

NTTRU: Truly Fast NTRU Using NTT

Vadim Lyubashevsky and **Gregor Seiler**

IBM Research – Zurich

Aug 26, 2019

Motivation

- Symmetric key operations like sampling of random polynomials make up for majority of runtime in many LWE-based schemes
- NTRU schemes require less pseudo-randomness as there is no expansion of uniform public polynomial
- Expensive polynomial inversion during key generation in NTRU schemes is simple when using NTT-based arithmetic

Motivation

- Symmetric key operations like sampling of random polynomials make up for majority of runtime in many LWE-based schemes
- NTRU schemes require less pseudo-randomness as there is no expansion of uniform public polynomial
- Expensive polynomial inversion during key generation in NTRU schemes is simple when using NTT-based arithmetic

Goal: Design extremely *fast* variant of NTRU HRSS around state-of-the-art vectorized NTT arithmetic while maintaining *competitive sizes*

Motivation

- Symmetric key operations like sampling of random polynomials make up for majority of runtime in many LWE-based schemes
- NTRU schemes require less pseudo-randomness as there is no expansion of uniform public polynomial
- Expensive polynomial inversion during key generation in NTRU schemes is simple when using NTT-based arithmetic

Goal: Design extremely *fast* variant of NTRU HRSS around state-of-the-art vectorized NTT arithmetic while maintaining *competitive sizes*

This requires non-power-of-two NTT and non-fully-splitting prime modulus

Cyclic NTT

Let $\zeta \in \mathbb{Z}_q$ be a primitive n -th root of unity, i.e. $\zeta^n = 1$ but $\zeta^k \neq 1$ for $0 < k < n$.

For $f \in \mathbb{Z}_q[X]/(X^n - 1)$,

$$\text{NTT}(f) = (f(\zeta^0), f(\zeta^1), \dots, f(\zeta^{n-1})) \in \mathbb{Z}_q^n$$

defines an isomorphism

In particular, polynomial multiplication/division in $\mathbb{Z}_q[X]/(X^n - 1)$ translates to coefficientwise multiplication/division in \mathbb{Z}_q^n

Negacyclic NTT

We want an irreducible defining polynomial φ for our ring $\mathcal{R} = \mathbb{Z}_q[X]/(\varphi)$

If $n = 2^k$, then $X^n + 1$ is the irreducible $2n$ -th cyclotomic polynomial

Some schemes compute twisting map

$$\mathbb{Z}_q[X]/(X^n + 1) \xrightarrow{X \mapsto \zeta X} \mathbb{Z}_q[X]/(X^n - 1)$$

and then use the cyclic NTT. This is slightly non-optimal.

Algebraic Formulation of NTT

If there exists primitive n -th root of unity ζ in \mathbb{Z}_q , then

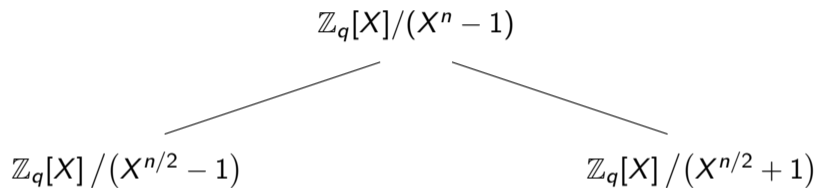
$$X^n - 1 = (X - 1)(X - \zeta) \cdots (X - \zeta^{n-1})$$

Now, by the Chinese remainder theorem,

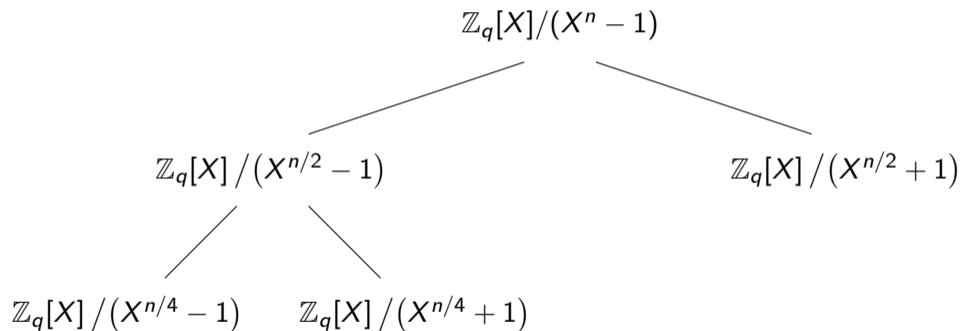
$$\mathbb{Z}_q[X]/(X^n - 1) \cong \mathbb{Z}_q[X]/(X - 1) \times \cdots \times \mathbb{Z}_q[X]/(X - \zeta^{n-1})$$

The NTT is this CRT map

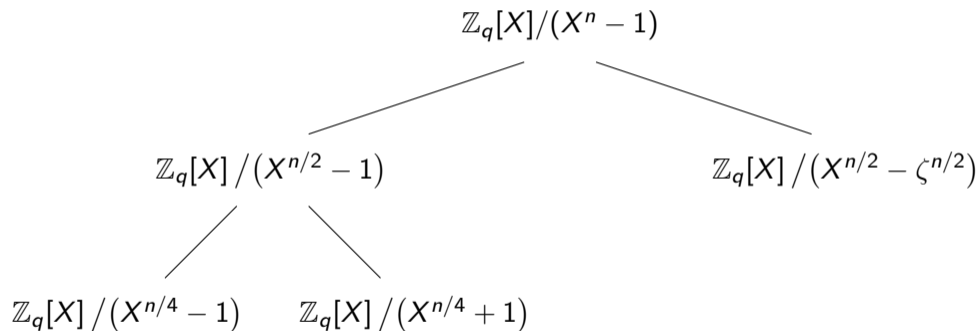
Recursive Algorithm



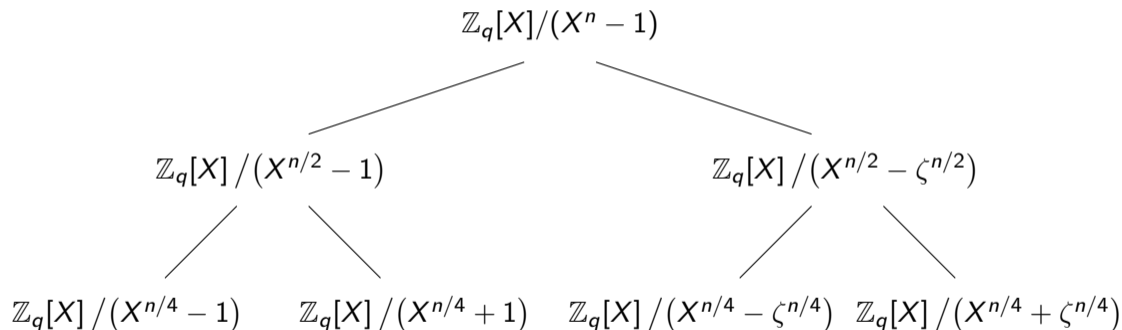
Recursive Algorithm



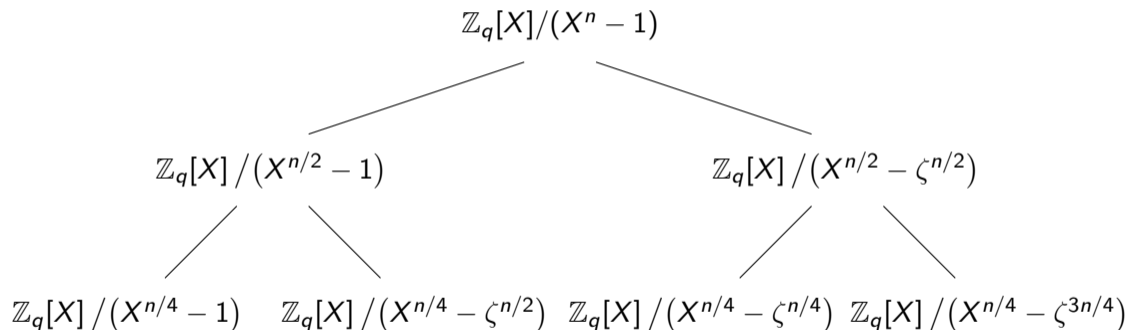
Recursive Algorithm



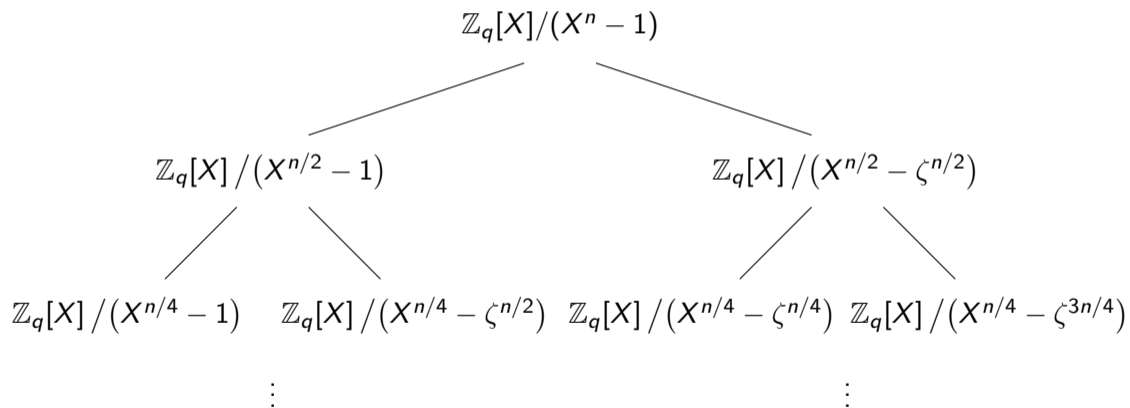
Recursive Algorithm



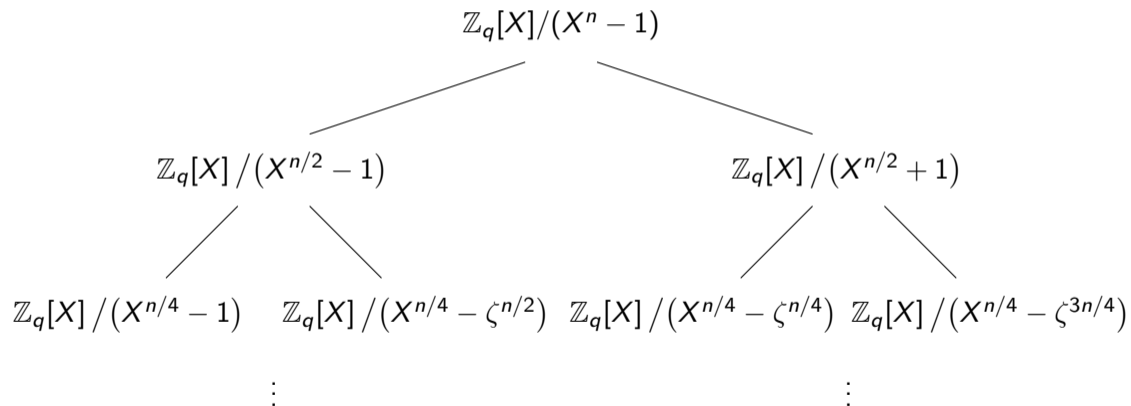
Recursive Algorithm



Recursive Algorithm



Recursive Algorithm



Our NTT of Length 768

Let $\mathcal{R} = \mathbb{Z}_{7681}[X]/(X^{768} - X^{384} + 1)$ and $\zeta \in \mathbb{Z}_{7681}$ be a primitive 768-th root of unity

We want to compute

$$\mathbb{Z}_{7681}[X]/(X^{768} - X^{384} + 1) \cong \mathbb{Z}_{7681}[X]/(X^3 - \zeta) \times \cdots \times \mathbb{Z}_{7681}[X]/(X^3 - \zeta^{767})$$

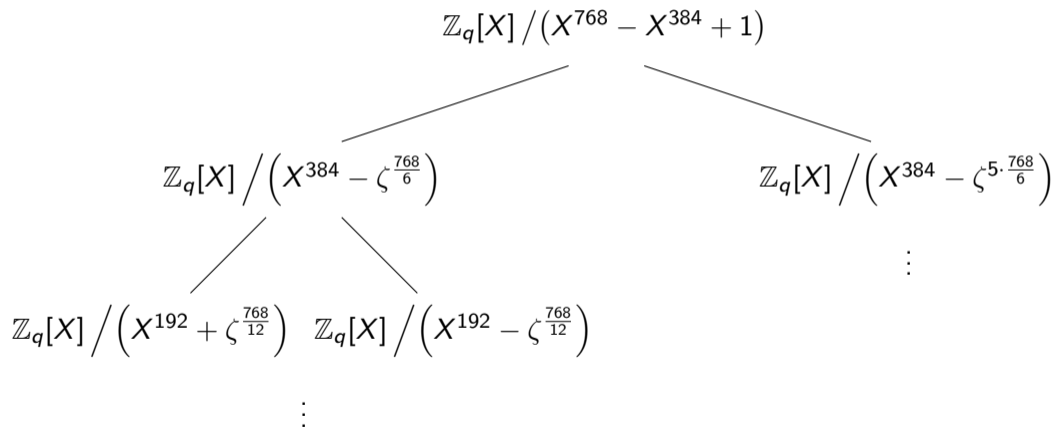
Splitting Strategy

$$\mathbb{Z}_q[X] / (X^{768} - X^{384} + 1)$$

Splitting Strategy

$$\begin{array}{ccc} & \mathbb{Z}_q[X] / (X^{768} - X^{384} + 1) & \\ & \swarrow \quad \searrow & \\ \mathbb{Z}_q[X] / (X^{384} - \zeta^{\frac{768}{6}}) & & \mathbb{Z}_q[X] / (X^{384} - \zeta^{5 \cdot \frac{768}{6}}) \end{array}$$

Splitting Strategy



Splitting Strategy

$$\begin{array}{ccc} & \mathbb{Z}_q[X] / (X^{768} - X^{384} + 1) & \\ & \swarrow \quad \searrow & \\ \mathbb{Z}_q[X] / (X^{384} - \zeta^{\frac{768}{6}}) & & \mathbb{Z}_q[X] / (X^{384} - \zeta^{5 \cdot \frac{768}{6}}) \end{array}$$

Observe: $\zeta^{\frac{768}{6}}$ is a root of $X^2 - X + 1$. Hence $\zeta^{5 \cdot \frac{768}{6}} = 1 - \zeta^{\frac{768}{6}}$.

Splitting Strategy

$$\begin{array}{ccc} & \mathbb{Z}_q[X] / (X^{768} - X^{384} + 1) & \\ & \swarrow \quad \searrow & \\ \mathbb{Z}_q[X] / (X^{384} - \zeta^{\frac{768}{6}}) & & \mathbb{Z}_q[X] / (X^{384} + (\zeta^{\frac{768}{6}} - 1)) \end{array}$$

Observe: $\zeta^{\frac{768}{6}}$ is a root of $X^2 - X + 1$. Hence $\zeta^{5 \cdot \frac{768}{6}} = 1 - \zeta^{\frac{768}{6}}$.

Vectorized Implementations of the NTT

Two possibilities to vectorize products with roots of unity on AVX2

- 1 Pack only eight 16 bit coefficients in 256 bit registers and leave room for intermediate 32 bit products using instruction *vpmulld*
- 2 Densely pack sixteen 16 bit coefficients in 256 bit registers and compute separate low and high parts of 32-bit products using instructions *vpmullw* and *vpmulhw*

We use second approach with a variant of the Montgomery reduction algorithm that naturally handles this representation

Signed Montgomery Reduction

Hensel remainder of c modulo q : Unique r such that

$$c = mq + r2^{16}$$

We have $r \equiv c2^{-16} \pmod{q}$

Algorithm:

- 1 Multiply c by q^{-1} modulo 2^{16} ; gives m
- 2 Multiply m by q and subtract from c ; gives $r2^{16}$
- 3 Divide by 2^{16} (shift right); gives r

Fast Mulmod

For product $c = ab = mq + r2^{16}$ compute

$$c = c_0 + c_12^{16}$$

- 1 Multiply c by q^{-1} modulo 2^{16} ; gives m
Need only low word c_0 of c
- 2 Multiply m by q and subtract from c ; gives $r2^{16}$
 mq and c have equal low word; Sufficient to compute only high word of mq and subtract from high word c_1 of c ; This already gives r

Further Optimization: If b is precomputable constant, can also precompute $bq^{-1} \bmod 2^{16}$ and skip first reduction step

Full mulmod in \mathbb{Z}_q with precomputed constant costing only three half products!

Results

Bytes pk/ct	Cycles Key generation	Cycles Signing	Cycles Verification
1248	6431	6101	7878

Measurements performed on Intel Skylake Core i7-6600U CPU

Possible Tweaks

- Use prime modulus $q = 3457$ instead of 7681
Would result in about the same sizes as NTRU HRSS

Possible Tweaks

- Use prime modulus $q = 3457$ instead of 7681
Would result in about the same sizes as NTRU HRSS
- Deterministic noise