LECTURE 2: THE NUMBER THEORETIC TRANSFORM (NTT)

Post-Quantum Cryptography

Florida Atlantic University

Fall 2026



Outline

01 The cost problem

02 Evaluate, multiply, go back

03 A fast NTT

04 Going back: the inverse NTT

05 The NTT in ML-DSA and ML-KEM

06 Summary

One polynomial product costs $n^2 = 65{,}536$ multiplications. Can we do better?

01

The cost problem

Where we stopped

- Our running example in $\mathbb{Z}_{17}[x]/(x^4+1)$:

$$a = 1 + 2x + 3x^2 + 4x^3, \qquad b = 5 + 6x + 7x^2 + 8x^3$$

$$a \cdot b = 12 + 15x + 2x^2 + 9x^3$$

- Schoolbook: multiply every pair, collect, fold back with $x^4 = -1$, reduce mod 17.
- The cost is n^2 multiplications:

n	4	16	256	1024
n^2	16	256	65,536	1,048,576

Where we stopped

- Our running example in $\mathbb{Z}_{17}[x]/(x^4+1)$:

$$a = 1 + 2x + 3x^2 + 4x^3, \qquad b = 5 + 6x + 7x^2 + 8x^3$$

$$a \cdot b = 12 + 15x + 2x^2 + 9x^3$$

- Schoolbook: multiply every pair, collect, fold back with $x^4 = -1$, reduce mod 17.
- The cost is n^2 multiplications:

n	4	16	256	1024
n^2	16	256	65,536	1,048,576

You have seen this kind of trick before

Task	Slow way	Fast way	Lecture
multiply big integers	schoolbook, n^2	Karatsuba, $n^{1.58}$	FF multiplication
reduce mod p	long division	Barrett, Montgomery	FF reduction
multiply in $\mathcal{R}_q$	schoolbook, n^2	NTT, $n \log_2 n$	today

- Each time, a change of representation makes the expensive step cheap.
- NTT stands for Number Theoretic Transform. It is the FFT, done with integers mod q instead of complex numbers.

You have seen this kind of trick before

Task	Slow way	Fast way	Lecture
multiply big integers	schoolbook, n^2	Karatsuba, $n^{1.58}$	FF multiplication
reduce mod p	long division	Barrett, Montgomery	FF reduction
multiply in $\mathcal{R}_q$	schoolbook, n^2	NTT, $n \log_2 n$	today

- Each time, a change of representation makes the expensive step cheap.
- NTT stands for Number Theoretic Transform. It is the FFT, done with integers mod q instead of complex numbers.

02

Evaluate, multiply, go back

A polynomial is also its values

- Two points fix a line. Three points fix a parabola.
- In general, a polynomial with n coefficients is fixed by its values at n different points.
- So we can store a in two ways:

Coefficients

$a = (1, 2, 3, 4)$

meaning $1 + 2x + 3x^2 + 4x^3$

Values at 4 points

$a(2) = 15,\ \ a(8) = 13,$

$a(15) = 11,\ \ a(9) = 16 \pmod{17}$

- Check one: $a(2) = 1 + 2 \cdot 2 + 3 \cdot 4 + 4 \cdot 8 = 49 = 2 \cdot 17 + 15.$

A polynomial is also its values

- Two points fix a line. Three points fix a parabola.
- In general, a polynomial with n coefficients is fixed by its values at n different points.
- So we can store a in two ways:

Coefficients

$a = (1, 2, 3, 4)$

meaning $1 + 2x + 3x^2 + 4x^3$

Values at 4 points

$a(2) = 15$, $a(8) = 13$,

$a(15) = 11$, $a(9) = 16$ (mod 17)

- Check one: $a(2) = 1 + 2 \cdot 2 + 3 \cdot 4 + 4 \cdot 8 = 49 = 2 \cdot 17 + 15$.

A polynomial is also its values

- Two points fix a line. Three points fix a parabola.
- In general, a polynomial with n coefficients is fixed by its values at n different points.
- So we can store a in two ways:

Coefficients

$a = (1, 2, 3, 4)$

meaning $1 + 2x + 3x^2 + 4x^3$

Values at 4 points

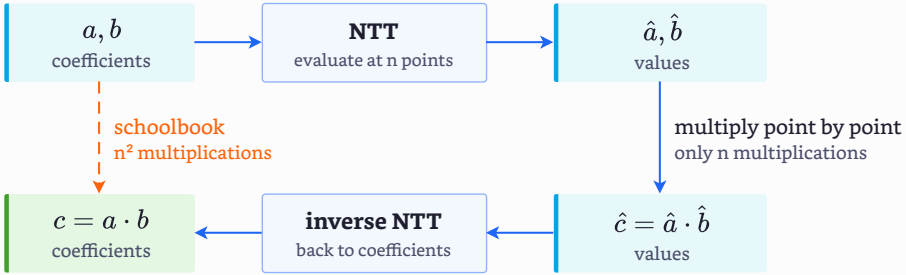
$a(2) = 15$, $a(8) = 13$,

$a(15) = 11$, $a(9) = 16$ (mod 17)

- Check one: $a(2) = 1 + 2 \cdot 2 + 3 \cdot 4 + 4 \cdot 8 = 49 = 2 \cdot 17 + 15$.

Multiplying values is cheap

- If $c = a \cdot b$, then at any point r: $\quad c(r) = a(r) \cdot b(r)$.
- One multiplication per point, so n multiplications in total.



- What remains is to make the two conversions cheap: the NTT and its inverse.

The wrap-around needs special points

- The product in $\mathcal{R}_q$ is folded with $x^4 = -1$. A random point r does not know about the fold.
- Unless the point itself satisfies $r^4 = -1$. Then, at that point:

$$r^4 = -1, \qquad r^5 = -r, \qquad r^6 = -r^2$$

exactly the fold rule from Lecture 1.

Choose the points well

Use the n roots of $x^n + 1$, the numbers r with $r^n = -1 \pmod q$.

Then $c(r) = a(r) \cdot b(r)$ holds for the folded product too.

- For $q = 17, n = 4$ we need four numbers with $r^4 \equiv 16$. Do they exist?

The wrap-around needs special points

- The product in $\mathcal{R}_q$ is folded with $x^4 = -1$. A random point r does not know about the fold.
- Unless the point itself satisfies $r^4 = -1$. Then, at that point:

$$r^4 = -1, \qquad r^5 = -r, \qquad r^6 = -r^2$$

exactly the fold rule from Lecture 1.

Choose the points well

Use the n roots of $x^n + 1$, the numbers r with $r^n = -1 \pmod{q}$.

Then $c(r) = a(r) \cdot b(r)$ holds for the folded product too.

- For $q = 17, n = 4$ we need four numbers with $r^4 \equiv 16$. Do they exist?

The wrap-around needs special points

- The product in $\mathcal{R}_q$ is folded with $x^4 = -1$. A random point r does not know about the fold.
- Unless the point itself satisfies $r^4 = -1$. Then, at that point:

$$r^4 = -1, \qquad r^5 = -r, \qquad r^6 = -r^2$$

exactly the fold rule from Lecture 1.

Choose the points well

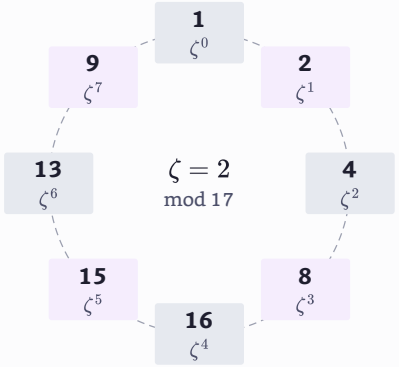
Use the n roots of $x^n + 1$, the numbers r with $r^n = -1 \pmod{q}$.

Then $c(r) = a(r) \cdot b(r)$ holds for the folded product too.

- For $q = 17$, $n = 4$ we need four numbers with $r^4 \equiv 16$. Do they exist?

Powers of 2 mod 17

- $2^4 = 16 = -1$ and $2^8 = 1$, so the powers of $\zeta = 2$ go around a circle of 8.



odd powers: 2, 8, 15, 9
each has $r^4 = 16 = -1$

even powers: 1, 4, 16, 13
$r^4 = +1$, not used

- ζ is a *primitive $2n$-th root of unity*: $\zeta^{2n} = 1$ and $\zeta^n = -1$.
- Check one point: $8^4 = 4096 = 240 \cdot 17 + 16 \equiv -1 \pmod{17}$.

The idea, checked on numbers

- Evaluate a and b at the four points, multiply, and compare with $c = 12 + 15x + 2x^2 + 9x^3$:

point r	$a(r)$	$b(r)$	$a(r) \cdot b(r)$	**mod 17**	$c(r)$
2	15	7	105	3	3
8	13	7	91	6	6
15	11	8	88	3	3
9	16	15	240	2	2

- Every row matches. Four multiplications gave the values of the product.
- The values $(3, 6, 3, 2)$ fix c. Getting the coefficients back from them is the inverse NTT.

The idea, checked on numbers

- Evaluate a and b at the four points, multiply, and compare with $c = 12 + 15x + 2x^2 + 9x^3$:

point r	$a(r)$	$b(r)$	$a(r) \cdot b(r)$	**mod 17**	$c(r)$
2	15	7	105	3	3
8	13	7	91	6	6
15	11	8	88	3	3
9	16	15	240	2	2

- Every row matches. Four multiplications gave the values of the product.
- The values $(3, 6, 3, 2)$ fix c. Getting the coefficients back from them is the inverse NTT.

Definition of the NTT

Number Theoretic Transform

$\mathrm{NTT}(a)$ is the list of values of a at the n odd powers of ζ:

$$\mathrm{NTT}(a) = \big(a(\zeta^1),\ a(\zeta^3),\ a(\zeta^5),\ \ldots,\ a(\zeta^{2n-1})\big)$$

- Toy: $\mathrm{NTT}(a) = (a(2), a(8), a(15), a(9)) = (15, 13, 11, 16)$, in some order.
- Done directly, each value costs n multiplications, and there are n values: n^2 again.

So far

The idea is right, but direct evaluation saves nothing. We need a fast way to compute all n values.

Definition of the NTT

Number Theoretic Transform

$\mathrm{NTT}(a)$ is the list of values of a at the n odd powers of ζ:

$$\mathrm{NTT}(a) = \big(a(\zeta^1),\, a(\zeta^3),\, a(\zeta^5),\, \ldots,\, a(\zeta^{2n-1})\big)$$

- Toy: $\mathrm{NTT}(a) = (a(2), a(8), a(15), a(9)) = (15, 13, 11, 16)$, in some order.
- Done directly, each value costs n multiplications, and there are n values: n^2 again.

So far

The idea is right, but direct evaluation saves nothing. We need a fast way to compute all n values.

03

A fast NTT

Split the problem in half

- Since $4^2 = 16 = -1$:

$$x^4 + 1 = (x^2 - 4)(x^2 + 4) \pmod{17}$$

- The four points split into two pairs:
  - 2 and 15 are the roots of $x^2 - 4$ $\quad (2^2 = 4,\ 15^2 = 225 \equiv 4)$
  - 8 and 9 are the roots of $x^2 + 4$ $\quad (8^2 = 64 \equiv 13 = -4,\ 9^2 = 81 \equiv 13)$

- Plan: first reduce $a \bmod x^2 - 4$ and $\bmod x^2 + 4$. Each half has only 2 coefficients and is responsible for 2 of the points.

Split the problem in half

- Since $4^2 = 16 = -1$:
$$x^4 + 1 = (x^2 - 4)(x^2 + 4) \pmod{17}$$
- The four points split into two pairs:
  - 2 and 15 are the roots of $x^2 - 4$ ($2^2 = 4$, $15^2 = 225 \equiv 4$)
  - 8 and 9 are the roots of $x^2 + 4$ ($8^2 = 64 \equiv 13 = -4$, $9^2 = 81 \equiv 13$)
- Plan: first reduce a mod $x^2 - 4$ and mod $x^2 + 4$. Each half has only 2 coefficients and is responsible for 2 of the points.

Reducing mod $x^2 - c$ is one line

- Write $a = (a_0 + a_1 x) + x^2(a_2 + a_3 x)$ and replace x^2 by c or by $-c$:

mod $x^2 - c$

$(a_0 + c\,a_2) + (a_1 + c\,a_3)\,x$

mod $x^2 + c$

$(a_0 - c\,a_2) + (a_1 - c\,a_3)\,x$

Our $a = (1, 2, 3, 4)$ with $c = 4$

mod $x^2 - 4$: $(1 + 4 \cdot 3) + (2 + 4 \cdot 4)\,x = 13 + 18x \equiv 13 + x$

mod $x^2 + 4$: $(1 - 4 \cdot 3) + (2 - 4 \cdot 4)\,x = -11 - 14x \equiv 6 + 3x$

- Both results reuse the same products $4 \cdot 3$ and $4 \cdot 4$: two multiplications for the whole step.

Reducing mod $x^2 - c$ is one line

- Write $a = (a_0 + a_1 x) + x^2(a_2 + a_3 x)$ and replace x^2 by c or by $-c$:

mod $x^2 - c$

$(a_0 + c\,a_2) + (a_1 + c\,a_3)\,x$

mod $x^2 + c$

$(a_0 - c\,a_2) + (a_1 - c\,a_3)\,x$

Our $a = (1, 2, 3, 4)$ **with** $c = 4$

mod $x^2 - 4$: $(1 + 4 \cdot 3) + (2 + 4 \cdot 4)\,x = 13 + 18x \equiv 13 + x$

mod $x^2 + 4$: $(1 - 4 \cdot 3) + (2 - 4 \cdot 4)\,x = -11 - 14x \equiv 6 + 3x$

- Both results reuse the same products $4 \cdot 3$ and $4 \cdot 4$: two multiplications for the whole step.

Reducing mod $x^2 - c$ is one line

- Write $a = (a_0 + a_1 x) + x^2(a_2 + a_3 x)$ and replace x^2 by c or by $-c$:

mod $x^2 - c$

$(a_0 + c\,a_2) + (a_1 + c\,a_3)\,x$

mod $x^2 + c$

$(a_0 - c\,a_2) + (a_1 - c\,a_3)\,x$

Our $a = (1, 2, 3, 4)$ **with** $c = 4$

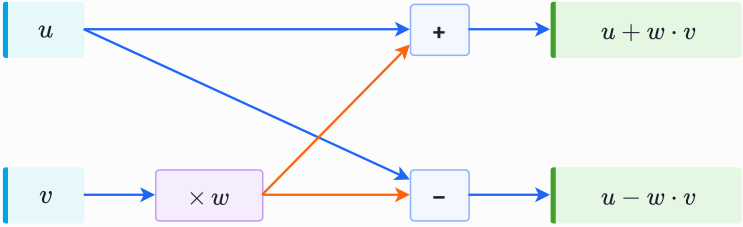
mod $x^2 - 4$: $(1 + 4 \cdot 3) + (2 + 4 \cdot 4)\,x = 13 + 18x \equiv 13 + x$

mod $x^2 + 4$: $(1 - 4 \cdot 3) + (2 - 4 \cdot 4)\,x = -11 - 14x \equiv 6 + 3x$

- Both results reuse the same products $4 \cdot 3$ and $4 \cdot 4$: two multiplications for the whole step.

The butterfly

- The pattern "$u + w \cdot v$ and $u - w \cdot v$" is called a **Cooley-Tukey butterfly**. w is the twiddle factor.

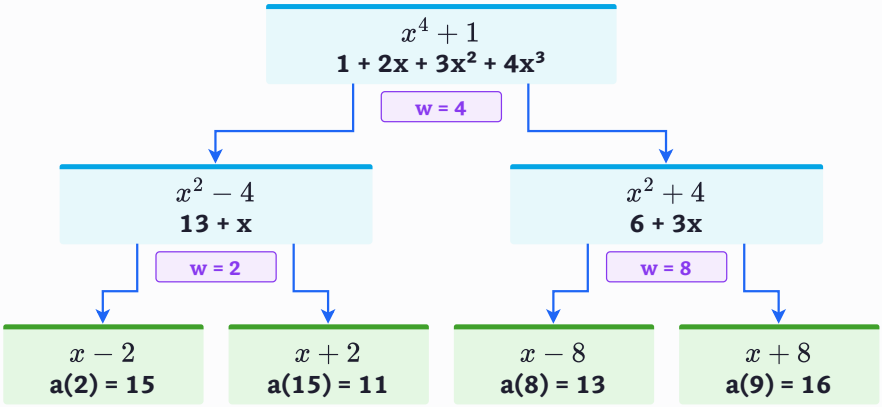


Example: $u = 1,\ v = 3,\ w = 4$: $(1 + 12,\ 1 - 12) = (13, 6) \pmod{17}$

- One multiplication, one addition, one subtraction, all mod q.

Split again until single numbers remain

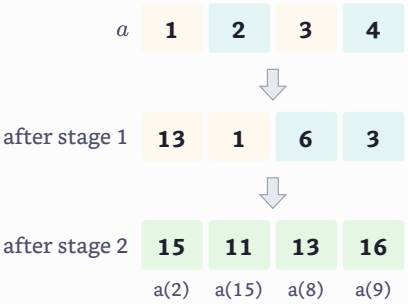
- $x^2 - 4 = (x-2)(x+2)$ and $x^2 + 4 = (x-8)(x+8)$. Reducing $u + vx$ mod $x - r$ gives $u + r\,v$, the value at r.



- $13 + x \to 13 \pm 2 = 15, 11$ $\quad 6 + 3x \to 6 \pm 8 \cdot 3 = 30, -18 \equiv 13, 16$

The same steps on an array

- In code, a is one array that is updated in place, one stage at a time. Same colour, same butterfly.



Stage 1, $w = 4$

pairs (a_0, a_2) and (a_1, a_3)

$1 \pm 4 \cdot 3 = 13, -11 \equiv 13, 6$

$2 \pm 4 \cdot 4 = 18, -14 \equiv 1, 3$

Stage 2, $w = 2$ **and** $w = 8$

$(13, 1)$: $13 \pm 2 \cdot 1 = 15, 11$

$(6, 3)$: $6 \pm 8 \cdot 3 = 30, -18 \equiv 13, 16$

- 4 multiplications in total, against 16 for direct evaluation.

Your turn: $\mathrm{NTT}(b)$

- Compute $\mathrm{NTT}(b)$ for $b = (5, 6, 7, 8)$ with the same stages: $w = 4$, then $w = 2$ and $w = 8$.

Your turn: $\mathrm{NTT}(b)$

- Compute $\mathrm{NTT}(b)$ for $b = (5, 6, 7, 8)$ with the same stages: $w = 4$, then $w = 2$ and $w = 8$.

Stage 1, $w = 4$

$5 \pm 4 \cdot 7 = 33, -23 \equiv 16, 11 \qquad 6 \pm 4 \cdot 8 = 38, -26 \equiv 4, 8$

array: $(16, 4, 11, 8)$

Stage 2

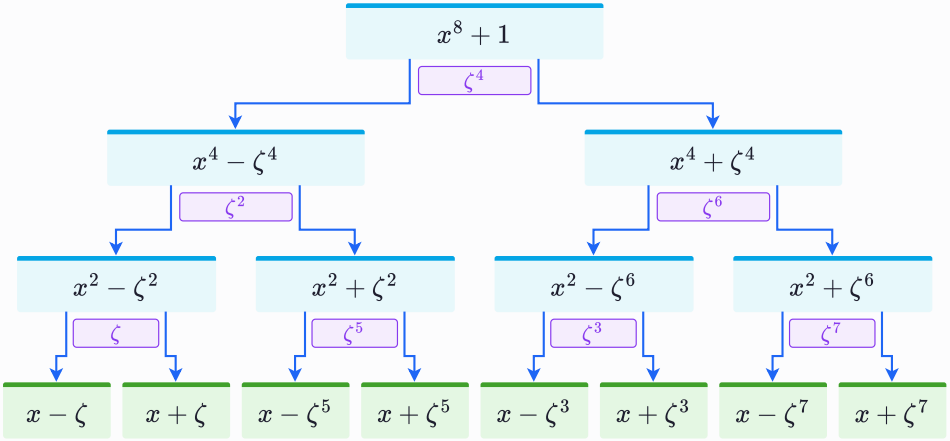
$(16, 4)$**,** $w = 2$**:** $\quad 16 \pm 8 = 24, 8 \equiv 7, 8$

$(11, 8)$**,** $w = 8$**:** $\quad 11 \pm 64 = 75, -53 \equiv 7, 15$

$\mathrm{NTT}(b) = (7, 8, 7, 15) = (b(2), b(15), b(8), b(9))$

Bigger n, deeper tree

- $n = 8$: three levels of splits. Each level is one stage of 4 butterflies; the chips are the twiddles.



- Every stage touches every coefficient once: $n/2$ butterflies per stage.

Counting the work

Cost of a fast NTT

$\log_2 n$ stages $\times$ $n/2$ butterflies $\times$ 1 multiplication $= \dfrac{n}{2} \log_2 n$

n	stages	fast NTT	direct evaluation n^2
4	2	4	16
8	3	12	64
256	8	1,024	65,536

- Each split halves the problem, and 256 halves down to 1 in only 8 steps.

Why the output order looks shuffled

- The fast NTT returned $(a(2), a(15), a(8), a(9))$, which is $(a(\zeta^1), a(\zeta^5), a(\zeta^3), a(\zeta^7))$.
- Output i is the value at $\zeta^{2\,\mathrm{brv}(i)+1}$, where brv reverses the bits of i:

i	0	1	2	3	4	5	6	7
i in binary	000	001	010	011	100	101	110	111
reversed	000	100	010	110	001	101	011	111
$\mathrm{brv}(i)$	0	4	2	6	1	5	3	7

- The order does not matter. Pointwise multiplication treats every position alike, and the inverse NTT puts things back.

Why the output order looks shuffled

- The fast NTT returned $(a(2), a(15), a(8), a(9))$, which is $(a(\zeta^1), a(\zeta^5), a(\zeta^3), a(\zeta^7))$.
- Output i is the value at $\zeta^{2\,\mathrm{brv}(i)+1}$, where brv reverses the bits of i:

i	0	1	2	3	4	5	6	7
i in binary	000	001	010	011	100	101	110	111
reversed	000	100	010	110	001	101	011	111
$\mathrm{brv}(i)$	0	4	2	6	1	5	3	7

- The order does not matter. Pointwise multiplication treats every position alike, and the inverse NTT puts things back.

Twiddle factors are a fixed table

- The twiddles depend only on n, q and ζ, so they are computed once and stored. Block k uses $\zeta^{\mathrm{brv}(k)}$.

Ring	Twiddles used	First entries
toy, $q = 17, \zeta = 2$	3	$4, 2, 8$ (that is $\zeta^2, \zeta^1, \zeta^3$)
ML-DSA, $\zeta = 1753$	255	$4808194, 3765607, \ldots$
ML-KEM, $\zeta = 17$	127	$1729, 2580, 3289, \ldots$

- Software keeps them in a constant array, hardware in a small ROM next to the multiplier.

The whole forward NTT in code

```
def ntt(a, q, zetas):
    n = len(a)
    k = 1
    length = n // 2
    while length >= 1:
        for start in range(0, n, 2*length):
            w = zetas[k]
            k += 1
            for j in range(start, start+length):
                t = w * a[j+length] % q
                a[j+length] = (a[j] - t) % q
                a[j] = (a[j] + t) % q
        length //= 2
    return a
```

- `length`: distance between the two inputs of a butterfly. $n/2$, then $n/4$, down to 1.
- `start`: one block per twiddle.
- Inner loop: the butterfly.
- Same shape as Algorithm 41 in FIPS 204. You write it in Assignment 7.

04

Going back: the inverse NTT

Undoing one butterfly

- The forward butterfly gave $u' = u + wv$ and $v' = u - wv$. Add and subtract:

$$u' + v' = 2u, \qquad u' - v' = 2wv$$

- So $u = (u' + v')/2$ and $v = (u' - v') \cdot w^{-1}/2$.

Example: $u = 6,\ v = 8,\ w^{-1} = 13$: $(14,\ (6 - 8) \cdot 13) = (14, 8) \pmod{17}$

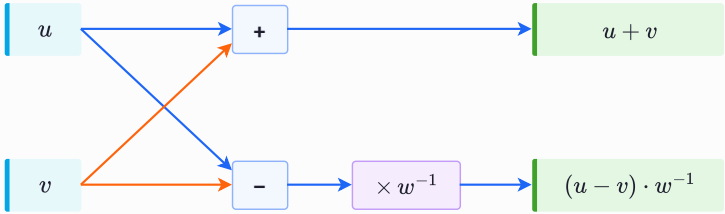
- This is the **Gentleman-Sande butterfly**. The halves pile up, one per stage, so we divide by n once at the end.

Undoing one butterfly

- The forward butterfly gave $u' = u + wv$ and $v' = u - wv$. Add and subtract:

$$u' + v' = 2u, \qquad u' - v' = 2wv$$

- So $u = (u' + v')/2$ and $v = (u' - v') \cdot w^{-1}/2$.

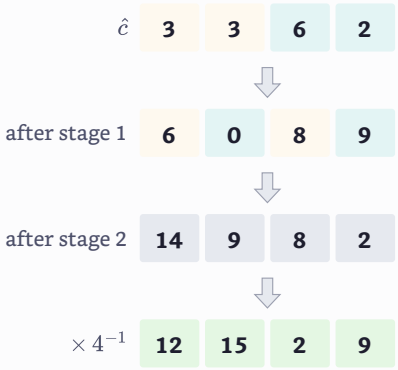


Example: $u = 6,\ v = 8,\ w^{-1} = 13$: $(14,\ (6 - 8) \cdot 13) = (14, 8) \pmod{17}$

- This is the **Gentleman-Sande butterfly**. The halves pile up, one per stage, so we divide by n once at the end.

The inverse NTT on numbers

- Inverse twiddles mod 17: $2^{-1} = 9, 8^{-1} = 15, 4^{-1} = 13$. The stages run in reverse order.



Stage 1, $w^{-1} = 9$ **and** $w^{-1} = 15$

$(3, 3)$: $(3 + 3,\ 0 \cdot 9) = (6, 0)$

$(6, 2)$: $(6 + 2,\ 4 \cdot 15) = (8, 60) \equiv (8, 9)$

Stage 2, $w^{-1} = 13$

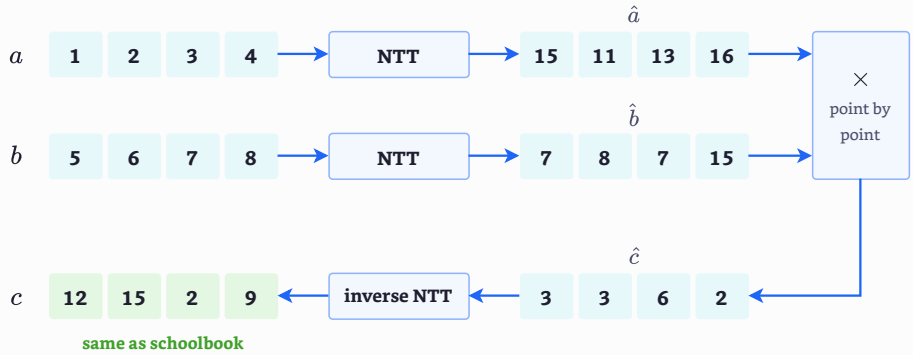
$(6, 8)$: $(14,\ -2 \cdot 13) \equiv (14, 8)$

$(0, 9)$: $(9,\ -9 \cdot 13) \equiv (9, 2)$

Scale by $4^{-1} = 13$

$(14, 9, 8, 2) \cdot 13 \equiv (12, 15, 2, 9)$, the schoolbook result

The full pipeline



Pointwise step

$15 \cdot 7 = 105 \equiv 3, \quad 11 \cdot 8 = 88 \equiv 3$

$13 \cdot 7 = 91 \equiv 6, \quad 16 \cdot 15 = 240 \equiv 2$

Cost for one product

2 forward NTTs, n pointwise products,

1 inverse NTT, n scalings

Is it really faster?

- Count every multiplication: $3 \cdot \frac{n}{2} \log_2 n + 2n$, against n^2 for schoolbook.

n	**schoolbook**	**NTT way**	
4	16	20	NTT is slower
16	256	128	2× faster
64	4,096	704	6× faster
256	65,536	3,584	18× faster
1024	1,048,576	17,408	60× faster

- Our toy is too small to profit. The win grows with n, and $n = 256$ is in the profitable range.

Is it really faster?

- Count every multiplication: $3 \cdot \frac{n}{2} \log_2 n + 2n$, against n^2 for schoolbook.

n	**schoolbook**	**NTT way**	
4	16	20	NTT is slower
16	256	128	2× faster
64	4,096	704	6× faster
256	65,536	3,584	18× faster
1024	1,048,576	17,408	60× faster

- Our toy is too small to profit. The win grows with n, and $n = 256$ is in the profitable range.

05

The NTT in ML-DSA and ML-KEM

When does a good ζ exist?

Rule

A primitive $2n$-th root of unity mod a prime q exists exactly when $2n$ divides $q - 1$.

Ring	$q - 1$	$2n$	ζ **of order** $2n$**?**
toy	$16 = 2^4$	8	yes, $\zeta = 2$
ML-DSA	$8380416 = 2^{13} \cdot 1023$	512	yes, $\zeta = 1753$
ML-KEM	$3328 = 2^8 \cdot 13$	512	no: only order 256, $\zeta = 17$

- ML-KEM picked a small q to keep keys short, and pays for it with a slightly different NTT.

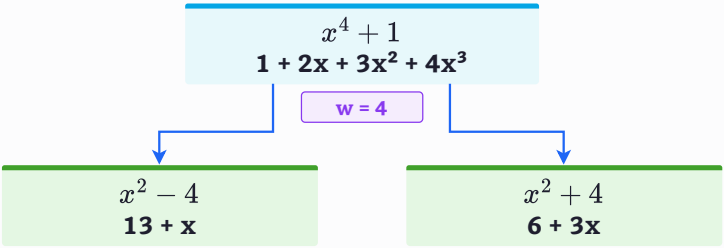
ML-DSA: the full NTT

- $n = 256$, $q = 8380417$, $\zeta = 1753$, with $1753^{256} \equiv -1 \pmod{q}$.
- 8 stages of 128 butterflies, all the way down to single numbers.
- Pointwise product: 256 multiplications, $\hat{c}_i = \hat{a}_i \cdot \hat{b}_i \bmod q$.

Step	**Multiplications**
2 forward NTTs	$2 \cdot 1024 = 2048$
pointwise	256
inverse NTT and scaling	$1024 + 256 = 1280$
total 3,584	(schoolbook: 65,536)

ML-KEM: stop one stage early

- Without a 512th root, the last split cannot happen. ML-KEM runs 7 stages and stops at pieces of degree 1.
- Toy version: stop after stage 1.



- ML-KEM ends with 128 pieces $\hat{a}_{2i} + \hat{a}_{2i+1}x$, each reduced mod $x^2 - \gamma_i$.

Multiplying two degree-1 pieces

BaseCaseMultiply (mod $x^2 - \gamma$)

$$(a_0 + a_1 x)(b_0 + b_1 x) = (a_0 b_0 + \gamma\, a_1 b_1) + (a_0 b_1 + a_1 b_0)\, x$$

Toy, after stage 1: $\hat{a} = (13, 1, 6, 3)$, $\hat{b} = (16, 4, 11, 8)$

$\gamma = 4$: $\quad (13 + x)(16 + 4x) \to (208 + 4 \cdot 4,\ 52 + 16) = (224, 68) \equiv (3, 0)$

$\gamma = -4 \equiv 13$: $\quad (6 + 3x)(11 + 8x) \to (66 + 13 \cdot 24,\ 48 + 33) = (378, 81) \equiv (4, 13)$

- One inverse stage on $(3, 0, 4, 13)$ gives $(7, 13, 4, 1)$. Times $2^{-1} = 9$: $(12, 15, 2, 9)$ again.
- Cost per piece: 5 multiplications.

Multiplying two degree-1 pieces

BaseCaseMultiply (mod $x^2 - \gamma$)

$$(a_0 + a_1 x)(b_0 + b_1 x) = (a_0 b_0 + \gamma\, a_1 b_1) + (a_0 b_1 + a_1 b_0)\, x$$

Toy, after stage 1: $\hat{a} = (13, 1, 6, 3)$, $\hat{b} = (16, 4, 11, 8)$

$\gamma = 4$: $(13 + x)(16 + 4x) \to (208 + 4 \cdot 4,\ 52 + 16) = (224, 68) \equiv (3, 0)$

$\gamma = -4 \equiv 13$: $(6 + 3x)(11 + 8x) \to (66 + 13 \cdot 24,\ 48 + 33) = (378, 81) \equiv (4, 13)$

- One inverse stage on $(3, 0, 4, 13)$ gives $(7, 13, 4, 1)$. Times $2^{-1} = 9$: $(12, 15, 2, 9)$ again.
- Cost per piece: 5 multiplications.

Multiplying two degree-1 pieces

BaseCaseMultiply (mod $x^2 - \gamma$)

$$(a_0 + a_1 x)(b_0 + b_1 x) = (a_0 b_0 + \gamma\, a_1 b_1) + (a_0 b_1 + a_1 b_0)\, x$$

Toy, after stage 1: $\hat{a} = (13, 1, 6, 3)$, $\hat{b} = (16, 4, 11, 8)$

$\gamma = 4$: $\quad (13 + x)(16 + 4x) \to (208 + 4 \cdot 4,\ 52 + 16) = (224, 68) \equiv (3, 0)$

$\gamma = -4 \equiv 13$: $\quad (6 + 3x)(11 + 8x) \to (66 + 13 \cdot 24,\ 48 + 33) = (378, 81) \equiv (4, 13)$

- One inverse stage on $(3, 0, 4, 13)$ gives $(7, 13, 4, 1)$. Times $2^{-1} = 9$: $(12, 15, 2, 9)$ again.
- Cost per piece: 5 multiplications.

ML-KEM in numbers

- $\gamma_i = 17^{2\,\mathrm{brv}(i)+1} \bmod 3329$, a second fixed table: $17, -17, 2761, -2761, \ldots$

Step	**Multiplications**
2 forward NTTs, 7 stages each	$2 \cdot 896 = 1792$
128 base-case products	$128 \cdot 5 = 640$
inverse NTT and scaling	$896 + 256 = 1152$
total 3,584	(schoolbook: 65,536)

- Same total as ML-DSA by coincidence: one stage less, but a more expensive pointwise step.

Stay in the NTT domain

- The schemes avoid converting back and forth:
  - The matrix **A** is random anyway, so it is generated directly as NTT values.
  - The secret **s** is transformed once and reused in all k rows.
  - Only the final results go through the inverse NTT.

ML-KEM-768: computing $\mathbf{A} \cdot \mathbf{s}$

schoolbook: $9 \cdot 65{,}536 = 589{,}824$ multiplications

NTT way: 3 NTTs of $\mathbf{s} + 9 \cdot 640$ base cases $+ 3$ inverse NTTs $= 11{,}904$, about 50 times fewer

Stay in the NTT domain

- The schemes avoid converting back and forth:
  - The matrix **A** is random anyway, so it is generated directly as NTT values.
  - The secret **s** is transformed once and reused in all k rows.
  - Only the final results go through the inverse NTT.

ML-KEM-768: computing $\mathbf{A} \cdot \mathbf{s}$

schoolbook: $9 \cdot 65{,}536 = 589{,}824$ multiplications

NTT way: 3 NTTs of $\mathbf{s} + 9 \cdot 640$ base cases $+ 3$ inverse NTTs $= 11{,}904$, about 50 times fewer

06

Summary

Summary

The NTT in five lines

1. Store a polynomial by its values at the n roots of $x^n + 1$ (the odd powers of ζ).
2. In that form, multiplication in $\mathcal{R}_q$ is pointwise: n multiplications.
3. The fast NTT splits $x^n + 1$ in half $\log_2 n$ times, with $n/2$ butterflies per split.
4. The inverse NTT undoes the splits with Gentleman-Sande butterflies and a final n^{-1}.
5. ML-DSA uses the full NTT. ML-KEM stops one stage early and multiplies degree-1 pieces.

Assignment 7

Build all of this in Python: schoolbook, slow NTT, butterflies, fast NTT and inverse, then ML-DSA size.

Summary

The NTT in five lines

1. Store a polynomial by its values at the n roots of $x^n + 1$ (the odd powers of ζ).
2. In that form, multiplication in $\mathcal{R}_q$ is pointwise: n multiplications.
3. The fast NTT splits $x^n + 1$ in half $\log_2 n$ times, with $n/2$ butterflies per split.
4. The inverse NTT undoes the splits with Gentleman-Sande butterflies and a final n^{-1}.
5. ML-DSA uses the full NTT. ML-KEM stops one stage early and multiplies degree-1 pieces.

Assignment 7

Build all of this in Python: schoolbook, slow NTT, butterflies, fast NTT and inverse, then ML-DSA size.

Now we can afford the arithmetic. Next: ML-KEM and ML-DSA.