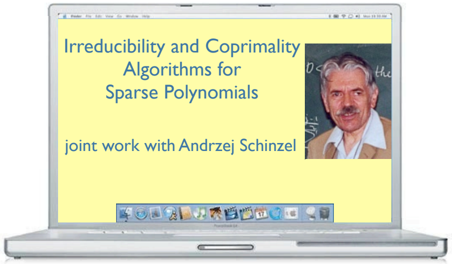

§ Introduction

Suppose we want to check the primality of

$$N = 2^{30402457} - 1.$$

How fast can we do this computation?

How fast can we expect to do it?

If the number of binary operations for a computation is bounded by a polynomial in the length of the input, then we say it can be done in polynomial time.

§ Introduction

Suppose we want to check the primality of

$$N = 2^{30402457} - 1.$$

How fast can we do this computation?

How fast can we expect to do it?

What is the length of the input?

The number N contains 30402457 bits.

Determining if N is prime in 30402457^2 steps would be good.

§ Introduction

Suppose we want to check the primality of

$$N = 2^{30402457} - 1.$$

How fast can we do this computation?

How fast can we expect to do it?

What is the length of the input?

To clarify, typing

```
2^30402457-1
```

takes 12 keystrokes.

But this is a talk about polynomials

$$f(x) \in \mathbb{Z}[x].$$

Suppose f has degree n, height $\le H$ and $\le r$ ~~non-zero~~ terms.

all terms are non-zero

maximum coefficient in absolute value

But this is a talk about polynomials

$$f(x) \in \mathbb{Z}[x].$$

Suppose f has degree n, height $\le H$ and $\le r$ non-zero terms.

Traditionally, $f(x)$ has $n+1$ coefficients and each coefficient can have "length" on the order of $\log H$ so that the total length of the input is of order $n \log H$. Actually, I should say $n\big(\log H + \log n\big)$.

But this is a talk about polynomials

$$f(x) \in \mathbb{Z}[x].$$

Suppose f has degree n, height $\le H$ and $\le r$ non-zero terms.

Lenstra, Lenstra and Lovasz showed that one can factor f in time that is polynomial in n and $\log H$.

But this is a talk about polynomials

$$f(x) \in \mathbb{Z}[x].$$

Suppose f has degree n, height $\le H$ and $\le r$ non-zero terms.

We might expect an algorithm exists that runs in time that is polynomial in $\log n$, r and $\log H$ except that the factors might well take time that is polynomial in n and $\log H$ to output.

Example: Factor

$$x^{101} + x^{77} - x^{76} - x^{13} + x^{12} - 1.$$

The answer is

$$\begin{aligned}(x-1)\big(&x^{100} + x^{99} + x^{98} + x^{97}\\ &+ x^{96} + x^{95} + x^{94} + x^{93} + x^{92}\\ &+ x^{91} + x^{90} + x^{89} + x^{88} + x^{87}\\ &+ x^{86} + x^{85} + x^{84} + x^{83} + x^{82}\\ &+ x^{81} + x^{80} + x^{79} + x^{78} + x^{77}\\ &+ 2x^{76} + x^{75} + x^{74} + x^{73} + x^{72}\\ &+ x^{71} + x^{70} + x^{69} + x^{68} + x^{67}\end{aligned}$$

$$\begin{aligned}&+ x^{66} + x^{65} + x^{64} + x^{63} + x^{62}\\ &+ x^{61} + x^{60} + x^{59} + x^{58} + x^{57}\\ &+ x^{56} + x^{55} + x^{54} + x^{53} + x^{52}\\ &+ x^{51} + x^{50} + x^{49} + x^{48} + x^{47}\\ &+ x^{46} + x^{45} + x^{44} + x^{43} + x^{42}\\ &+ x^{41} + x^{40} + x^{39} + x^{38} + x^{37}\\ &+ x^{36} + x^{35} + x^{34} + x^{33} + x^{32}\\ &+ x^{31} + x^{30} + x^{29} + x^{28} + x^{27}\\ &+ x^{26} + x^{25} + x^{24} + x^{23} + x^{22}\\ &+ x^{21} + x^{20} + x^{19} + x^{18} + x^{17}\end{aligned}$$

$$\begin{aligned}&+ x^{16} + x^{15} + x^{14} + x^{13} + x^{11}\\ &+ x^{10} + x^{9} + x^{8} + x^{7} + x^{6} + x^{5}\\ &+ x^{4} + x^{3} + x^{2} + x + 1\big)\end{aligned}$$

We might expect an algorithm exists that runs in time that is polynomial in $\log n$, r and $\log H$ except that the factors might well take time that is polynomial in n and $\log H$ to output.

But this is a talk about *irreducibility testing* of polynomials

$$f(x) \in \mathbb{Z}[x].$$

Here, it is more reasonable to expect an algorithm to run in time that is polynomial in $\log n$, r and $\log H$.

But we won't do that.

Thereom (A. Schinzel and M.F.): *There exist*

$$c_1 = c_1(H, r) \quad \textit{and} \quad c_2 = c_2(H, r)$$

and an algorithm that decides if a given nonreciprocal $f(x) \in \mathbb{Z}[x]$ of degree n, which has height $\le H$ and $\le r$ non-zero terms, is irreducible and that runs in time

$$O(c_1 (\log n)^{c_2}).$$

$f(x)$ is reciprocal means that if $f(\alpha) = 0$, then $\alpha \ne 0$ and $f(1/\alpha) = 0$

Thereom (A. Schinzel and M.F.): *There exist*

$$c_1 = c_1(H, r) \quad \textit{and} \quad c_2 = c_2(H, r)$$

and an algorithm that decides if a given nonreciprocal $f(x) \in \mathbb{Z}[x]$ of degree n, which has height $\le H$ and $\le r$ non-zero terms, is irreducible and that runs in time

$$O(c_1 (\log n)^{c_2}).$$

$f(x)$ is reciprocal means that

$$f(x) = \pm x^{\deg f} f(1/x)$$

Thereom (A. Schinzel and M.F.): *There exist*

$$c_1 = c_1(H, r) \quad \textit{and} \quad c_2 = c_2(H, r)$$

and an algorithm that decides if a given nonreciprocal $f(x) \in \mathbb{Z}[x]$ of degree n, which has height $\le H$ and $\le r$ non-zero terms, is irreducible and that runs in time

$$O(c_1 (\log n)^{c_2}).$$

$$\boxed{f(x) \ne \pm x^{\deg f} f(1/x)}$$

Thereom (A. Schinzel and M.F.): *There exist*

$$c_1 = c_1(H, r) \quad \textit{and} \quad c_2 = c_2(H, r)$$

and an algorithm that decides if a given nonreciprocal $f(x) \in \mathbb{Z}[x]$ of degree n, which has height $\le H$ and $\le r$ non-zero terms, is irreducible and that runs in time

$$O(c_1 (\log n)^{c_2}).$$

Remark: If the polynomial is reducible, then it is possible to determine a non-trivial factor in the same time but ...

- If f has a cyclotomic factor, then the algorithm will detect this and output an $m \in \mathbb{Z}^+$ with $\Phi_m(x)$ a factor.
- If f has no cyclotomic factor but has a reciprocal factor, then the algorithm will give an explicit reciprocal factor.
- Otherwise, the algorithm outputs the complete factorization of $f(x)$ into irreducible polynomials over $\mathbb{Q}$.

Comment: It is not even obvious that such output can be given in time that is less than polynomial in $\deg f$.

- If f has a cyclotomic factor, then the algorithm will detect this and output an $m \in \mathbb{Z}^+$ with $\Phi_m(x)$ a factor.
- If f has no cyclotomic factor but has a reciprocal factor, then the algorithm will give an explicit reciprocal factor.
- Otherwise, the algorithm outputs the complete factorization of $f(x)$ into irreducible polynomials over $\mathbb{Q}$.

The algorithm does these in the order listed.

Corollary: *If $f(x) \in \mathbb{Z}[x]$ is nonreciprocal and reducible, then $f(x)$ has a non-trivial factor in $\mathbb{Z}[x]$ which contains $\leq c(r, H)$ terms.*

Example: For almost any $a_j \in \mathbb{Z}$ with $|a_j| \leq 1000$ and any positive integers $e_1, \ldots, e_{100}$, if the polynomial

$$a_0 + a_1 x^{e_1} + a_2 x^{e_2} + \cdots + a_{100} x^{e_{100}}$$

is reducible over $\mathbb{Q}$, then it has a non-trivial factor with $\leq c$ terms.

- If f has a cyclotomic factor, then the algorithm will detect this and output an $m \in \mathbb{Z}^+$ with $\Phi_m(x)$ a factor.

Theorem (A. Schinzel and M.F., 2004): *There is an algorithm which determines if a given $f(x) \in \mathbb{Z}[x]$ of degree $n > 1$, which has height H and $r > 1$ non-zero terms, has a cyclotomic factor and that runs in time big-oh of*

$$c_1(H, r)(\log n)^{c_2(r)}$$

as r tends to infinity.

- If f has a cyclotomic factor, then the algorithm will detect this and output an $m \in \mathbb{Z}^+$ with $\Phi_m(x)$ a factor.

Lemma: *Let $f(x) \in \mathbb{Z}[x]$ have r non-zero terms. If $f(x)$ is divisible by a cyclotomic polynomial, then there is a positive integer m such that $\Phi_m(x) | f(x)$ and every prime factor of m is $\leq r$.*

- If f has a cyclotomic factor, then the algorithm will detect this and output an $m \in \mathbb{Z}^+$ with $\Phi_m(x)$ a factor.

The division algorithm for polynomials takes time that is polynomial in the degrees of the input polynomials.

$$x^{100} - x^{18} + 1 = (x^3 + x + 1)q(x) + r(x)$$

$q(x)$ has 96 terms

$$r(x) = 101010478x^2 - 19122919x - 60075671$$

- If f has a cyclotomic factor, then the algorithm will detect this and output an $m \in \mathbb{Z}^+$ with $\Phi_m(x)$ a factor.

The division algorithm for polynomials takes time that is polynomial in the degrees of the input polynomials.

So how does one check if $\Phi_m(x) | f(x)$?

If m is small, this is easy (reduce the exponents of $f(x)$ mod m and do the division).

- If f has a cyclotomic factor, then the algorithm will detect this and output an $m \in \mathbb{Z}^+$ with $\Phi_m(x)$ a factor.

$$x^{100} - x^{88} + 1 = (x^6 + x^3 + 1)q(x) + r(x)$$

$$x^{100} - x^{88} + 1 = (x^9 - 1)q_2(x) + r_2(x)$$

$$r(x) \equiv r_2(x) \pmod{x^6 + x^3 + 1}$$

$$r_2(x) \equiv x^{100} - x^{88} + 1 \pmod{x^9 - 1}$$

$$x^{100} \equiv x \pmod{x^9 - 1}$$

- If f has a cyclotomic factor, then the algorithm will detect this and output an $m \in \mathbb{Z}^+$ with $\Phi_m(x)$ a factor.

$$x^{100} - x^{88} + 1 = (x^6 + x^3 + 1)q(x) + r(x)$$

$$x^{100} - x^{88} + 1 = (x^9 - 1)q_2(x) + r_2(x)$$

$$r(x) \equiv r_2(x) \pmod{x^6 + x^3 + 1}$$

$$r_2(x) \equiv x^{100} - x^{88} + 1 \pmod{x^9 - 1}$$

$$r_2(x) \equiv -x^7 + x + 1 \pmod{x^6 + x^3 + 1}$$

- If f has a cyclotomic factor, then the algorithm will detect this and output an $m \in \mathbb{Z}^+$ with $\Phi_m(x)$ a factor.

$$x^{100} - x^{88} + 1 = (x^6 + x^3 + 1)q(x) + r(x)$$

$$x^{100} - x^{88} + 1 = (x^9 - 1)q_2(x) + r_2(x)$$

$$r(x) \equiv r_2(x) \pmod{x^6 + x^3 + 1}$$

$$r(x) \equiv -x^7 + x + 1 \pmod{x^6 + x^3 + 1}$$

$$r(x) \equiv x^4 + 2x + 1 \pmod{x^6 + x^3 + 1}$$

- If f has a cyclotomic factor, then the algorithm will detect this and output an $m \in \mathbb{Z}^+$ with $\Phi_m(x)$ a factor.

To check whether a fixed $\Phi_m(x)$ divides $f(x)$, check instead whether

$$(x^m - 1) \mid f(x) \cdot \prod_{\substack{d|m \\ d \neq m}} (x^d - 1).$$

- If f has no cyclotomic factor but has a reciprocal factor, then the algorithm will give an explicit reciprocal factor.

We'll come back to this.

- Otherwise, the algorithm outputs the complete factorization of $f(x)$ into irreducible polynomials over $\mathbb{Q}$.

$$f(x) = \sum_{j=0}^{r} a_j x^{d_j}$$

f has no reciprocal factors

(other than constants)

- Otherwise, the algorithm outputs the complete factorization of $f(x)$ into irreducible polynomials over $\mathbb{Q}$.

$$f(x) = \sum_{j=0}^{r} a_j x^{d_j}$$

$$\begin{aligned} F = F(x_1, x_2, \ldots, x_r) \\ = a_r x_r + \cdots + a_1 x_1 + a_0, \end{aligned}$$

$$f(x) = F(x^{d_1}, x^{d_2}, \ldots, x^{d_r})$$

$$\boxed{f(x) = \sum_{j=0}^{r} a_j x^{d_j}, \quad F(x_1, \ldots, x_r) = a_0 + \sum_{j=1}^{r} a_j x_j}$$

$$(1) \qquad \begin{pmatrix} d_1 \\ \vdots \\ d_r \end{pmatrix} = (m_{ij})_{r \times t} \begin{pmatrix} v_1 \\ \vdots \\ v_t \end{pmatrix}$$

$$d_i = m_{i1} v_1 + \cdots + m_{it} v_t, \ 1 \leq i \leq r$$

$$\boxed{f(x) = \sum_{j=0}^{r} a_j x^{d_j}, \quad F(x_1, \ldots, x_r) = a_0 + \sum_{j=1}^{r} a_j x_j}$$

(1) $\quad d_i = m_{i1} v_1 + \cdots + m_{it} v_t, \ 1 \leq i \leq r$

(m_{ij}) will come from a finite set depending only on F

$v_j \in \mathbb{Z}$ show exist for some (m_{ij})

$$f(x) = \sum_{j=0}^{r} a_j x^{d_j}, \quad F(x_1,\dots,x_r) = a_0 + \sum_{j=1}^{r} a_j x_j$$

(1) $\quad d_i = m_{i1}v_1 + \cdots + m_{it}v_t, \ 1 \le i \le r$

$$F(y_1^{m_{11}} \cdots y_t^{m_{1t}}, \dots, y_1^{m_{r1}} \cdots y_t^{m_{rt}})$$

$$y_j = x^{v_j}, \quad 1 \le j \le t$$

$$F(x^{d_1}, x^{d_2}, \dots, x^{d_r}) = f(x)$$

Thought: A factorization in $\mathbb{Z}[y_1,\dots,y_t]$ implies a factorization of $f(x)$ in $\mathbb{Z}[x]$.

$$f(x) = \sum_{j=0}^{r} a_j x^{d_j}, \quad F(x_1,\dots,x_r) = a_0 + \sum_{j=1}^{r} a_j x_j$$

(1) $\quad d_i = m_{i1}v_1 + \cdots + m_{it}v_t, \ 1 \le i \le r$

$$F(y_1^{m_{11}} \cdots y_t^{m_{1t}}, \dots, y_1^{m_{r1}} \cdots y_t^{m_{rt}})$$

$$y_j = x^{v_j}, \quad 1 \le j \le t$$

$$F(x^{d_1}, x^{d_2}, \dots, x^{d_r}) = f(x)$$

Counter-Thought: We want m_{ij} and v_j in $\mathbb{Z}$, but not necessarily positive.

$$f(x) = \sum_{j=0}^{r} a_j x^{d_j}, \quad F(x_1,\dots,x_r) = a_0 + \sum_{j=1}^{r} a_j x_j$$

(1) $\quad d_i = m_{i1}v_1 + \cdots + m_{it}v_t, \ 1 \le i \le r$

$$J\,F(y_1^{m_{11}} \cdots y_t^{m_{1t}}, \dots, y_1^{m_{r1}} \cdots y_t^{m_{rt}})$$

$$y_1^{u_1} \cdots y_t^{u_t} F(y_1^{m_{11}} \cdots y_t^{m_{1t}}, \dots, y_1^{m_{r1}} \cdots y_t^{m_{rt}})$$

Recall: Factor and substitute $y_j = x^{v_j}$.

$$f(x) = \sum_{j=0}^{r} a_j x^{d_j}, \quad F(x_1,\dots,x_r) = a_0 + \sum_{j=1}^{r} a_j x_j$$

(1) $\quad d_i = m_{i1}v_1 + \cdots + m_{it}v_t, \ 1 \le i \le r$

(2) $$y_1^{u_1} \cdots y_t^{u_t} F(y_1^{m_{11}} \cdots y_t^{m_{1t}}, \dots, y_1^{m_{r1}} \cdots y_t^{m_{rt}}) = F_1(y_1,\dots,y_t) \cdots F_s(y_1,\dots,y_t)$$

(3) $$f(x) = \prod_{i=1}^{s} x^{w_i} F_i(x^{v_1}, \dots, x^{v_t})$$

Recall: Factor and substitute $y_j = x^{v_j}$.

$$f(x) = \sum_{j=0}^{r} a_j x^{d_j}, \quad F(x_1,\dots,x_r) = a_0 + \sum_{j=1}^{r} a_j x_j$$

(1) $\quad d_i = m_{i1}v_1 + \cdots + m_{it}v_t, \ 1 \le i \le r$

(2) $$y_1^{u_1} \cdots y_t^{u_t} F(y_1^{m_{11}} \cdots y_t^{m_{1t}}, \dots, y_1^{m_{r1}} \cdots y_t^{m_{rt}}) = F_1(y_1,\dots,y_t) \cdots F_s(y_1,\dots,y_t)$$

(3) $$f(x) = \prod_{i=1}^{s} x^{w_i} F_i(x^{v_1}, \dots, x^{v_t})$$

Conclusion: (1) and (2) imply (3)

Theorem (A. Schinzel, 1969): *Fix*

$$F = a_r x_r + \cdots + a_1 x_1 + a_0,$$

with a_j nonzero integers. There exists a finite computable set of matrices S with integer entries, depending only on F, with the following property:

Suppose the vector

$$\overrightarrow{d} = \langle d_1, d_2, \dots, d_r \rangle \in \mathbb{Z}^r,$$

with $d_r > \cdots > d_1 > 0$, is such that

$$f(x) = F(x^{d_1}, x^{d_2}, \dots, x^{d_r})$$

has no non-constant reciprocal factor.

Then $\exists$ $r \times t$ matrix $M = (m_{ij}) \in S$ of rank $t \le r$ and a vector

$$\overrightarrow{v} = \langle v_1, v_2, \dots, v_t \rangle \in \mathbb{Z}^t$$

such that

$$\begin{pmatrix} d_1 \\ \vdots \\ d_r \end{pmatrix} = M \begin{pmatrix} v_1 \\ \vdots \\ v_t \end{pmatrix}$$

holds and the factorization given by

$$y_1^{u_1} \cdots y_t^{u_t} F(y_1^{m_{11}} \cdots y_t^{m_{1t}}, \dots, y_1^{m_{r1}} \cdots y_t^{m_{rt}}) = F_1(y_1,\dots,y_t) \cdots F_s(y_1,\dots,y_t)$$

in $\mathbb{Z}[y_1,\dots,y_t]$ into irreducibles implies

$$f(x) = \prod_{i=1}^{s} x^{w_i} F_i(x^{v_1}, \dots, x^{v_t})$$

as a product of polynomials in $\mathbb{Z}[x]$ each of which is either irreducible over $\mathbb{Q}$ or a constant.

This all works for "some" $(m_{ij}) \in S$.

---

(1) $\quad d_i = m_{i1}v_1 + \cdots + m_{it}v_t, \ 1 \le i \le r$

(2) $$y_1^{u_1} \cdots y_t^{u_t} F(y_1^{m_{11}} \cdots y_t^{m_{1t}}, \dots, y_1^{m_{r1}} \cdots y_t^{m_{rt}}) = F_1(y_1,\dots,y_t) \cdots F_s(y_1,\dots,y_t)$$

$$y_j = x^{v_j}, \quad 1 \le j \le t$$

(3) $$f(x) = \prod_{i=1}^{s} x^{w_i} F_i(x^{v_1}, \dots, x^{v_t})$$

This Part of Algorithm:

- Compute set of matrices S.

The set of matrices depends on

$$F = a_r x_r + \cdots + a_1 x_1 + a_0,$$

not on $d_1, d_2, \dots, d_r$.

This Part of Algorithm:

- Compute set of matrices S.
- Determine all solutions to (1).

(1) $\quad d_i = m_{i1}v_1 + \cdots + m_{it}v_t, \ 1 \le i \le r$

Easy Lemma: *There's an algorithm that determines for a given integral matrix $(m_{ij}) \in S$ whether (1) holds for some $v_j \in \mathbb{Z}$. If it does, the solution is unique and the algorithm outputs the solution. The algorithm runs in time $O_{r,H}(\log n)$.*

This Part of Algorithm:

- Compute set of matrices S.
- Determine all solutions to (1).
- For each solution, completely factor $J\,F(y_1^{m_{11}} \cdots y_t^{m_{1t}}, \dots, y_1^{m_{r1}} \cdots y_t^{m_{rt}})$.

(2) $$y_1^{u_1} \cdots y_t^{u_t} F(y_1^{m_{11}} \cdots y_t^{m_{1t}}, \dots, y_1^{m_{r1}} \cdots y_t^{m_{rt}}) = F_1(y_1,\dots,y_t) \cdots F_s(y_1,\dots,y_t)$$

This Part of Algorithm:

- Compute set of matrices S.
- Determine all solutions to (1).
- For each solution, completely factor $J\,F(y_1^{m_{11}} \cdots y_t^{m_{1t}}, \dots, y_1^{m_{r1}} \cdots y_t^{m_{rt}})$.
- Substitute $y_j = x^{v_j}$ to obtain (3)'s.

(3) $$f(x) = \prod_{i=1}^{s} x^{w_i} F_i(x^{v_1}, \dots, x^{v_t})$$

Each $x^{w_i} F_i(x^{v_1}, \dots, x^{v_t})$ is a constant or irreducible for some solution to (1).

This Part of Algorithm:

- Compute set of matrices S.
- Determine all solutions to (1).
- For each solution, completely factor $J\,F(y_1^{m_{11}} \cdots y_t^{m_{1t}}, \dots, y_1^{m_{r1}} \cdots y_t^{m_{rt}})$.
- Substitute $y_j = x^{v_j}$ to obtain (3)'s.
- Choose (3) with the largest number of non-constant $x^{w_i} F_i(x^{v_1}, \dots, x^{v_t})$.

(3) $$f(x) = \prod_{i=1}^{s} x^{w_i} F_i(x^{v_1}, \dots, x^{v_t})$$

- If f has no cyclotomic factor but has a reciprocal factor, then the algorithm will give an explicit reciprocal factor.

We've checked:

f does not have a cyclotomic factor.

We want to know:

Does f have a reciprocal factor?

- If f has no cyclotomic factor but has a reciprocal factor, then the algorithm will give an explicit reciprocal factor.

Does f have a reciprocal factor?

Suppose $w(x)$ is a reciprocal factor.

$$w(\alpha) = 0 \implies \alpha \ne 0 \text{ and } w(1/\alpha) = 0$$
$$\implies f(\alpha) = 0 \text{ and } g(\alpha) = 0,$$

where $g(x) = x^{\deg f} f(1/x) \ne f(x)$

We want to compute $\gcd(f, g)$.

- If f has no cyclotomic factor but has a reciprocal factor, then the algorithm will give an explicit reciprocal factor.

In general, if f and g are sparse polynomials around degree n in $\mathbb{Z}[x]$, how does one compute $\gcd(f,g)$?

Some items to keep in mind:

→ The Euclidean algorithm will run in time that is polynomial in n, not $\log n$.

- If f has no cyclotomic factor but has a reciprocal factor, then the algorithm will give an explicit reciprocal factor.

→ Plaisted (1977) has shown that this problem is at least as hard as any problem in NP.

Plaisted's takes f and g to be divisors of $x^N - 1$ where N is a product of small primes.

We are interested in the case that both f and g do not have a cyclotomic factor.

Theorem (A. Schinzel and M.F.): *There is an algorithm which takes as input two polynomials $f(x)$ and $g(x)$ in $\mathbb{Z}[x]$, each of degree $\le n$ and height $\le H$ and having $\le r+1$ nonzero terms, with at least one of $f(x)$ and $g(x)$ free of any cyclotomic factors, and outputs the value of $\gcd_{\mathbb{Z}}(f(x),g(x))$ and runs in time $O_{r,H}(\log n)$.*

Corollary: *If $f(x), g(x) \in \mathbb{Z}[x]$ with $f(x)$ or $g(x)$ not divisible by a cyclotomic polynomial, then $\gcd_{\mathbb{Z}}(f(x),g(x))$ has $O_{r,H}(1)$ terms.*

Example: For almost any $a_j, b_j \in \mathbb{Z}$ with $|a_j| \le 1000$ and $|b_j| \le 1000$ and positive integers $e_1,\dots,e_{100}$ and $f_1,\dots,f_{100}$, the greatest common divisor of

$$f(x) = \sum_{j=0}^{100} a_j x^{e_j} \quad \text{and} \quad g(x) = \sum_{j=0}^{100} b_j x^{f_j}$$

has $\le c$ terms.

Corollary: *If $f(x), g(x) \in \mathbb{Z}[x]$ with $f(x)$ or $g(x)$ not divisible by a cyclotomic polynomial, then $\gcd_{\mathbb{Z}}(f(x),g(x))$ has $O_{r,H}(1)$ terms.*

Note that if a and b are relatively prime positive integers, then

$$\gcd\left(x^{ab}-1, (x^a-1)(x^b-1)\right) = \frac{(x^a-1)(x^b-1)}{x-1},$$

which can have arbitrarily many terms.

Theorem (A. Schinzel and M.F.): *There is an algorithm which takes as input two polynomials $f(x)$ and $g(x)$ in $\mathbb{Z}[x]$, each of degree $\le n$ and height $\le H$ and having $\le r+1$ nonzero terms, with at least one of $f(x)$ and $g(x)$ free of any cyclotomic factors, and outputs the value of $\gcd_{\mathbb{Z}}(f(x),g(x))$ and runs in time $O_{r,H}(\log n)$.*

$$f(x) = \sum_{j=1}^{k} a_j x^{d_j} \;\rightarrow\; F_1(x) = \sum_{j=1}^{k} a_j x_j$$

Lemma (Bombieri and Zannier): *Let*

$$F_1, F_2 \in \mathbb{Q}[x_1,\dots,x_k]$$

be coprime polynomials. There exists a number $c_1(F_1,F_2)$ with the following property. If $\overrightarrow{u} = \langle u_1,\dots,u_k\rangle \in \mathbb{Z}^k$, $\xi \ne 0$ is algebraic and

$$F_1(\xi^{u_1},\dots,\xi^{u_k}) = F_2(\xi^{u_1},\dots,\xi^{u_k}) = 0,$$

then either ξ is a root of unity or there exists a non-zero vector $\overrightarrow{v} \in \mathbb{Z}^k$ having length at most c_1 and orthogonal to $\overrightarrow{u}$.

$$f(x) = \sum_{j=1}^{k} a_j x^{d_j} \;\rightarrow\; F_1(x) = \sum_{j=1}^{k} a_j x_j$$

Lemma (Bombieri and Zannier): *Let*

$$F_1, F_2 \in \mathbb{Q}[x_1,\dots,x_k]$$

be coprime polynomials. There exists a number $c_1(F_1,F_2)$ with the following property. If $f(\xi) = g(\xi) = 0$, then there exists a non-zero vector $\overrightarrow{v} \in \mathbb{Z}^k$ having length at most c_1 and orthogonal to $\overrightarrow{u}$.

$$\overrightarrow{u} = \langle d_1,\dots,d_k\rangle$$

Idea: The lattice of vectors orthogonal to $\overrightarrow{v}$ is $(k-1)$-dimensional so that there exists a vector $\langle e_1,\dots,e_{k-1}\rangle$ and a matrix $\mathcal{M}$ in $\mathbb{Z}^{k-1}$ satisfying

$$\begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_k \end{pmatrix} = \mathcal{M} \cdot \begin{pmatrix} e_1 \\ e_2 \\ \vdots \\ e_{k-1} \end{pmatrix}.$$

So

$$d_i = \sum_{j=1}^{k-1} m_{ij} e_j,$$

with the $m_{ij} \in \mathbb{Z}$ bounded.

$$d_i = \sum_{j=1}^{k-1} m_{ij} e_j$$

$$f(x) = \sum_{i=1}^{k} a_i x^{d_i} = \sum_{i=1}^{k} a_i \prod_{j=1}^{k-1} \left(x^{e_j}\right)^{m_{ij}}$$

$$F_1^{(2)}(y_1,\dots,y_{k-1}) = \sum_{i=1}^{k} a_i \prod_{j=1}^{k-1} y_j^{m_{ij}}$$

$$\Longrightarrow \quad \begin{aligned} F_1^{(2)}(x^{e_1},\dots,x^{e_{k-1}}) &= f(x) \\ F_2^{(2)}(x^{e_1},\dots,x^{e_{k-1}}) &= g(x) \end{aligned}$$

$$F_1^{(2)}(y_1,\dots,y_{k-1}) = \sum_{i=1}^{k} a_i \prod_{j=1}^{k-1} y_j^{m_{ij}}$$

$$F_2^{(2)}(y_1,\dots,y_{k-1}) = \sum_{i=1}^{k} b_i \prod_{j=1}^{k-1} y_j^{m_{ij}}$$

Issues to Deal With:

- Bombieri & Zannier's work requires *relatively prime* multivariate polynomials.

Issues to Deal With:

- Bombieri & Zannier's work requires *relatively prime* multivariate polynomials.

  Divide by $\gcd(F_1^{(2)}, F_2^{(2)})$.

  Keep track of the gcd's. They are part of $\gcd(f(x), g(x))$.

Issues to Deal With:

- Bombieri & Zannier's work requires *relatively prime* multivariate polynomials.
- Bombieri & Zannier's work requires polynomials.

$$F_1^{(2)}(y_1,\dots,y_{k-1}) = \sum_{i=1}^{k} a_i \prod_{j=1}^{k-1} y_j^{m_{ij}}$$

$$F_2^{(2)}(y_1,\dots,y_{k-1}) = \sum_{i=1}^{k} b_i \prod_{j=1}^{k-1} y_j^{m_{ij}}$$

Issues to Deal With:

- Bombieri & Zannier's work requires *relatively prime* multivariate polynomials.
- Bombieri & Zannier's work requires polynomials.

  Use $J F_1^{(2)}$ and $J F_2^{(2)}$.

Issues to Deal With:

- Bombieri & Zannier's work requires *relatively prime* multivariate polynomials.
- Bombieri & Zannier's work requires polynomials.
- Some variables may be missing.

$$F_1^{(2)}(y_1,\dots,y_{k-1}) = \sum_{i=1}^{k} a_i \prod_{j=1}^{k-1} y_j^{m_{ij}}$$

Issues to Deal With:

- Bombieri & Zannier's work requires *relatively prime* multivariate polynomials.
- Bombieri & Zannier's work requires polynomials.
- Some variables may be missing.
- The induction step may end before it ends.

  So what?