Faster factorization into coprimes

D. J. Bernstein

Thanks to:
University of Illinois at Chicago
NSF DMS–0140542
Alfred P. Sloan Foundation
Problem: Convert
\[ x \equiv a \pmod{299}, \]
\[ x \equiv b \pmod{799} \]
into a single congruence.

Solution:
\[ x \equiv 799 \cdot 180 \cdot a - 299 \cdot 481 \cdot b \pmod{299 \cdot 799}. \]

Underlying computation, by Euclid’s algorithm:
\[ 799 \cdot 180 - 299 \cdot 481 = 1. \]
Problem: Convert
\[ x \equiv a \pmod{299}, \]
\[ x \equiv b \pmod{793} \]
into a single congruence.

Much more difficult.
Can’t write 1 as 793u + 299v; 793 and 299 aren’t coprime.

Euclid’s algorithm discovers
\[ \gcd\{299, 793\} = 13: \text{ specifically,} \]
\[ 13 = 793 \cdot 20 - 299 \cdot 53, \]
\[ 299 = 13 \cdot 23, \quad 793 = 13 \cdot 61. \]
gcd\{13, 23\} = 1. Thus
\[x \equiv a \pmod{299} \iff x \equiv a \pmod{13}, \]
\[x \equiv a \pmod{23}. \]

gcd\{13, 61\} = 1. Thus
\[x \equiv b \pmod{793} \iff x \equiv b \pmod{13}, \]
\[x \equiv b \pmod{61}. \]

Underlying computations:
\[23 \cdot 4 - 13 \cdot 7 = 1; \]
\[61 \cdot 3 - 13 \cdot 14 = 1. \]
Assuming $a \equiv b \pmod{13}$:

\begin{align*}
x &\equiv a \pmod{299}, \\
x &\equiv b \pmod{793} \iff \\
x &\equiv a \pmod{13}, \\
x &\equiv a \pmod{23}, \\
x &\equiv b \pmod{61} \iff \\
x &\equiv -1 \cdot 23 \cdot 61 \cdot a \\
&\quad + 13 \cdot 21 \cdot 61 \cdot a \\
&\quad - 13 \cdot 23 \cdot 51 \cdot b \\
&\pmod{13 \cdot 23 \cdot 61}.
\end{align*}
Problem: Convert

\[ x \equiv a \pmod{103816603}, \]
\[ x \equiv b \pmod{22649627} \]
into a single congruence.

\[
\gcd\{103816603, 22649627\} = 187; \]
\[ 103816603 = 187 \cdot 555169; \]
\[ 22649627 = 187 \cdot 121121. \]

Now encounter another difficulty:

187, 555169 aren’t coprime;
congruence mod 103816603
is not equivalent to
separate congruences
mod 187 and mod 555169.
Continue computing gcads and exact quotients:

\[
\gcd\{555169, 187\} = 17; \\
555169/17 = 32657; 187/17 = 11; \\
32657/17 = 1921; 1921/17 = 113; \\
121121/11 = 11011; \\
11011/11 = 1001; 1001/11 = 91.
\]

11, 17, 91, 113 are coprime;

\[
103816603 = 11 \cdot 17^4 \cdot 113; \\
22649627 = 11^4 \cdot 17 \cdot 91.
\]

\[x \equiv \cdots \pmod{11^4 \cdot 17^4 \cdot 91 \cdot 113}.\]
For any set $S \subseteq \{1, 2, 3, \ldots\}$: The natural coprime base for $S$, written cb $S$, is the unique $P \subseteq \{2, 3, \ldots\}$ such that

- each element of $P$ can be obtained from $S \cup \{1\}$ via product, exact quotient, gcd;
- $P$ is coprime: $\gcd\{a, b\} = 1$ for all distinct $a, b \in P$; and
- each element of $S$ can be obtained from $P \cup \{1\}$ via product.

e.g. $\text{cb}\{103816603, 22649627\}$

$= \{11, 17, 91, 113\}$. 
Obvious algorithm to compute $\text{cb} \ S$ and factor $S$ over $\text{cb} \ S$: time $O(n^3)$ for $n$ input bits. (frequently reinvented)

More careful algorithm, avoiding pointless gcd computations: $O(n^2)$. (1990 Bach Driscoll Shallit)

Can do much better for large $n$: $n^{1+o(1)}$; more precisely, $n(\lg n)^O(1)$. (1995 Bernstein)

New algorithm: $n(\lg n)^{4+o(1)}$. (2004 Bernstein)
This line of work has also led to \( n(\lg n)^{3+o(1)} \), and sometimes \( n(\lg n)^{2+o(1)} \), algorithms for various constrained examples of factoring into coprimes.

Unexpected applications to proving primality, detecting perfect powers, factoring into primes, et al.
Can apply same algorithms in more generality: e.g., replace integers with polynomials.

Typical application:
Consider a squarefree $g \in (\mathbb{Z}/2)[x]$. What are $g$’s irreducible divisors?

One answer: Find basis $h_1, h_2, \ldots$ for $\{ h \in (\mathbb{Z}/2)[x] : (gh)' = h^2 \}$ as a vector space over $\mathbb{Z}/2$. Then $\text{cb}\{g, h_1, h_2, \ldots \}$ contains all irreducible divisors of $g$.

(1993 Niederreiter, 1994 Göttfert)
Fast product, quotient, gcd

Given $a, b \in \mathbb{Z}$, can compute $ab$ in time $\leq n(\lg n)^{1+o(1)}$ where $n$ is number of input bits.

(1971 Pollard; independently 1971 Nicholson; independently 1971 Schönhage Strassen)

Also time $\leq n(\lg n)^{1+o(1)}$ where $n$ is number of input bits:

Given $a, b \in \mathbb{Z}$ with $b \neq 0$, compute $\lfloor a/b \rfloor$ and $a \mod b$.

(reduction to product: 1966 Cook)
Time $\leq n(\lg n)^{2+o(1)}$:
Given $a, b \in \mathbb{Z}$, compute $\gcd\{a, b\}$.

(1971 Schönhage; core idea: 1938 Lehmer; $n(\lg n)^{5+o(1)}$: 1971 Knuth)

Better time bound when $a$ is much larger than $b$:
$\leq n(\lg n)^{1+o(1)} + m(\lg m)^{2+o(1)}$

where $m$ is number of bits in $b$.

Idea: $\gcd\{b, a \mod b\}$.

For survey of these algorithms:
http://cr.yp.to/papers.html
#multapps
Modular squaring ad nauseam

Time \leq n(\lg n)^{2+o(1)}:

Given \(a, b \in \mathbb{Z}\) with \(a \neq 0\), compute \(\gcd\{a, b^\infty\}\).

Algorithm:

Compute \(b \mod a\),

\[b^2 \mod a = (b \mod a)^2 \mod a,\]

\[b^4 \mod a = (b^2 \mod a)^2 \mod a,\]

\[b^8 \mod a = (b^4 \mod a)^2 \mod a,\]

etc., until \(b^{2^k}\) with \(2^k \geq n\).

Then compute \(\gcd\{a, b^\infty\}\) as \(\gcd\left\{a, b^{2^k} \mod a\right\}\).
Factoring \(a, b\) into coprimes

Given \(a, b \in \mathbb{Z}, a \geq b \geq 2:\)
Compute \(a_0 = a, g_0 = \gcd\{a_0, b\}, a_1 = a_0 / g_0, g_1 = \gcd\{a_1, g_0^2\}, a_2 = a_1 / g_1, g_2 = \gcd\{a_2, g_1^2\},\) etc., stopping when \(g_k = 1.\)

How long does this take?

e.g. \(a = 2^{100} 3^{100}, b = 2^{137} 3^{13}:\)
\(a_0 = 2^{100} 3^{100}, g_0 = 2^{100} 3^{13},\)
\(a_1 = 3^{87}, g_1 = 3^{26},\)
\(a_2 = 3^{61}, g_2 = 3^{52},\)
\(a_3 = 3^9, g_3 = 3^9,\)
\(a_4 = 1, g_4 = 1.\)
Consider a prime $p$.
Define $e = \text{ord}_p a$: i.e., $p^e$ divides $a$ but $p^{e+1}$ doesn’t.
Define $f = \text{ord}_p b$.

<table>
<thead>
<tr>
<th>$e &gt;$</th>
<th>$f$</th>
<th>$3f$</th>
<th>$7f$</th>
<th>$15f$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$e \leq$</td>
<td>$f$</td>
<td>$3f$</td>
<td>$7f$</td>
<td>$15f$</td>
</tr>
<tr>
<td>$\text{ord}_p a_0$</td>
<td>$e$</td>
<td>$e$</td>
<td>$e$</td>
<td>$e$</td>
</tr>
<tr>
<td>$\text{ord}_p g_0$</td>
<td>$e$</td>
<td>$f$</td>
<td>$f$</td>
<td>$f$</td>
</tr>
<tr>
<td>$\text{ord}_p a_1$</td>
<td>$0$</td>
<td>$e - f$</td>
<td>$e - f$</td>
<td>$e - f$</td>
</tr>
<tr>
<td>$\text{ord}_p g_1$</td>
<td>$0$</td>
<td>$e - f$</td>
<td>$2f$</td>
<td>$2f$</td>
</tr>
<tr>
<td>$\text{ord}_p a_2$</td>
<td>$0$</td>
<td>$0$</td>
<td>$e - 3f$</td>
<td>$e - 3f$</td>
</tr>
<tr>
<td>$\text{ord}_p g_2$</td>
<td>$0$</td>
<td>$0$</td>
<td>$e - 3f$</td>
<td>$4f$</td>
</tr>
<tr>
<td>$\text{ord}_p a_3$</td>
<td>$0$</td>
<td>$0$</td>
<td>$0$</td>
<td>$e - 7f$</td>
</tr>
<tr>
<td>$\text{ord}_p g_3$</td>
<td>$0$</td>
<td>$0$</td>
<td>$0$</td>
<td>$e - 7f$</td>
</tr>
</tbody>
</table>
\[2^e \leq p^e \leq a < 2^n\] so \(e < n\).

Thus \(g_k = 1\) for \(k = \lfloor \log n \rfloor\).

Time to divide \(a_i\) by \(g_i\), square \(g_i\), and compute \(\gcd\{a_{i+1}, g_i^2\}\):

\[\leq n (\log n)^{1+o(1)} + m_i (\log m_i)^{2+o(1)}\]

where \(m_i\) is number of bits in \(g_i\).

\[a = a_k \prod g_i\] so \(\sum m_i \leq O(n)\).

Total time for all \(a_i, g_i\):

\[\leq n (\log n)^{2+o(1)}\].
Next step: Compute $b \mod g_1, b \mod g_2, \ldots$ using a **remainder tree** (1972 Fiduccia, 1972 Moenck Borodin):

$$b \mod g_1 g_2 g_3 g_4$$

$$\quad b \mod g_1 g_2 \quad \text{and} \quad b \mod g_3 g_4$$

$$\quad \quad b \mod g_2 \quad \quad b \mod g_4$$

$$\quad \quad b \mod g_1 \quad \quad b \mod g_3$$

Total time $\leq n(\log n)^{1+o(1)}$. 
Next step: Compute

\[ x_0 = \frac{g_0}{\gcd\{g_0, g_1^{\infty}\}} , \]
\[ x_1 = \frac{g_1}{\gcd\{g_1, g_2^{\infty}\}} , \]

etc.

Write \( m'_i = m_i + m_{i+1} \).

Time \( \leq \sum m'_i (\log m'_i)^2 + o(1) \)
\( \leq n (\log n)^2 + o(1) \).

E.g. \( a = 2^{100} 3^{100} , b = 2^{137} 3^{13} \):
\[ g_0 = 2^{100} 3^{13} , g_1 = 3^{26} , \]
\[ g_2 = 3^{52} , g_3 = 3^{9} , g_4 = 1 ; \]
\[ x_0 = 2^{100} , x_1 = 1 , x_2 = 1 , \]
\[ x_3 = 3^{9} . \]
Next step: Compute

\[ y_0 = \gcd\{b, x_0^\infty\}, \]
\[ y_1 = \gcd\{g_0, x_1^\infty\}, \]
\[ y_2 = \gcd\{\gcd\{b \mod g_1, g_1\}, x_2^\infty\}, \]
\[ y_3 = \gcd\{\gcd\{b \mod g_2, g_2\}, x_3^\infty\}, \]
\[ y_4 = \gcd\{\gcd\{b \mod g_3, g_3\}, x_4^\infty\}, \]

etc.

\[ \text{Time} \leq n(\lg n)^{2+o(1)}. \]

e.g. \( a = 2^{100}3^{100}, b = 2^{137}3^{13}. \)
\[ x_0 = 2^{100}, x_1 = 1, x_2 = 1, x_3 = 3^9; \]
\[ y_0 = 2^{137}, y_1 = 1, y_2 = 1, y_3 = 3^{13}. \]
Now cb\{a, b\} is disjoint union of
cb\{x_0, y_0/x_0\},
cb\{x_1, y_1\}, cb\{x_2, y_2\}, \ldots ,
\{a_k\} − \{1\}, \{b/gcd\{b, a^\infty\}\} − \{1\}.

E.g. cb\{2^{100}3^{100}, 2^{137}3^{13}\} =
cb\{2^{100}, 2^{37}\} \cup cb\{3^9, 3^{13}\}.

Recursion multiplies total time by a constant factor, since
product \(x_0(y_0/x_0)x_1y_1x_2y_2 \ldots \)
is at most \(ab/a^{1/3} \leq (ab)^{5/6}\).

Time \(\leq n(lg\ n)^{2+o(1)}\)
to compute cb\{a, b\}. 
Outline of the general case

Time $\leq (k + 1)n(\lg n)^{2+o(1)}$:
Given multiset $S$ and coprime set $P$ with $\#P \leq 2^k$, compute $\gcd\{s, p^\infty\}$ for each $s \in S$, each $p \in P$.

Time $\leq n(\lg n)^{2+o(1)}$:
Given $a$ and coprime set $Q$, compute $\mathsf{cb}(\{a\} \cup Q)$.

http://cr.yp.to/papers.html
#dcba2
Remaining constructions are the same as in 1995:

http://cr.yp.to/papers.html

Time $\leq n (\lg n)^{3+o(1)}$:
Given coprime $P$, coprime $Q$, compute $cb(P \cup Q)$.

Time $\leq n (\lg n)^{4+o(1)}$:
Given $S$, compute $cb S$.
Also handle factorizations.
Detecting multiplicative relations

Does $91^{1952681} 119^{1513335} 221^{634643}$ equal $1547^{1708632} 6898073^{439346}$?

Each side has logarithm

$\approx 19466590.674872$.

More generally:

What is kernel of $(a, b, c, d, e) \mapsto 91^a 119^b 221^c 1547^{-d} 6898073^{-e}$?
Factor into coprimes:
91 = 7 \cdot 13; 119 = 7 \cdot 17;
221 = 13 \cdot 17; 1547 = 7 \cdot 13 \cdot 17;
6898073 = 7^4 \cdot 13^2 \cdot 17.

\((a, b, c, d, e) \mapsto \frac{91^a 119^b 221^c 1547^{-d} 6898073^{-e}}{7^{a+b-d-4e} 13^{a+c-d-2e} 17^{b+c-d-e}}.\)

Kernel is generated by
\((1, 1, 1, 2, 0)\) and \((3, 2, 0, 1, 1)\).
Useful in modern “combination of congruence” algorithms to factor into primes, compute discrete logs, compute class groups, etc.

Discrete-log example:
Factor 9974, 1, 9975, 2, 9976, 3, . . . into coprimes and compute a kernel to combine the congruences

\[
\begin{align*}
9974/1 & \equiv 1 \pmod{9973}, \\
9975/2 & \equiv 1 \pmod{9973}, \\
9976/3 & \equiv 1 \pmod{9973}, \ldots \\
\end{align*}
\]

into \(2^{1515}/11^{243} \equiv 1 \pmod{9973}\).
Detecting perfect powers

Given integer $b$ with $1 < b < 2^n$. Want largest integer $k$ such that $b$ is a $k$th power.

Find integer $r_k$ within 0.9 of $b^{1/k}$ for $1 \leq k < n$.

Can check if $(r_k)^k = b$ for each $k$ in total time $n \exp(O(\sqrt{\lg n} \lg \lg n))$. (1995 Bernstein, using linear forms in logarithms)
Time $n(\lg n)^{O(1)}$ using fast factorization into coprimes:

Compute $P = \text{cb}\{r_1, r_2, \ldots\}$.

$b$ is a $k$th power if and only if $k$ divides $\operatorname{ord}_q b$ for each $q \in P$. Largest $k$ is $\gcd\{\operatorname{ord}_q b : q \in P\}$.

(1994 Lenstra Pila; 2004 Bernstein Lenstra Pila)