-factored part.

NFS step 4: Choose L.
Discard all values $b^6 f(a/b)$ with not-yet-factored parts above L.

How to choose L? Answer:
Balance time for step 5 with time for step 3.

NFS step 5: fully factor

Have some pairs (a, b).
For each value $b^6 f(a/b)$: know small prime divisors; not-yet-factored part $\leq L$.

NFS step 5: Identify values $b^6 f(a/b)$ that are 2^{40}-smooth.

Should replace "2^{40}-smooth" with slightly different notions, not discussed in this talk.
(e.g. 1993 Coppersmith)

abort

(a, b).
, know
rs
ed part.

se L.
$b^6 f(a/b)$ with
arts above L.

Answer:
tep 5
3.

NFS step 5: fully factor

Have some pairs (a, b).
For each value $b^6 f(a/b)$:
know small prime divisors;
not-yet-factored part $\le L$.

NFS step 5: Identify values
$b^6 f(a/b)$ that are 2^{40}-smooth.

Should replace "2^{40}-smooth"
with slightly different notions,
not discussed in this talk.
(e.g. 1993 Coppersmith)

Assume that origi
are smooth with p
step 3 spends tim
step 5 spends tim

With proper balan
time roughly $RT($
to find one smoot
(1982 Pomerance)

Want 12 as large a
to move from RT
But want 12 belo
and want 15 small
sieving fits into L1

`cr.yp.to/bib/entries.`

NFS step 5: fully factor

Have some pairs (a, b).
For each value $b^6 f(a/b)$:
know small prime divisors;
not-yet-factored part $\le L$.

NFS step 5: Identify values
$b^6 f(a/b)$ that are 2^{40}-smooth.

Should replace "2^{40}-smooth"
with slightly different notions,
not discussed in this talk.
(e.g. 1993 Coppersmith)

Assume that original values
are smooth with probability $1/R$;
step 3 spends time S per value;
step 5 spends time T per value.

With proper balance,
time roughly $RT(S/T)^{12/40}$
to find one smooth value.
(1982 Pomerance)

Want 12 as large as possible
to move from RT towards RS.
But want 12 below 15,
and want 15 small so that
sieving fits into L1 cache.

`cr.yp.to/bib/entries.html#1982/pomerance` factor

a, b).
$f(a/b)$:
divisors;
art $\leq L$.

ify values
2^{40}-smooth.

^{40}-smooth"
ent notions,
his talk.
smith)

Assume that original values
are smooth with probability $1/R$;
step 3 spends time S per value;
step 5 spends time T per value.

With proper balance,
time roughly $RT(S/T)^{12/40}$
to find one smooth value.
(1982 Pomerance)

Want 12 as large as possible
to move from RT towards RS.
But want 12 below 15,
and want 15 small so that
sieving fits into L1 cache.

`cr.yp.to/bib/entries.html#1982/pomerance`

Traditional algorit
For each pair $(a, b$
use ECM to find p
dividing $b^6 f(a/b)$.
Complications sav
"rho," more abort

Much faster to ha
a big batch of pai
(2000 Bernstein)

Save even more ti
smoothness witho
primes. (2004 Fra
Morain Wirth)

Assume that original values are smooth with probability $1/R$; step 3 spends time S per value; step 5 spends time T per value.

With proper balance, time roughly $RT(S/T)^{12/40}$ to find one smooth value. (1982 Pomerance)

Want 12 as large as possible to move from RT towards RS. But want 12 below 15, and want 15 small so that sieving fits into L1 cache.

`cr.yp.to/bib/entries.html#1982/pomerance`

Traditional algorithm for step 5: For each pair (a, b) separately, use ECM to find primes $\leq 2^{40}$ dividing $b^6 f(a/b)$.

Complications save time: "rho," more aborts, et al.

Much faster to handle a big batch of pairs (a, b). (2000 Bernstein)

Save even more time by checking smoothness without first finding primes. (2004 Franke Kleinjung Morain Wirth)

nal values
robability $1/R$;
e S per value;
e T per value.

ce,
$S/T)^{12/40}$
h value.

as possible
towards RS.
v 15,
so that
cache.

`html#1982/pomerance`

Traditional algorithm for step 5: For each pair (a, b) separately, use ECM to find primes $\leq 2^{40}$ dividing $b^6 f(a/b)$. Complications save time: "rho," more aborts, et al.

Much faster to handle a big batch of pairs (a, b). (2000 Bernstein)

Save even more time by checking smoothness without first finding primes. (2004 Franke Kleinjung Morain Wirth)

Streamlined batch
(2004 Bernstein):

Multiply primes $\leq$
in pairs, pairs of p
to obtain their pro
Relies on fast disk
multiplication of h

Compute P mod v
Relies on fast divis

Now a value v is s
$(P \bmod v)^{2^{\lceil \lg \lg v \rceil}}$

`cr.yp.to/papers.html#`

Traditional algorithm for step 5:
For each pair (a, b) separately,
use ECM to find primes $\leq 2^{40}$
dividing $b^6 f(a/b)$.
Complications save time:
"rho," more aborts, et al.

Much faster to handle
a big batch of pairs (a, b).
(2000 Bernstein)

Save even more time by checking
smoothness without first finding
primes. (2004 Franke Kleinjung
Morain Wirth)

Streamlined batch algorithm
(2004 Bernstein):

Multiply primes $\leq 2^{40}$
in pairs, pairs of pairs, etc.,
to obtain their product P.
Relies on fast disk-based
multiplication of huge integers.

Compute $P \bmod v$ for each value v.
Relies on fast division.

Now a value v is smooth iff
$(P \bmod v)^{2^{\lceil \lg \lg v \rceil}} \bmod v = 0$.

`cr.yp.to/papers.html#smoothparts`

hm for step 5:
) separately,
rimes $\leq 2^{40}$

e time:
s, et al.

ndle
rs (a, b).

me by checking
ut first finding
nke Kleinjung

Streamlined batch algorithm
(2004 Bernstein):

Multiply primes $\leq 2^{40}$
in pairs, pairs of pairs, etc.,
to obtain their product P.
Relies on fast disk-based
multiplication of huge integers.

Compute $P \bmod v$ for each value v.
Relies on fast division.

Now a value v is smooth iff
$(P \bmod v)^{2^{\lceil \lg \lg v \rceil}} \bmod v = 0$.

cr.yp.to/papers.html#smoothparts

Many lower-level s

Compute P with "
≈ 1.5 times faster

Compute $P \bmod v$
"scaled remainder
≈ 2.6 times faster
(2004 Bernstein, a
2003 Bostan Lecer

Reduce communic
(2004–2005 Berns

cr.yp.to/papers.html#
cr.yp.to/papers.html#

Streamlined batch algorithm (2004 Bernstein):

Multiply primes $\leq 2^{40}$ in pairs, pairs of pairs, etc., to obtain their product P. Relies on fast disk-based multiplication of huge integers.

Compute P mod v for each value v. Relies on fast division.

Now a value v is smooth iff $(P \bmod v)^{2^{\lceil \lg \lg v \rceil}} \bmod v = 0$.

`cr.yp.to/papers.html#smoothparts`

Many lower-level speedups.

Compute P with "FFT doubling": ≈ 1.5 times faster. (2004 Kramer)

Compute P mod v with "scaled remainder tree": ≈ 2.6 times faster. (2004 Bernstein, adapting 2003 Bostan Lecerf Schost)

Reduce communication costs. (2004–2005 Bernstein)

`cr.yp.to/papers.html#multapps`
`cr.yp.to/papers.html#scaledmod`

algorithm

2^{40}

airs, etc.,

oduct P.

-based

uge integers.

v for each value v.

sion.

smooth iff

mod $v = 0$.

#smoothparts

Many lower-level speedups.

Compute P with "FFT doubling": ≈ 1.5 times faster. (2004 Kramer)

Compute $P \bmod v$ with "scaled remainder tree": ≈ 2.6 times faster. (2004 Bernstein, adapting 2003 Bostan Lecerf Schost)

Reduce communication costs. (2004–2005 Bernstein)

cr.yp.to/papers.html#multapps
cr.yp.to/papers.html#scaledmod

Contrary to popula

properly designed

computers can dra

price-performance

Huge improvemen

(2001 Bernstein)

The batch algorith

on today's badly d

(Pentium, PowerP

but will eventually

http://www.shar

cryptanalytic-hard

cr.yp.to/talks.html#2

cr.yp.to/papers.html#

Many lower-level speedups.

Compute P with "FFT doubling": ≈ 1.5 times faster. (2004 Kramer)

Compute $P \bmod v$ with "scaled remainder tree": ≈ 2.6 times faster. (2004 Bernstein, adapting 2003 Bostan Lecerf Schost)

Reduce communication costs. (2004–2005 Bernstein)

`cr.yp.to/papers.html#multapps`
`cr.yp.to/papers.html#scaledmod`

Contrary to popular myth, properly designed parallel computers can dramatically improve price-performance ratio.

Huge improvement for ECM etc. (2001 Bernstein)

The batch algorithms are better on today's badly designed CPUs (Pentium, PowerPC, Athlon, etc.) but will eventually be obsolete.

`http://www.sharcs.org`: new cryptanalytic-hardware workshop.

`cr.yp.to/talks.html#2005.06.11-1`
`cr.yp.to/papers.html#nfscircuit`

speedups.

"FFT doubling":
. (2004 Kramer)

v with
tree":
.
adapting
rf Schost)

ation costs.
tein)

#multapps
#scaledmod

Contrary to popular myth, properly designed parallel computers can dramatically improve price-performance ratio.

Huge improvement for ECM etc. (2001 Bernstein)

The batch algorithms are better on today's badly designed CPUs (Pentium, PowerPC, Athlon, etc.) but will eventually be obsolete.

`http://www.sharcs.org`: new cryptanalytic-hardware workshop.

`cr.yp.to/talks.html#2005.06.11-1`
`cr.yp.to/papers.html#nfscircuit`

NFS step 6: linear

Have some pairs (
with complete fact
of the values $b^6 f($

NFS step 6: Find
of pairs (a, b) for
$a - b\alpha$ both have
Here $\alpha \neq m$ is a

Do this by finding
dependency amon
Guaranteed to suc
if there are enough

Contrary to popular myth, properly designed parallel computers can dramatically improve price-performance ratio.

Huge improvement for ECM etc. (2001 Bernstein)

The batch algorithms are better on today's badly designed CPUs (Pentium, PowerPC, Athlon, etc.) but will eventually be obsolete.

`http://www.sharcs.org`: new cryptanalytic-hardware workshop.

`cr.yp.to/talks.html#2005.06.11-1`
`cr.yp.to/papers.html#nfscircuit`

NFS step 6: linear algebra

Have some pairs (a, b) with complete factorizations of the values $b^6 f(a/b)$.

NFS step 6: Find nonempty subset of pairs (a, b) for which $a - bm$ and $a - b\alpha$ both have square product. Here $\alpha \neq m$ is a root of f.

Do this by finding a linear dependency among vectors mod 2. Guaranteed to succeed if there are enough vectors.

ar myth,
parallel
matically improve
ratio.

t for ECM etc.

ms are better
esigned CPUs
C, Athlon, etc.)
be obsolete.

`rcs.org`: new
ware workshop.

`2005.06.11-1`
`nfscircuit`

NFS step 6: linear algebra

Have some pairs (a, b)
with complete factorizations
of the values $b^6 f(a/b)$.

NFS step 6: Find nonempty subset
of pairs (a, b) for which $a - bm$ and
$a - b\alpha$ both have square product.
Here $\alpha \neq m$ is a root of f.

Do this by finding a linear
dependency among vectors mod 2.
Guaranteed to succeed
if there are enough vectors.

Choose prime bou
to minimize total
linear algebra and

Larger bound wou
of previous steps,
algebra would be
Reduce bound to
algebra with previ

This balancing me
somewhat less imp
speedups in partic

NFS step 6: linear algebra

Have some pairs (a, b) with complete factorizations of the values $b^6 f(a/b)$.

NFS step 6: Find nonempty subset of pairs (a, b) for which $a - bm$ and $a - b\alpha$ both have square product. Here $\alpha \neq m$ is a root of f.

Do this by finding a linear dependency among vectors mod 2. Guaranteed to succeed if there are enough vectors.

Choose prime bound 2^{40} to minimize total time of linear algebra and previous steps.

Larger bound would minimize time of previous steps, but then linear algebra would be a bottleneck. Reduce bound to balance linear algebra with previous steps.

This balancing means somewhat less impact of speedups in particular steps.

r algebra

$a, b)$
torizations
$a/b)$.

nonempty subset
which $a - bm$ and
square product.
root of f.

a linear
g vectors mod 2.
ceed
n vectors.

Choose prime bound 2^{40}
to minimize total time of
linear algebra and previous steps.

Larger bound would minimize time
of previous steps, but then linear
algebra would be a bottleneck.
Reduce bound to balance linear
algebra with previous steps.

This balancing means
somewhat less impact of
speedups in particular steps.

NFS step 7: squar

Have some pairs (
Product of $a - bm$
Product of $a - b\alpha$

NFS step 7: Use p
factor n, maybe n

Simplest method,
$\sqrt{\prod(a - b\alpha)}$, is n
Other methods in
waste of programm

Choose prime bound 2^{40}
to minimize total time of
linear algebra and previous steps.

Larger bound would minimize time of previous steps, but then linear algebra would be a bottleneck. Reduce bound to balance linear algebra with previous steps.

This balancing means somewhat less impact of speedups in particular steps.

NFS step 7: square roots

Have some pairs (a, b).
Product of $a - bm$ is square.
Product of $a - b\alpha$ is square.

NFS step 7: Use pairs to factor n, maybe nontrivially.

Simplest method, computing $\sqrt{\prod(a - b\alpha)}$, is not a bottleneck. Other methods in literature are a waste of programmer time.