BILL START RECORDING

Quadratic Sieve Factoring

Notation Reminder

1) $\mathbf{GCD(x, y)}$ is the **Greatest Common Divisor** of x, y.

Notation Reminder

1) $\mathbf{GCD(x,y)}$ is the **Greatest Common Divisor** of x, y.
2) **Sums and Products**

$$\sum_{i=1}^{n} a_i = a_1 + a_2 + \cdots + a_n.$$

$$\prod_{i=1}^{n} a_i = a_1 \times a_2 \times \cdots \times a_n.$$

Notation Reminder

1) $\mathrm{GCD}(\mathbf{x},\mathbf{y})$ is the **Greatest Common Divisor** of x, y.

2) **Sums and Products**

$$\sum_{i=1}^{n} a_i = a_1 + a_2 + \cdots + a_n.$$

$$\prod_{i=1}^{n} a_i = a_1 \times a_2 \times \cdots \times a_n.$$

3) **More Sums and Products** We **summed** or **producted** over $\{1, \ldots, n\}$. Can use other sets.

Notation Reminder

1) $\mathrm{GCD}(\mathbf{x},\mathbf{y})$ is the **Greatest Common Divisor** of x, y.

2) **Sums and Products**

$$\sum_{i=1}^{n} a_i = a_1 + a_2 + \cdots + a_n.$$

$$\prod_{i=1}^{n} a_i = a_1 \times a_2 \times \cdots \times a_n.$$

3) **More Sums and Products** We **summed** or **producted** over $\{1, \ldots, n\}$. Can use other sets.
If $A = \{1, 4, 9\}$ then

$$\sum_{i \in A} a_i = a_1 + a_4 + a_9.$$

$$\prod_{i \in A} a_i = a_1 \times a_4 \times a_9.$$

More Notation Reminder

4) $a_1, \dots, a_n$ could be **vectors.**

$$\sum_{i \in A} \vec{a}_i = \vec{a}_1 + \vec{a}_4 + \vec{a}_9.$$

Addition is **component-wise.**

More Notation Reminder

4) $a_1, \ldots, a_n$ could be **vectors.**

$$\sum_{i \in A} \vec{a}_i = \vec{a}_1 + \vec{a}_4 + \vec{a}_9.$$

Addition is **component-wise.**

We will not be using any notion of a product of vectors.

More Notation Reminder

4) $a_1, \ldots, a_n$ could be **vectors.**

$$\sum_{i \in A} \vec{a}_i = \vec{a}_1 + \vec{a}_4 + \vec{a}_9.$$

Addition is **component-wise.**
We will not be using any notion of a product of vectors.

5) We extend mod notation to vectors of integers. Example:

$$(8, 1, 0, 9) \pmod{2} = (0, 1, 0, 1).$$

Back from our Aside on Sieves

Time Analysis of Sieve of E

The Sieve of E can find all primes $\leq N$ in time

$$\leq N \sum_{p \leq N} \frac{1}{p} \sim N \ln(\ln(N))$$

Time Analysis of Sieve of E

The Sieve of E can find all primes $\le N$ in time

$$\le N \sum_{p \le N} \frac{1}{p} \sim N \ln(\ln(N))$$

How long would finding all primes $\le N$ be the stupid way?

Testing if a number is prime takes $(\log n)^3$ steps (we did not do this in class) So testing all numbers $n \le N$ for primality takes time:

$$\sum_{n \le N} (\log n)^3 \sim N(\log N)^3$$

Time Analysis of Sieve of E

The Sieve of E can find all primes $\le N$ in time

$$\le N \sum_{p \le N} \frac{1}{p} \sim N \ln(\ln(N))$$

How long would finding all primes $\le N$ be the stupid way?

Testing if a number is prime takes $(\log n)^3$ steps (we did not do this in class) So testing all numbers $n \le N$ for primality takes time:

$$\sum_{n \le N} (\log n)^3 \sim N(\log N)^3$$

- Time diff not impressive. When we modify the Sieve to actually factor, it will be much more impressive.

Time Analysis of Sieve of E

The Sieve of E can find all primes $\le N$ in time

$$\le N \sum_{p \le N} \frac{1}{p} \sim N \ln(\ln(N))$$

How long would finding all primes $\le N$ be the stupid way?

Testing if a number is prime takes $(\log n)^3$ steps (we did not do this in class) So testing all numbers $n \le N$ for primality takes time:

$$\sum_{n \le N} (\log n)^3 \sim N(\log N)^3$$

- Time diff not impressive. When we modify the Sieve to actually factor, it will be much more impressive.
- The key to the speed of The Sieve of E is that when it marks it DOES NOT look at (say) 3 and say **Oh, thats not even** . It literally does not look at all!

The B-Factoring Sieve of E: Example

The Sieve of E **marked** all evens.
Better Divide by 2 knowing it will work. Then divide by 2 again (it might not work) until factor out all powers of 2.

The Sieve of E **marked** all numbers $\equiv 0 \pmod 3$
Better Divide by 3 knowing it will work. Then divide by 3 again (it might not work) until factor out all powers of 3.

Do this for the first B primes and you will have B-factored many numbers.

B-factoring all $N \leq 48$, the Smart Way

Write down numbers ≤ 48. We 2-factor them, so divide by 2,3.

2	3	4	5	6	7	8	9	10	11	12	13	14	15

16	17	18	19	20	21	22	23	24	25	26	27

28	29	30	31	32	33	34	35	36	37	38	39

40	41	42	43	44	45	46	47	48

B-factoring all $N \leq 48$, the Smart Way

Write down numbers ≤ 48. We 2-factor them, so divide by 2,3.

2	3	4	5	6	7	8	9	10	11	12	13	14	15

16	17	18	19	20	21	22	23	24	25	26	27

28	29	30	31	32	33	34	35	36	37	38	39

40	41	42	43	44	45	46	47	48

First unmarked is 2. DIVIDE mults of 2 by 2.

Divide by 2, Repeatedly

2	3	4	5	6	7	8	9	10	11	12	13	14	15
$2*1$		$2*2$		$2*3$		2^3		$2*5$		2^2*3		$2*7$	

16	17	18	19	20	21	22	23	24	25	26	27
2^4		$2*9$		2^2*5		$2*11$		2^3*3		$2*13$	

28	29	30	31	32	33	34	35	36	37	38	39
2^2*7		$2*15$		2^5		$2*17$		2^2*9		$2*19$	

40	41	42	43	44	45	46	47	48
2^3*5		$2*21$		2^2*11		$2*23$		2^4*3

First unmarked is 2. DIVIDE mults of 3 by 3.

Divide by 3, Repeatedly

We only show the last row (for reasons of space).

40	41	42	43	44	45	46	47	48
$2^3 * 5$		$2 * 3 * 7$		$2^2 * 11$	$3^2 * 5$	$2 * 23$		$2^4 * 3$

- 48 was 2-factored
- Nothing else was.

The B-Factoring Sieve of E: Algorithm

1. Input(N, B)
2. Write down $2, 3, \ldots, N$. All are blank in box.
3. (BOX STEP) Goto the first blank box, p. (When have visited this step B times then stop).
4. Factor out p from p, $2p$, …, $\left\lfloor \frac{N}{p} \right\rfloor p$.

   Factor out p from p^2, $2p^2$, …, $\left\lfloor \frac{N}{p^2} \right\rfloor p^2$

   Factor out ….
5. GOTO BOX STEP.

Time:

$$\sum_{p \le B} \frac{N}{p} + \sum_{p \le B} \frac{N}{p^2} + \sum_{p \le B} \frac{N}{p^3} + \sum_{p \le B} \frac{N}{p^4} \cdots$$

$$= N\left(\sum_{p \le B} \frac{1}{p} + \sum_{p \le B} \frac{1}{p^2} + \sum_{p \le B} \frac{1}{p^3} + \sum_{p \le B} \frac{1}{p^4} + \cdots\right)$$

The B-Factoring Sieve of E: Analysis

$$= N\left(\sum_{p\le B}\frac{1}{p}+\sum_{p\le B}\frac{1}{p^2}+\sum_{p\le B}\frac{1}{p^3}+\sum_{p\le B}\frac{1}{p^4}+\cdots\right)$$

$$N\sum_{p\le B}\frac{1}{p}+N\sum_{p\le B}\frac{1}{p^2}+N\sum_{p\le B}\frac{1}{p^3}+N\sum_{p\le B}\frac{1}{p^4}+\cdots$$

$$= N\ln(\ln(B))+N\sum_{a=2}^{\infty}\sum_{p\le B}\frac{1}{p^a}$$

Next slide shows that $N\sum_{a=2}^{\infty}\sum_{p\le B}\frac{1}{p^a}\le(0.5)N$, so time is

The B-Factoring Sieve of E: Analysis

$$= N\left(\sum_{p\le B}\frac{1}{p}+\sum_{p\le B}\frac{1}{p^2}+\sum_{p\le B}\frac{1}{p^3}+\sum_{p\le B}\frac{1}{p^4}+\cdots\right)$$

$$N\sum_{p\le B}\frac{1}{p}+N\sum_{p\le B}\frac{1}{p^2}+N\sum_{p\le B}\frac{1}{p^3}+N\sum_{p\le B}\frac{1}{p^4}+\cdots$$

$$= N\ln(\ln(B))+N\sum_{a=2}^{\infty}\sum_{p\le B}\frac{1}{p^a}$$

Next slide shows that $N\sum_{a=2}^{\infty}\sum_{p\le B}\frac{1}{p^a}\le(0.5)N$, so time is

$$\le N\ln(\ln(B))+(0.5)N.$$

Note: The mult constants really are ≤ 1 and it does matter for real world performance.

The B-Factoring The Sieve of E: Last term is $\leq N$

$$= N \sum_{a=2}^{\infty} \sum_{p \leq B} \frac{1}{p^a} = N \sum_{p \leq B} \sum_{a=2}^{\infty} \frac{1}{p^a}$$

The B-Factoring The Sieve of E: Last term is $\leq N$

$$= N \sum_{a=2}^{\infty} \sum_{p \leq B} \frac{1}{p^a} = N \sum_{p \leq B} \sum_{a=2}^{\infty} \frac{1}{p^a}$$

$$= N \sum_{p \leq B} \frac{1/p^2}{1 - (1/p)}$$

$$= N \sum_{p \leq B} \frac{1}{p^2 - p} \sim N \sum_{p \leq B} \frac{1}{p^2}$$

How big is $\sum_{p \leq B} \frac{1}{p^2}$?

The B-Factoring The Sieve of E: Last term is $\leq N$

$$= N \sum_{a=2}^{\infty} \sum_{p \leq B} \frac{1}{p^a} = N \sum_{p \leq B} \sum_{a=2}^{\infty} \frac{1}{p^a}$$

$$= N \sum_{p \leq B} \frac{1/p^2}{1 - (1/p)}$$

$$= N \sum_{p \leq B} \frac{1}{p^2 - p} \sim N \sum_{p \leq B} \frac{1}{p^2}$$

How big is $\sum_{p \leq B} \frac{1}{p^2}$?

1. $\sum_{n=1}^{\infty} \frac{1}{n^2}$ cvg. Do you know to what?

The B-Factoring The Sieve of E: Last term is $\le N$

$$= N \sum_{a=2}^{\infty} \sum_{p \le B} \frac{1}{p^a} = N \sum_{p \le B} \sum_{a=2}^{\infty} \frac{1}{p^a}$$

$$= N \sum_{p \le B} \frac{1/p^2}{1-(1/p)}$$

$$= N \sum_{p \le B} \frac{1}{p^2 - p} \sim N \sum_{p \le B} \frac{1}{p^2}$$

How big is $\sum_{p \le B} \frac{1}{p^2}$?

1. $\sum_{n=1}^{\infty} \frac{1}{n^2}$ cvg. Do you know to what? $\frac{\pi^2}{6} \sim 1.644$

The B-Factoring The Sieve of E: Last term is $\le N$

$$= N \sum_{a=2}^{\infty} \sum_{p \le B} \frac{1}{p^a} = N \sum_{p \le B} \sum_{a=2}^{\infty} \frac{1}{p^a}$$

$$= N \sum_{p \le B} \frac{1/p^2}{1 - (1/p)}$$

$$= N \sum_{p \le B} \frac{1}{p^2 - p} \sim N \sum_{p \le B} \frac{1}{p^2}$$

How big is $\sum_{p \le B} \frac{1}{p^2}$?

1. $\sum_{n=1}^{\infty} \frac{1}{n^2}$ cvg. Do you know to what? $\frac{\pi^2}{6} \sim 1.644$
2. $\sum_{p=1}^{\infty} \frac{1}{p^2}$ cvg. Do you know to what?

The B-Factoring The Sieve of E: Last term is $\le N$

$$= N\sum_{a=2}^{\infty}\sum_{p\le B}\frac{1}{p^a} = N\sum_{p\le B}\sum_{a=2}^{\infty}\frac{1}{p^a}$$

$$= N\sum_{p\le B}\frac{1/p^2}{1-(1/p)}$$

$$= N\sum_{p\le B}\frac{1}{p^2-p} \sim N\sum_{p\le B}\frac{1}{p^2}$$

How big is $\sum_{p\le B}\frac{1}{p^2}$?

1. $\sum_{n=1}^{\infty}\frac{1}{n^2}$ cvg. Do you know to what? $\frac{\pi^2}{6}\sim 1.644$
2. $\sum_{p=1}^{\infty}\frac{1}{p^2}$ cvg. Do you know to what? ~ 0.45.

Time For The Factoring Sieve of E VS Naive Alg

Given N, B want to B-factor $\{2, \ldots, N\}$.

Time For The Factoring Sieve of E VS Naive Alg

Given N, B want to B-factor $\{2, \ldots, N\}$.

Naive Algorithm B-factor 2, B-factor 3, …, B-factor N. To B-factor x takes $\sim B$. So this takes time:

$$O(NB).$$

Time For The Factoring Sieve of E VS Naive Alg

Given N, B want to B-factor $\{2, \ldots, N\}$.

Naive Algorithm B-factor 2, B-factor 3, …, B-factor N. To B-factor x takes $\sim B$. So this takes time:

$$O(NB).$$

The B-Factoring Sieve of E takes time:

$$\leq N \ln(\ln(B)) + 0.5N$$

Time For The Factoring Sieve of E VS Naive Alg

Given N, B want to B-factor $\{2, \ldots, N\}$.

Naive Algorithm B-factor 2, B-factor 3, ..., B-factor N. To B-factor x takes $\sim B$. So this takes time:

$$O(NB).$$

The B-Factoring Sieve of E takes time:

$$\leq N \ln(\ln(B)) + 0.5N$$

This is much better since often $B \sim N^a$ for some $0 < a < 1$. Can easily modify to get a fast algorithm for B-factoring $N_1, \ldots, N_1 + N$.

Variants of The B-Factoring Sieve of E

Can easily modify to get a fast algorithm for the following:
Given N_1, B, N, B-factoring $N_1, N_1 + 1, \ldots, N_1 + N$.

Variants of The B-Factoring Sieve of E

Can easily modify to get a fast algorithm for the following:
Given N_1, B, N, B-factoring $N_1, N_1+1, \ldots, N_1+N$.

Time will still be $\le N\ln(\ln(B)) + 0.5N$.

This is not the problem we originally needed to solve, though it's close. We now go back to our original problem.

Back to Quadratic Sieve Factoring Algorithm

Recall Quad Sieve Alg: First Attempt

Given N let $x = \left\lceil \sqrt{N} \right\rceil$. All $\equiv$ are mod N. B, M are params.

$$
\begin{array}{rl}
(x+0)^2 \equiv y_0 & \text{Try to } B\text{-Factor } y_0 \text{ to get parity } \vec{v}_0 \\
\vdots & \vdots \\
(x+M)^2 \equiv y_M & \text{Try to } B\text{-Factor } y_M \text{ to get parity } \vec{v}_M
\end{array}
$$

Recall Quad Sieve Alg: First Attempt

Given N let $x = \left\lceil \sqrt{N} \right\rceil$. All $\equiv$ are mod N. B, M are params.

$$
\begin{array}{rl}
(x+0)^2 \equiv y_0 & \text{Try to } B\text{-Factor } y_0 \text{ to get parity } \vec{v}_0 \\
\vdots & \vdots \\
(x+M)^2 \equiv y_M & \text{Try to } B\text{-Factor } y_M \text{ to get parity } \vec{v}_M
\end{array}
$$

STOP

Recall Quad Sieve Alg: First Attempt

Given N let $x = \left\lceil \sqrt{N} \right\rceil$. All $\equiv$ are mod N. B, M are params.

$$\begin{array}{rl} (x+0)^2 \equiv y_0 & \text{Try to } B\text{-Factor } y_0 \text{ to get parity } \vec{v}_0 \\ \vdots & \vdots \\ (x+M)^2 \equiv y_M & \text{Try to } B\text{-Factor } y_M \text{ to get parity } \vec{v}_M \end{array}$$

STOP

1. We just spend a long aside on B-factoring, in bulk,

$$N_1, N_1+1, \ldots, N_1+N.$$

Recall Quad Sieve Alg: First Attempt

Given N let $x = \left\lceil \sqrt{N} \right\rceil$. All $\equiv$ are mod N. B, M are params.

$$\begin{array}{rl} (x+0)^2 \equiv y_0 & \text{Try to } B\text{-Factor } y_0 \text{ to get parity } \vec{v}_0 \\ \vdots & \vdots \\ (x+M)^2 \equiv y_M & \text{Try to } B\text{-Factor } y_M \text{ to get parity } \vec{v}_M \end{array}$$

STOP

1. We just spend a long aside on B-factoring, in bulk,

$$N_1, N_1+1, \ldots, N_1+N.$$

2. The problem we need solved is similar: B-factor, in bulk.

$$(x+0)^2 \pmod{N}, (x+1)^2 \pmod{N}, \ldots, (x+M)^2 \pmod{N}.$$

Recall Quad Sieve Alg: First Attempt

Given N let $x = \left\lceil \sqrt{N} \right\rceil$. All $\equiv$ are mod N. B, M are params.

$$\begin{array}{rl} (x+0)^2 \equiv y_0 & \text{Try to } B\text{-Factor } y_0 \text{ to get parity } \vec{v}_0 \\ \vdots & \vdots \\ (x+M)^2 \equiv y_M & \text{Try to } B\text{-Factor } y_M \text{ to get parity } \vec{v}_M \end{array}$$

STOP

1. We just spend a long aside on B-factoring, in bulk,

$$N_1, N_1+1, \ldots, N_1+N.$$

2. The problem we need solved is similar: B-factor, in bulk.

$$(x+0)^2 \pmod{N}, (x+1)^2 \pmod{N}, \ldots, (x+M)^2 \pmod{N}.$$

But before we do that, lets go back to the algorithm and remind ourselves what it does.

Recall Quad Sieve Alg: First Attempt (Again)

Given N let $x = \left\lceil \sqrt{N} \right\rceil$. All $\equiv$ are mod N. B, M are params.

$$
\begin{array}{rl}
(x+0)^2 \equiv y_0 & \text{Try to } B\text{-Factor } y_0 \text{ to get parity } \vec{v}_0 \\
\vdots & \vdots \\
(x+M)^2 \equiv y_M & \text{Try to } B\text{-Factor } y_M \text{ to get parity } \vec{v}_M
\end{array}
$$

Recall Quad Sieve Alg: First Attempt (Again)

Given N let $x = \left\lceil \sqrt{N} \right\rceil$. All $\equiv$ are mod N. B, M are params.

$$\begin{array}{rl} (x+0)^2 \equiv y_0 & \text{Try to } B\text{-Factor } y_0 \text{ to get parity } \vec{v}_0 \\ \vdots & \vdots \\ (x+M)^2 \equiv y_M & \text{Try to } B\text{-Factor } y_M \text{ to get parity } \vec{v}_M \end{array}$$

$I \subseteq \{0, \ldots, M\}$ s.t. $(\forall i \in I)$, y_i is B-factored. Find $J \subseteq I$ such that $\sum_{i \in J} \vec{v}_i = \vec{0}$, so $\prod_{i \in J} y_i$ has even exponents, so:

Recall Quad Sieve Alg: First Attempt (Again)

Given N let $x = \left\lceil \sqrt{N} \right\rceil$. All $\equiv$ are mod N. B, M are params.

$$\begin{array}{rl} (x+0)^2 \equiv y_0 & \text{Try to } B\text{-Factor } y_0 \text{ to get parity } \vec{v}_0 \\ \vdots & \vdots \\ (x+M)^2 \equiv y_M & \text{Try to } B\text{-Factor } y_M \text{ to get parity } \vec{v}_M \end{array}$$

$I \subseteq \{0, \ldots, M\}$ s.t. $(\forall i \in I)$, y_i is B-factored. Find $J \subseteq I$ such that $\sum_{i \in J} \vec{v}_i = \vec{0}$, so $\prod_{i \in J} y_i$ has even exponents, so:

$$\prod_{i \in J} y_i = Y^2$$

$$(\prod_{i \in J} (x+i))^2 \equiv \prod_{i \in J} y_i = Y^2 \pmod{N}$$

Let $X = \prod_{i \in J}(x+i) \pmod{N}$ and $Y = \prod_{i=1}^{B} q_i^{e_i} \pmod{N}$.

Recall Quad Sieve Alg: First Attempt (Again)

Given N let $x = \left\lceil \sqrt{N} \right\rceil$. All $\equiv$ are mod N. B, M are params.

$$\begin{array}{rl} (x+0)^2 \equiv y_0 & \text{Try to } B\text{-Factor } y_0 \text{ to get parity } \vec{v}_0 \\ \vdots & \vdots \\ (x+M)^2 \equiv y_M & \text{Try to } B\text{-Factor } y_M \text{ to get parity } \vec{v}_M \end{array}$$

$I \subseteq \{0, \ldots, M\}$ s.t. $(\forall i \in I)$, y_i is B-factored. Find $J \subseteq I$ such that $\sum_{i \in J} \vec{v}_i = \vec{0}$, so $\prod_{i \in J} y_i$ has even exponents, so:

$$\prod_{i \in J} y_i = Y^2$$

$$\left(\prod_{i \in J} (x+i)\right)^2 \equiv \prod_{i \in J} y_i = Y^2 \pmod{N}$$

Let $X = \prod_{i \in J}(x+i) \pmod{N}$ and $Y = \prod_{i=1}^{B} q_i^{e_i} \pmod{N}$.

$$X^2 - Y^2 \equiv 0 \pmod{N}.$$

$\mathrm{GCD}(X - Y, N)$, $\mathrm{GCD}(X + Y, N)$ should yield factors.

Recall Quad Sieve Alg: First Attempt, First Step

Given N let $x = \left\lceil \sqrt{N} \right\rceil$. All $\equiv$ are mod N. B, M are params.

$$\begin{array}{rl}
(x+0)^2 \equiv y_0 & \text{Try to } B\text{-Factor } y_0 \text{ to get parity } \vec{v}_0 \\
\vdots & \vdots \\
(x+M)^2 \equiv y_M & \text{Try to } B\text{-Factor } y_M \text{ to get parity } \vec{v}_M
\end{array}$$

How do we B-factor all of those numbers?

Recall Quad Sieve Alg: First Attempt, First Step

Given N let $x = \left\lceil \sqrt{N} \right\rceil$. All $\equiv$ are mod N. B, M are params.

$$
\begin{array}{ll}
(x+0)^2 \equiv y_0 & \text{Try to } B\text{-Factor } y_0 \text{ to get parity } \vec{v}_0 \\
\vdots & \vdots \\
(x+M)^2 \equiv y_M & \text{Try to } B\text{-Factor } y_M \text{ to get parity } \vec{v}_M
\end{array}
$$

How do we B-factor all of those numbers?
Modified Sieve of E B-factored $N_1 + 1, \ldots, N_1 + N$.

Recall Quad Sieve Alg: First Attempt, First Step

Given N let $x = \left\lceil \sqrt{N} \right\rceil$. All $\equiv$ are mod N. B, M are params.

$$\begin{array}{rl}
(x+0)^2 \equiv y_0 & \text{Try to } B\text{-Factor } y_0 \text{ to get parity } \vec{v}_0 \\
\vdots & \vdots \\
(x+M)^2 \equiv y_M & \text{Try to } B\text{-Factor } y_M \text{ to get parity } \vec{v}_M
\end{array}$$

How do we B-factor all of those numbers?
Modified Sieve of E B-factored $N_1+1, \ldots, N_1+N$.
We need to B-factor $y_0, y_1, \ldots, y_M$.

Recall Quad Sieve Alg: First Attempt, First Step

Given N let $x = \left\lceil \sqrt{N} \right\rceil$. All $\equiv$ are mod N. B, M are params.

$$\begin{array}{rl} (x+0)^2 \equiv y_0 & \text{Try to } B\text{-Factor } y_0 \text{ to get parity } \vec{v}_0 \\ \vdots & \vdots \\ (x+M)^2 \equiv y_M & \text{Try to } B\text{-Factor } y_M \text{ to get parity } \vec{v}_M \end{array}$$

How do we B-factor all of those numbers?
Modified Sieve of E B-factored $N_1+1, \ldots, N_1+N$.
We need to B-factor $y_0, y_1, \ldots, y_M$.

Plan It was more efficient to B-factor $2, \ldots, N$ all at once then one at at time. Same will be true for $y_0, \ldots, y_M$.

The Quadratic Sieve: The Problem

New Problem Given N, B, M, x, want to B-factor

$$
\begin{array}{l}
(x+0)^2 \pmod{N} \\
(x+1)^2 \pmod{N} \\
\qquad \vdots \qquad \vdots \\
(x+M)^2 \pmod{N}
\end{array}
$$

We do an example on the next slide.

The Quadratic Sieve: Example

$N = 1147$, $B = 2$, $M = 10$, $x = 34$.
Want to 2-factor (so all powers of 2 and 3)

$$
\begin{array}{l}
(34+0)^2 \pmod{1147} \\
\quad \vdots \qquad \vdots \qquad \vdots \\
(34+10)^2 \pmod{1147}
\end{array}
$$

The Quadratic Sieve: Example

$N = 1147$, $B = 2$, $M = 10$, $x = 34$.
Want to 2-factor (so all powers of 2 and 3)
$(34 + 0)^2 \pmod{1147}$
$\vdots \quad \vdots \quad \vdots$
$(34 + 10)^2 \pmod{1147}$
For the Sieve of E when we wanted to divide by p we looked at every pth element. Is there an analog here?

The Quadratic Sieve: Example

$N = 1147$, $B = 2$, $M = 10$, $x = 34$.
Want to 2-factor (so all powers of 2 and 3)

$$\begin{array}{l} (34+0)^2 \pmod{1147} \\ \vdots \qquad \vdots \qquad \vdots \\ (34+10)^2 \pmod{1147} \end{array}$$

For the Sieve of E when we wanted to divide by p we looked at every pth element. Is there an analog here?

For which $0 \le i \le 10$ does 2 divide $(34+i)^2 \pmod{1147}$?

The Quadratic Sieve: Example

$N = 1147$, $B = 2$, $M = 10$, $x = 34$.
Want to 2-factor (so all powers of 2 and 3)
$(34 + 0)^2 \pmod{1147}$

$$\vdots \quad \vdots \quad \vdots$$

$(34 + 10)^2 \pmod{1147}$
For the Sieve of E when we wanted to divide by p we looked at every pth element. Is there an analog here?

For which $0 \le i \le 10$ does 2 divide $(34 + i)^2 \pmod{1147}$?
Next Slide

The Quadratic Sieve: Example of Dividing by 2

Need to know the set of $0 \leq i \leq 10$ such that 2 divides

$$((34+i)^2 \pmod{1147}).$$

The Quadratic Sieve: Example of Dividing by 2

Need to know the set of $0 \le i \le 10$ such that 2 divides

$$((34+i)^2 \pmod{1147}).$$

What is $(34+i)^2 \pmod{1147}$?

The Quadratic Sieve: Example of Dividing by 2

Need to know the set of $0 \le i \le 10$ such that 2 divides

$$((34+i)^2 \pmod{1147}).$$

What is $(34+i)^2 \pmod{1147}$? Since $0 \le i \le 10$,

The Quadratic Sieve: Example of Dividing by 2

Need to know the set of $0 \le i \le 10$ such that 2 divides

$$((34+i)^2 \pmod{1147}).$$

What is $(34+i)^2 \pmod{1147}$? Since $0 \le i \le 10$,

$$(34+0)^2 \le (34+i)^2 \le (34+10)^2$$

The Quadratic Sieve: Example of Dividing by 2

Need to know the set of $0 \le i \le 10$ such that 2 divides

$$((34+i)^2 \pmod{1147}).$$

What is $(34+i)^2 \pmod{1147}$? Since $0 \le i \le 10$,

$$(34+0)^2 \le (34+i)^2 \le (34+10)^2$$

$$1156 \le (34+i)^2 \le 1936$$

The Quadratic Sieve: Example of Dividing by 2

Need to know the set of $0 \le i \le 10$ such that 2 divides

$$((34+i)^2 \pmod{1147}).$$

What is $(34+i)^2 \pmod{1147}$? Since $0 \le i \le 10$,

$$(34+0)^2 \le (34+i)^2 \le (34+10)^2$$

$$1156 \le (34+i)^2 \le 1936$$

$$1147 + 9 \le (34+i)^2 \le 1147 + 789.$$

So $(34+i)^2 \pmod{1147} = (34+i)^2 - 1147$.

The Quadratic Sieve: Example of Dividing by 2

Need to know the set of $0 \leq i \leq 10$ such that 2 divides

$$((34+i)^2 \pmod{1147}).$$

What is $(34+i)^2 \pmod{1147}$? Since $0 \leq i \leq 10$,

$$(34+0)^2 \leq (34+i)^2 \leq (34+10)^2$$

$$1156 \leq (34+i)^2 \leq 1936$$

$$1147+9 \leq (34+i)^2 \leq 1147+789.$$

So $(34+i)^2 \pmod{1147} = (34+i)^2 - 1147$.

Our question is, for which i is

$$(34+i)^2 - 1147 \equiv 0 \pmod{2}.$$

The Quadratic Sieve: Example of Dividing by 2, cont

Need to know the set of $0 \leq i \leq 10$ such that 2 divides

$$((34 + i)^2 \pmod{1147}).$$

The Quadratic Sieve: Example of Dividing by 2, cont

Need to know the set of $0 \leq i \leq 10$ such that 2 divides

$$((34+i)^2 \pmod{1147}).$$

We know that

$$(34+i)^2 \pmod{1147} = (34+i)^2 - 1147.$$

The Quadratic Sieve: Example of Dividing by 2, cont

Need to know the set of $0 \leq i \leq 10$ such that 2 divides

$$((34+i)^2 \pmod{1147}).$$

We know that

$$(34+i)^2 \pmod{1147} = (34+i)^2 - 1147.$$

Our question is, for which i is

$$(34+i)^2 - 1147 \equiv 0 \pmod{2}$$

The Quadratic Sieve: Example of Dividing by 2, cont

Need to know the set of $0 \leq i \leq 10$ such that 2 divides

$$((34+i)^2 \pmod{1147}).$$

We know that

$$(34+i)^2 \pmod{1147} = (34+i)^2 - 1147.$$

Our question is, for which i is

$$(34+i)^2 - 1147 \equiv 0 \pmod{2}$$

$$i^2 - 1 \equiv 0 \pmod{2}$$

The Quadratic Sieve: Example of Dividing by 2, cont

Need to know the set of $0 \le i \le 10$ such that 2 divides

$$((34+i)^2 \pmod{1147}).$$

We know that

$$(34+i)^2 \pmod{1147} = (34+i)^2 - 1147.$$

Our question is, for which i is

$$(34+i)^2 - 1147 \equiv 0 \pmod{2}$$

$$i^2 - 1 \equiv 0 \pmod{2}$$

$$i \equiv 1 \pmod{2}.$$

Great!- just need to divide the y_i where $i \equiv 1 \pmod{2}$.

The Quadratic Sieve: Example of Dividing by 3

For which $0 \leq i \leq 10$ does 3 divide $(34 + i)^2 \pmod{1147}$?

The Quadratic Sieve: Example of Dividing by 3

For which $0 \leq i \leq 10$ does 3 divide $(34+i)^2 \pmod{1147}$?
We know that $(34+i)^2 \pmod{1147} = (34+i)^2 - 1147$.

Our question is, for which i is

$$(34+i)^2 - 1147 \equiv 0 \pmod{3}$$

$$(1+i)^2 - 1 \equiv 0 \pmod{3}$$

$$(i+1)^2 \equiv 1 \pmod{3}$$

$$i \equiv 0, 1 \pmod{3}.$$

Great!- just need to divide the y_i where $i \equiv 0, 1 \pmod{3}$.

The Quad Sieve: Example of Dividing by 5,7,11,13

$(34+i)^2 - 1147 \equiv 0 \pmod{5}$
$(4+i)^2 - 2 \equiv 0 \pmod{5}$
NO SOLUTIONS

$(34+i)^2 - 1147 \equiv 0 \pmod{7}$
$(6+i)^2 \equiv 1 \pmod{7}$
$i \equiv 0, 2 \pmod{7}$

$(34+i)^2 - 1147 \equiv 0 \pmod{11}$
$(1+i)^2 \equiv 3 \pmod{11}$
$i \equiv 4, 5 \pmod{11}$

$(34+i)^2 - 1147 \equiv 0 \pmod{13}$
$(8+i)^2 + 10 \equiv 0 \pmod{13}$
$i \equiv 1, 9 \pmod{13}$

The Quad Sieve: Example of Dividing by 17,19,23

$(34+i)^2 - 1147 \equiv 0 \pmod{17}$
$i^2 + 9 \equiv 0 \pmod{17}$
$i \equiv 5, 12 \pmod{17}$

$(34+i)^2 - 1147 \equiv 0 \pmod{19}$
$(15+i)^2 + 12 \equiv 0 \pmod{19}$
$i \equiv 8, 15 \pmod{19}$

$(34+i)^2 - 1147 \equiv 0 \pmod{23}$
$(11+i)^2 + 3 \equiv 0 \pmod{23}$
NO SOLUTIONS

The B-Factor Step Using Quad Sieve: Program

Problem Given N, B, M, x, want to B-factor

$$(x+0)^2 \pmod{N}$$

$$\vdots \qquad \vdots$$

$$(x+M)^2 \pmod{N}$$

The B-Factor Step Using Quad Sieve: Program

Problem Given N, B, M, x, want to B-factor

$$(x+0)^2 \pmod{N}$$

$$\vdots \qquad \vdots$$

$$(x+M)^2 \pmod{N}$$

Algorithm

As p goes through the first B primes.

Find $A \subseteq \{0, \ldots, p-1\}$: $i \in A$ iff $(x+i)^2 - N \equiv 0 \pmod{p}$

The B-Factor Step Using Quad Sieve: Program

Problem Given N, B, M, x, want to B-factor

$(x+0)^2 \pmod{N}$

$\vdots \qquad \vdots$

$(x+M)^2 \pmod{N}$

Algorithm

As p goes through the first B primes.

Find $A \subseteq \{0, \ldots, p-1\}$: $i \in A$ iff $(x+i)^2 - N \equiv 0 \pmod{p}$

for $a \in A$

for $k = 0$ to $\left\lceil \frac{M-a}{p} \right\rceil$

The B-Factor Step Using Quad Sieve: Program

Problem Given N, B, M, x, want to B-factor

$(x+0)^2 \pmod N$

$\vdots \qquad \vdots$

$(x+M)^2 \pmod N$

Algorithm

As p goes through the first B primes.

Find $A \subseteq \{0, \ldots, p-1\}$: $i \in A$ iff $(x+i)^2 - N \equiv 0 \pmod p$

for $a \in A$

for $k = 0$ to $\left\lceil \frac{M-a}{p} \right\rceil$

divide $(x+pk+a)^2$ by p (and then p again...)

How Much Time?

Algorithm

As p goes through the first B primes.

Find $A \subseteq \{0, \ldots, p-1\}$: $i \in A$ iff $(x+i)^2 - N \equiv 0 \pmod{p}$

How Much Time?

Algorithm

As p goes through the first B primes.

  Find $A \subseteq \{0, \ldots, p-1\}$: $i \in A$ iff $(x+i)^2 - N \equiv 0 \pmod{p}$

  for $a \in A$

    for $k = 0$ to $\left\lceil \frac{M-a}{p} \right\rceil$

Time $\leq \sum_{p \leq B} (\lg p + 2\frac{M-1}{p}) = \sum_{p \leq B} \lg p + 2M \sum_{p \leq B} \frac{1}{p}$.

$$= (\sum_{p \leq B} \lg p) + 2M \ln\ln(B) \leq B\ln(B) + 2M\ln(\ln(B)).$$

How Much Time?

Algorithm

As p goes through the first B primes.

Find $A \subseteq \{0, \ldots, p-1\}$: $i \in A$ iff $(x+i)^2 - N \equiv 0 \pmod{p}$

for $a \in A$

for $k = 0$ to $\left\lceil \frac{M-a}{p} \right\rceil$

Time $\leq \sum_{p \leq B} (\lg p + 2\frac{M-1}{p}) = \sum_{p \leq B} \lg p + 2M \sum_{p \leq B} \frac{1}{p}$.

$$= (\sum_{p \leq B} \lg p) + 2M \ln\ln(B) \leq B\ln(B) + 2M\ln(\ln(B)).$$

The inequality $\sum_{p \leq B} \lg p \leq B\ln(B)$ requires some hard math.
The sum is called **Chebyshev's Function** .

Names of Sieves

1. The **Sieve of E** is the Sieve that, given N, finds all of the primes $\leq N$. We may also use the name for finding all primes between N_1 and N_2.
2. The ***B*-Factoring Sieve of E** is the Sieve that, given N, tries to B-factors all of the numbers from 2 to N. We may also use the name for B-factoring all numbers between N_1 and N_2.
3. The **Quadratic Sieve** is from the last slide. Given N, B, M, x it tries to B-factor $(x+0)^2 \pmod{N}$, ..., $(x+M)^2 \pmod{N}$. Note that it is quite fast.

Quad Sieve Alg: Second Attempt, Algorithm

Given N let $x = \left\lceil \sqrt{N} \right\rceil$. All $\equiv$ are mod N. B, M are params.

B-factor $(x+0)^2 \pmod{N}, \ldots, (x+M)^2 \pmod{N}$ by Quad S.

Let $I \subseteq \{0, \ldots, M\}$ so that $(\forall i \in I)$, y_i is B-factored. Find $J \subseteq I$ such that $\sum_{i \in J} \vec{v}_i = \vec{0}$. Hence $\prod_{i \in J} y_i$ has all even exponents, so there exists Y

$$\prod_{i \in J} y_i = Y^2$$

$$\left(\prod_{i \in J} (x+i)\right)^2 \equiv \prod_{i \in J} y_i = Y^2 \pmod{N}$$

Let $X = \prod_{i \in J}(x+i) \pmod{N}$ and $Y = \prod_{i=1}^{B} q_i^{e_i} \pmod{N}$.

$$X^2 - Y^2 \equiv 0 \pmod{N}.$$

$\mathrm{GCD}(X - Y, N)$, $\mathrm{GCD}(X + Y, N)$ should yield factors.

Analysis of Quadratic Sieve Factoring Algorithm

Time to B-factor:

$$2B + 2M\ln(\ln(B)).$$

Time to find J: B^3.

Total Time:

$$2B + 2M\ln(\ln(B)) + B^3$$

Intuitive but not rigorous arguments yield run time

$$e^{\sqrt{\ln N \ln\ln N}} \sim e^{\sqrt{8\ln N}} \sim e^{2.8\sqrt{\ln N}}$$

Speed Up One

Recall:
$(34 + i)^2 - 1147 \equiv 0 \pmod{23}$
$(11 + i)^2 + 3 \equiv 0 \pmod{23}$
NO SOLUTIONS

Speed Up One

Recall:
$(34 + i)^2 - 1147 \equiv 0 \pmod{23}$
$(11 + i)^2 + 3 \equiv 0 \pmod{23}$
NO SOLUTIONS

If there is a prime p such that $z^2 \equiv 1147 \pmod{p}$ has NO SOLUTION then we should not ever consider it.

Speed Up One

Recall:
$(34+i)^2 - 1147 \equiv 0 \pmod{23}$
$(11+i)^2 + 3 \equiv 0 \pmod{23}$
NO SOLUTIONS

If there is a prime p such that $z^2 \equiv 1147 \pmod{p}$ has NO SOLUTION then we should not ever consider it.

There is a fast test to determine just if $z^2 \equiv 1147 \pmod{p}$ has a solution (and more generally $z^2 \equiv N \pmod{p}$). So can eliminate some primes $p \leq B$ before you start.

Speed Up Two

Recall:
We started with $x = \left\lceil \sqrt{N} \right\rceil$ and did $(x+i)^2$ for $0 \le i \le M$.

Speed Up Two

Recall:
We started with $x = \left\lceil \sqrt{N} \right\rceil$ and did $(x+i)^2$ for $0 \le i \le M$.

We can also (with some care) use $(x+i)^2$ when $i \le 0$.

Advantage Smaller numbers more likely to be B-fact.

Speed Up Three

Recall:
$(34+i)^2 - 1147 \equiv 0 \pmod{19}$
$(15+i)^2 + 12 \equiv 0 \pmod{19}$
$i \equiv 8, 15 \pmod{19}$

Speed Up Three

Recall:
$(34+i)^2 - 1147 \equiv 0 \pmod{19}$
$(15+i)^2 + 12 \equiv 0 \pmod{19}$
$i \equiv 8, 15 \pmod{19}$

We can have one more variable:
$(34j+i)^2 - 1147 \equiv 0 \pmod{19}$
$(15j+i)^2 + 12 \equiv 0 \pmod{19}$
$15j + i \equiv 8, 15 \pmod{19}$
Many values of (i, j) work, hence we find the set of y's that product to a square faster.

Speed Up Four—Use some primes $> B$

1. Look at all of the non B-factored numbers. For each one test if what is left is prime. Let P_1 be the set of all of those primes..
2. Look at all of the non B-factored numbers. For each of them try a factoring algorithm (e.g, Pollards rho) for a limited amount of time. Let P_2 be the set of primes you come across.
3. Do Q. Sieve on all of the non B-factored numbers using the primes in $P_1 \cup P_2$.

This will increase the number of B-factored numbers.

Speed Up Five—Avoid Division

For this slide lg means $\lceil \lg \rceil$ which is very fast on a computer.

Using Divisions Primes $q_1, \ldots, q_m < B$ divide x. Divide x by all the q_i. Also q_i^2, q_i^3, etc until does not work. When you are done you've B-factored the number or not.

Speed Up Five—Avoid Division

For this slide lg means $\lceil \lg \rceil$ which is very fast on a computer.

Using Divisions Primes $q_1, \ldots, q_m < B$ divide x. Divide x by all the q_i. Also q_i^2, q_i^3, etc until does not work. When you are done you've B-factored the number or not.

Using Subtraction Primes $q_1, \ldots, q_m < B$ divide x. Do

$$d = \lg(x) - \lg(q_1) - \lg(q_2) - \cdots - \lg(q_m)$$

Speed Up Five—Avoid Division

For this slide lg means $\lceil \lg \rceil$ which is very fast on a computer.

Using Divisions Primes $q_1, \ldots, q_m < B$ divide x. Divide x by all the q_i. Also q_i^2, q_i^3, etc until does not work. When you are done you've B-factored the number or not.

Using Subtraction Primes $q_1, \ldots, q_m < B$ divide x. Do

$$d = \lg(x) - \lg(q_1) - \lg(q_2) - \cdots - \lg(q_m)$$

If $d \sim 0$ then we think x IS B-fact, so B-factor x.

If far from 0 then DO NOT DIVIDE!

Speed Up Five—Avoid Division, Why Works

Why Does This Work? If $x = q_1 q_2 q_3$ then

$$\lg(x) = \lg(q_1) + \lg(q_2) + \lg(q_3)$$

$$\lg(x) - \lg(q_1) - \lg(q_2) - \lg(q_3) = 0$$

Speed Up Five—Avoid Division, Why Works

Why Does This Work? If $x = q_1 q_2 q_3$ then

$$\lg(x) = \lg(q_1) + \lg(q_2) + \lg(q_3)$$

$$\lg(x) - \lg(q_1) - \lg(q_2) - \lg(q_3) = 0$$

So why not insist that

$$\lg(x) - \lg(q_1) - \lg(q_2) - \cdots - \lg(q_m) = 0$$

1. Using $\lceil \lg \rceil$ may introduce approximations so you don't get 0.
2. If $x = q_1^2 q_2 q_3$ then

$$\lg(x) = \lg(q_1^2) + \lg(q_2) + \lg(q_3) = 2\lg(q_1) + \lg(q_2) + \lg(q_3)$$

$$\lg(x) - \lg(q_1) + \lg(q_2) + \lg(q_3) = \lg(q_1) \neq 0$$

3. We need to define **small** carefully. Will still err.

Speed Up Five—Avoid Division, Why Fast

Why is this fast?

1. Subtraction is much faster than division.
2. Most numbers are **not** B-fact, so don't do divisions that won't help.

Speed Up Five—Avoid Division, Example One

$B = 7$ so we are looking at $2, 3, 5, 7, 11, 13, 17$. Small is ≤ 10.

Speed Up Five—Avoid Division, Example One

$B = 7$ so we are looking at $2, 3, 5, 7, 11, 13, 17$. Small is ≤ 10.

108290 7-fact? We find that 2,5,7,13,17 all divide it.

Speed Up Five—Avoid Division, Example One

$B = 7$ so we are looking at $2, 3, 5, 7, 11, 13, 17$. Small is ≤ 10.

108290 7-fact? We find that 2,5,7,13,17 all divide it.

$$\lg(108290) - \lg(2) - \lg(5) - \lg(7) - \lg(13) - \lg(17) = 4 \leq 10$$

Speed Up Five—Avoid Division, Example One

$B = 7$ so we are looking at $2, 3, 5, 7, 11, 13, 17$. Small is ≤ 10.

108290 7-fact? We find that 2,5,7,13,17 all divide it.

$$\lg(108290) - \lg(2) - \lg(5) - \lg(7) - \lg(13) - \lg(17) = 4 \leq 10$$

So we think 108290 IS 7-fact. Is this correct? Yes:

Speed Up Five—Avoid Division, Example One

$B = 7$ so we are looking at $2, 3, 5, 7, 11, 13, 17$. Small is ≤ 10.

108290 7-fact? We find that 2,5,7,13,17 all divide it.

$$\lg(108290) - \lg(2) - \lg(5) - \lg(7) - \lg(13) - \lg(17) = 4 \leq 10$$

So we think 108290 IS 7-fact. Is this correct? Yes:

$$108290 = 2 \times 5 \times 7^2 \times 13 \times 17$$

Speed Up Five—Avoid Division, Example Two

Is 78975897 7-fact? We find that 3,7,11,13,17 all divide it.

Speed Up Five—Avoid Division, Example Two

Is 78975897 7-fact? We find that 3,7,11,13,17 all divide it.

$$\lg(78975897) - \lg(3) - \lg(7) - \lg(11) - \lg(13) - \lg(17) = 11 > 10$$

Speed Up Five—Avoid Division, Example Two

Is 78975897 7-fact? We find that 3,7,11,13,17 all divide it.

$$\lg(78975897) - \lg(3) - \lg(7) - \lg(11) - \lg(13) - \lg(17) = 11 > 10$$

So we think 78975897 is NOT 7-fact. Is this correct? No!

$$78975897 = 3 \times 7^2 \times 11 \times 13^2 \times 17^4.$$

Speed Up Five—Avoid Division, Example Two

Is 78975897 7-fact? We find that 3,7,11,13,17 all divide it.

$$\lg(78975897) - \lg(3) - \lg(7) - \lg(11) - \lg(13) - \lg(17) = 11 > 10$$

So we think 78975897 is NOT 7-fact. Is this correct? No!

$$78975897 = 3 \times 7^2 \times 11 \times 13^2 \times 17^4.$$

Cautionary Note

$78975897 = 3 \times 7^2 \times 11 \times 13^2 \times 17^4$. was thought to NOT be 7-fact. Erred because primes had large exponents. The large exponents made

$$\lg(78975897)$$

LARGER than

$$\lg(3) + \lg(7) + \lg(11) + \lg(13) + \lg(17) + 10$$

Speed Up Five—Avoid Division, Examples Three

Is 9699690 7-fact? We find that 2,3,5,7,11,13,17 all divide it.

Speed Up Five—Avoid Division, Examples Three

Is 9699690 7-fact? We find that 2,3,5,7,11,13,17 all divide it.

$$\lg(9699690) - \lg(2) - \lg(3) - \lg(5) - \lg(7) - \lg(11) - \lg(13) - \lg(17) = 1 \leq 10$$

Speed Up Five—Avoid Division, Examples Three

Is 9699690 7-fact? We find that 2,3,5,7,11,13,17 all divide it.

$$\lg(9699690) - \lg(2) - \lg(3) - \lg(5) - \lg(7) - \lg(11) - \lg(13) - \lg(17) = 1 \leq 10$$

So we think 9699690 is 7-fact. Is this correct? No!

$$\lg(9699690) - \lg(2) - \lg(3) - \lg(5) - \lg(7) - \lg(11) - \lg(13) - \lg(17) = 1 \leq 10$$

Cautionary Note $78975897 = 2 \times 3 \times 5 \times 7 \times 11 \times 13 \times 17 \times 19.$ was thought to NOT be 7-fact. Erred because it had low exponents and only one a small prime over B.

Speed Up Five—Avoid Division, Examples Three

Is 9699690 7-fact? We find that 2,3,5,7,11,13,17 all divide it.

$$\lg(9699690) - \lg(2) - \lg(3) - \lg(5) - \lg(7) - \lg(11) - \lg(13) - \lg(17) = 1 \leq 10$$

So we think 9699690 is 7-fact. Is this correct? No!

$$\lg(9699690) - \lg(2) - \lg(3) - \lg(5) - \lg(7) - \lg(11) - \lg(13) - \lg(17) = 1 \leq 10$$

Cautionary Note $78975897 = 2 \times 3 \times 5 \times 7 \times 11 \times 13 \times 17 \times 19$. was thought to NOT be 7-fact. Erred because it had low exponents and only one a small prime over B.
Lemon to Lemonade Not B-fact, but still useful.

Speed Up Five-extra—Avoid Division, One More Trick

We are just **approximating** if

$$\lg x - \lg(q_1) - \cdots - \lg(q_m)$$

is small.

Speed Up Five-extra—Avoid Division, One More Trick

We are just **approximating** if

$$\lg x - \lg(q_1) - \cdots - \lg(q_m)$$

is small.

$\lg 2$, $\lg 3$, $\lg 5$ are so tiny, don't bother with those.

Speed Up Five-extra—Avoid Division, One More Trick

We are just **approximating** if

$$\lg x - \lg(q_1) - \cdots - \lg(q_m)$$

is small.

$\lg 2$, $\lg 3$, $\lg 5$ are so tiny, don't bother with those.

If $B = 7$ then use:

$$2^3, 3^2, 5^2, 7, 11, 13, 17, 19$$

Speed Up Six

The Gaussian Elimination is over mod 2 and is for a sparse matrix (most of the entries are 0).

There are special purpose algorithms for this.

1. Can be done in $O(B^{2+\epsilon})$ steps rather than $O(B^3)$.
2. Can't store the entire matrix—too big.

Speed Up Seven

(This is a paragraph from a blog post about Quad Sieve `https://blogs.msdn.microsoft.com/devdev/2006/06/19/factoring-large-numbers-with-quadratic-sieve/`)

Is z B-fact? There is a light for each $p \leq B$ whose intensity is proportional to the $\lg p$. Each light turns on just two times every p cycles, corresponding to the two square roots of N mod p. A sensor senses the combined intensity of all the lights together, and if this is close enough to the $\lg z$ then z is a B-fact number candidate. Can do in parallel.

The Number Field Sieve

The Quad Sieve had run time:

$$e^{(\ln N \ln \ln N)^{1/2}} \sim e^{2.8(\ln N)^{1/2}}$$

The Number Field Sieve

The Quad Sieve had run time:

$$e^{(\ln N \ln\ln N)^{1/2}} \sim e^{2.8(\ln N)^{1/2}}$$

The Number Field Sieve which uses some of the same ideas has run time:

$$e^{1.9(\ln N)^{1/3}(\ln\ln N)^{2/3}} \sim e^{14(\ln N)^{1/3}}$$

Compare Run Times

Alg	Run Time as $N^{a/L^{\delta}}$	Run Time in terms of L
Naive	$N^{1/2}$	$2^{L/2}$
Pollard Rho	$N^{1/4}$	$2^{L/4}$
Linear Sieve	$N^{3.9/L^{1/2}}$	$2^{1.95L^{1/2}}$
Quad Sieve	$N^{2.8/L^{1/2}}$	$2^{1.4L^{1/2}}$
N.F. Sieve	$N^{14/L^{2/3}}$	$2^{20L^{1/3}}$

1. Times are more conjectured than proven.
2. Quad S. is better than Linear Sieve by **only** a constant in the exponent. Made a big difference IRL.
3. Quad Sieve is better than Pollard-Rho at about 10^{50}.

Relevance for RSA

Relevance for RSA

1. Carl Pomerance devised the Quad S. algorithm in 1982.

Relevance for RSA

1. Carl Pomerance devised the Quad S. algorithm in 1982.
2. People did not think it would work that well; however, he had friends at Sandia Labs who tried it out. Just for fun.

Relevance for RSA

1. Carl Pomerance devised the Quad S. algorithm in 1982.
2. People did not think it would work that well; however, he had friends at Sandia Labs who tried it out. Just for fun.
3. At the same time another group at Sandia Labs was working on a serious RSA project that would use 100-digit N.

Relevance for RSA

1. Carl Pomerance devised the Quad S. algorithm in 1982.
2. People did not think it would work that well; however, he had friends at Sandia Labs who tried it out. Just for fun.
3. At the same time another group at Sandia Labs was working on a serious RSA project that would use 100-digit N.
4. Quad Sieve could factor 100-digit numbers, so the RSA project had to be scrapped.

The Future of Factoring

I paraphrase **The Joy of Factoring** by Wagstaff:
The best factoring algorithms have time complexity of the form

$$e^{c(\ln N)^t(\ln\ln N)^{1-t}}$$

**with Q.Sieve using $t = \frac{1}{2}$ and N.F.Sieve using $t = \frac{1}{3}$.
Moreover, any method that uses B-factoring must take this long.**

The Future of Factoring

I paraphrase **The Joy of Factoring** by Wagstaff:
The best factoring algorithms have time complexity of the form

$$e^{c(\ln N)^t(\ln\ln N)^{1-t}}$$

with Q.Sieve using $t = \frac{1}{2}$ and N.F.Sieve using $t = \frac{1}{3}$. Moreover, any method that uses B-factoring must take this long.

- No progress since N.F.Sieve in 1988.

The Future of Factoring

I paraphrase **The Joy of Factoring** by Wagstaff:
The best factoring algorithms have time complexity of the form

$$e^{c(\ln N)^t(\ln\ln N)^{1-t}}$$

with Q.Sieve using $t = \frac{1}{2}$ and N.F.Sieve using $t = \frac{1}{3}$.
Moreover, any method that uses B-factoring must take this long.

- No progress since N.F.Sieve in 1988.
- My opinion: $e^{c(\ln N)^t(\ln\ln N)^{1-t}}$ is the best you can do ever, though t can be improved.

The Future of Factoring

I paraphrase **The Joy of Factoring** by Wagstaff:
The best factoring algorithms have time complexity of the form

$$e^{c(\ln N)^t(\ln\ln N)^{1-t}}$$

with Q.Sieve using $t=\frac{1}{2}$ and N.F.Sieve using $t=\frac{1}{3}$.
Moreover, any method that uses B-factoring must take this long.

- No progress since N.F.Sieve in 1988.
- My opinion: $e^{c(\ln N)^t(\ln\ln N)^{1-t}}$ is the best you can do ever, though t can be improved.
- Why hasn't t been improved? Wagstaff told me:

The Future of Factoring

I paraphrase **The Joy of Factoring** by Wagstaff:
The best factoring algorithms have time complexity of the form

$$e^{c(\ln N)^t(\ln\ln N)^{1-t}}$$

with Q.Sieve using $t = \frac{1}{2}$ and N.F.Sieve using $t = \frac{1}{3}$.
Moreover, any method that uses B-factoring must take this long.

- No progress since N.F.Sieve in 1988.
- My opinion: $e^{c(\ln N)^t(\ln\ln N)^{1-t}}$ is the best you can do ever, though t can be improved.
- Why hasn't t been improved? Wagstaff told me:
  - We've run out of parameters to optimize.

The Future of Factoring

I paraphrase **The Joy of Factoring** by Wagstaff:
The best factoring algorithms have time complexity of the form

$$e^{c(\ln N)^t(\ln\ln N)^{1-t}}$$

with Q.Sieve using $t = \frac{1}{2}$ and N.F.Sieve using $t = \frac{1}{3}$. Moreover, any method that uses B-factoring must take this long.

- No progress since N.F.Sieve in 1988.
- My opinion: $e^{c(\ln N)^t(\ln\ln N)^{1-t}}$ is the best you can do ever, though t can be improved.
- Why hasn't t been improved? Wagstaff told me:
  - We've run out of parameters to optimize.
  - Brandon, Solomon, Mark, and Ivan haven't worked on it yet.

BILL STOP RECORDING