Randomisation de la NTT pour les cryptosystèmes basés sur RLWE pour contrer les attaques par canaux cachés basées sur la Belief Propagation

Christophe Negre, Mbaye Ngom
Univ. Perpignan, France

RAIM 2024 - 4-6-th November 2024

Outline

1. RLWE cryptosystem and NTT
2. Belief propagation on NTT
3. Counter-measures
4. Simulation results and conclusion

Outline

Notations and RLWE problem

Notations

- p a prime integer.
- $n = 2^t$.
- $\mathbb{Z}_p = \mathbb{Z}/p\mathbb{Z}$.
- $\mathcal{R}_p = \mathbb{Z}_p[X]/(X^n - 1)$.
- $A \overset{\mathcal{U}}{\leftarrow} \mathcal{R}_p$ uniform distribution.
- $E \overset{\chi}{\leftarrow} \mathcal{R}_p$ distribution with small coefficients.

RLWE problem.

$A \overset{\mathcal{U}}{\leftarrow} \mathcal{R}_p$ and $S, E \overset{\chi}{\leftarrow} \mathcal{R}_p$ find S from:

$$A \text{ and } B = A \times S + E$$

RLWE cryptosystem

Alice → Bob

$$Pk = (A, B)$$

Key Generation

$$A \xleftarrow{\mathcal{U}} \mathcal{R}_p \text{ and } S, E \xleftarrow{\chi} \mathcal{R}_p$$

$$B = A \times S + E$$

Encryption

Plaintext $\mathrm{m} \in \mathcal{R}_p$

$$C_1 = A \times E_1 + E_2,$$

$$C_2 = B \times E_1 + E_3 + m$$

Decryption

$$\mathrm{m} = Round(C_2 - C_1 \times \mathbf{S}) \longleftarrow (C_1, C_2)$$

Multiplication in $\mathcal{R}_p$ with NTT/FFT

We recall $\mathcal{R}_p = \mathbb{Z}_p[X]/(X^n - 1)$ and ω is a primitive n-th root of unity. We have

$$(X^n - 1) = (X - \omega^0)(X - \omega^1)\cdots(X - \omega^{n-1})$$

$$\mathcal{R}_p \cong \mathbb{Z}_p[X]/(X - \omega^0) \times \mathbb{Z}_p[X]/(X - \omega^1) \times \cdots \times \mathbb{Z}_p[X]/(X - \omega^{n-1})$$

- The multiplication of F and G in $\mathcal{R}_p$ can be done as follows

$$\begin{array}{rcl}
F(X) & \xrightarrow{evaluation} & \hat{F} = (F(\omega^0), F(\omega^1), \ldots, F(\omega^{n-1})) \\
G(X) & \xrightarrow{evaluation} & \hat{G} = (G(\omega^0), G(\omega^1), \ldots, G(\omega^{n-1})) \\
 & & \downarrow \\
R(X) = F(X) \times G(X) & \xleftarrow{interpolation} & (\hat{f}_0 \times \hat{g}_0, \hat{f}_1 \times \hat{g}_1, \ldots, \hat{f}_{n-1} \times \hat{g}_{n-1})
\end{array}$$

FFT/NTT - Odd-even splitting

$$F(X) = \underbrace{\sum_{k=0}^{n/2} f_{2k} X^{2k}}_{F_e(X^2)} + X \times \underbrace{\sum_{k=0}^{n/2} f_{2k+1} X^{2k}}_{F_o(X^2)}$$

FFT/NTT - Odd-even splitting

$$F(X) = \underbrace{\sum_{k=0}^{n/2} f_{2k} X^{2k}}_{F_e(X^2)} + X \times \underbrace{\sum_{k=0}^{n/2} f_{2k+1} X^{2k}}_{F_o(X^2)}$$

For $j = 0, \ldots, n/2 - 1$

$$\begin{aligned} F(\omega^j) &= F_e(\omega^{2j}) + \omega^j F_o(\omega^{2j}) \\ F(\omega^{n/2+j}) &= F_e(\omega^{2j}) - \omega^j F_o(\omega^{2j}) \end{aligned}$$

FFT/NTT - Odd-even splitting

$$F(X) = \underbrace{\sum_{k=0}^{n/2} f_{2k} X^{2k}}_{F_e(X^2)} + X \times \underbrace{\sum_{k=0}^{n/2} f_{2k+1} X^{2k}}_{F_o(X^2)}$$

For $j = 0, \ldots, n/2 - 1$

$$\begin{aligned} F(\omega^j) &= F_e(\omega^{2j}) + \omega^j F_o(\omega^{2j}) \\ F(\omega^{n/2+j}) &= F_e(\omega^{2j}) - \omega^j F_o(\omega^{2j}) \end{aligned}$$

NTT(F, ω, n)

if $n = 1$ **then**
  return $\widehat{F} = [f_0]$
  $F_e = \sum_{k=0}^{\frac{n}{2}-1} f_{2k} X^k$
  $F_o = \sum_{k=0}^{\frac{n}{2}-1} f_{2k+1} X^k$

Recursion

$\widehat{R} \longleftarrow NTT(F_e, \omega', n/2)$
$\widehat{R'} \longleftarrow NTT(F_o, \omega', n/2)$

for $i = 0; i < n/2;$i++ **do**
  $\widehat{F}_i = \widehat{r}_i + \omega^i \widehat{r}_i'$
  $\widehat{F}_{i+n/2} = \widehat{r}_i - \omega^i \widehat{r}_i'$

return $\widehat{F}$

FFT/NTT - High-Low splitting

- $F(\omega^{2j}) = \sum_{k=0}^{\frac{n}{2}-1} (f_k + f_{n/2+k})(\omega^{2j})^k$
- $F(\omega^{2j+1}) = \sum_{k=0}^{\frac{n}{2}-1} (f_k - f_{n/2+k})\omega^k(\omega^{2j})^k$

NTT(F, ω, n)

if $n = 1$ **then**
  return $\widehat{F} = [f_0]$
  $\omega' = \omega^2$
  $R = \sum_{k=0}^{\frac{n}{2}-1} (f_k + f_{n/2+k})X^k$
  $R' = \sum_{k=0}^{\frac{n}{2}-1} (f_k - f_{n/2+k})\omega^k X^k$

  Recursion

  $\widehat{R} \longleftarrow NTT(R, \omega', n/2)$
  $\widehat{R'} \longleftarrow NTT(R', \omega', n/2)$

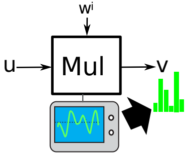
return $\widehat{F} = (\widehat{r}_0, \widehat{r}'_0, \cdots, \widehat{r}_{n/2-1}, \widehat{r}'_{n/2-1})$

Outline

1. RLWE cryptosystem and NTT
2. Belief propagation on NTT
3. Counter-measures
4. Simulation results and conclusion

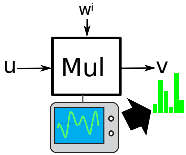
Belief propagation : leaking model and propagation

Leackage : Power consumption provides probabilities on processed values:

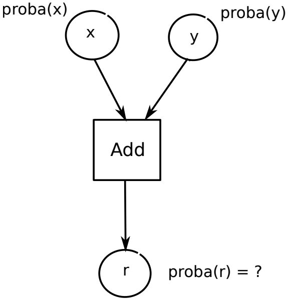


Belief propagation : leaking model and propagation

Leackage : Power consumption provides probabilities on processed values:



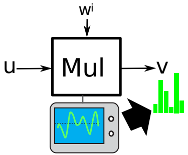
Belief propagation propagates probabilities as follows:



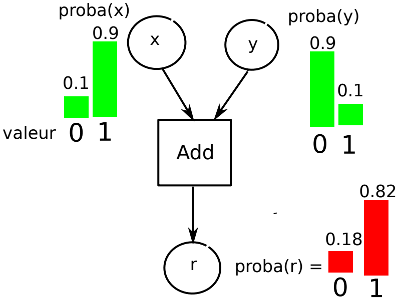
$$proba(r) = \sum_{x+y=r} proba(x) prob(y)$$

Belief propagation : leaking model and propagation

Leackage : Power consumption provides probabilities on processed values:

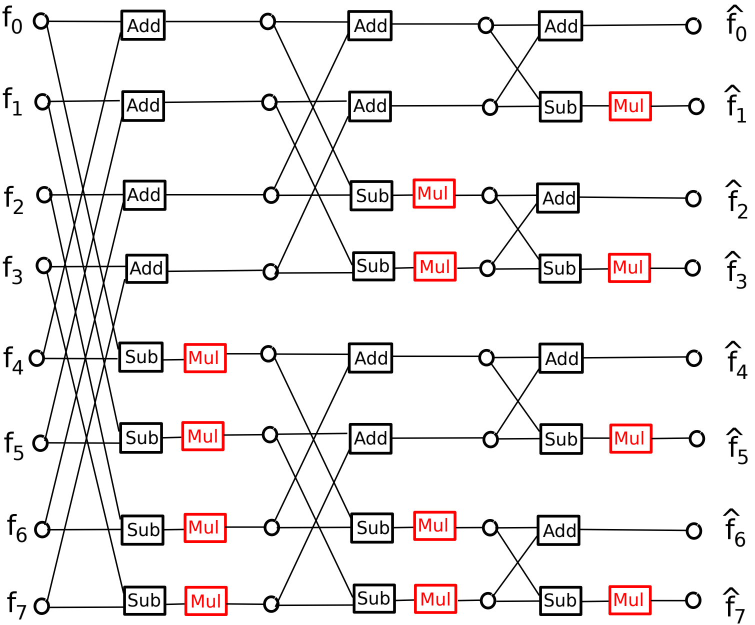


Belief propagation propagates probabilities as follows:

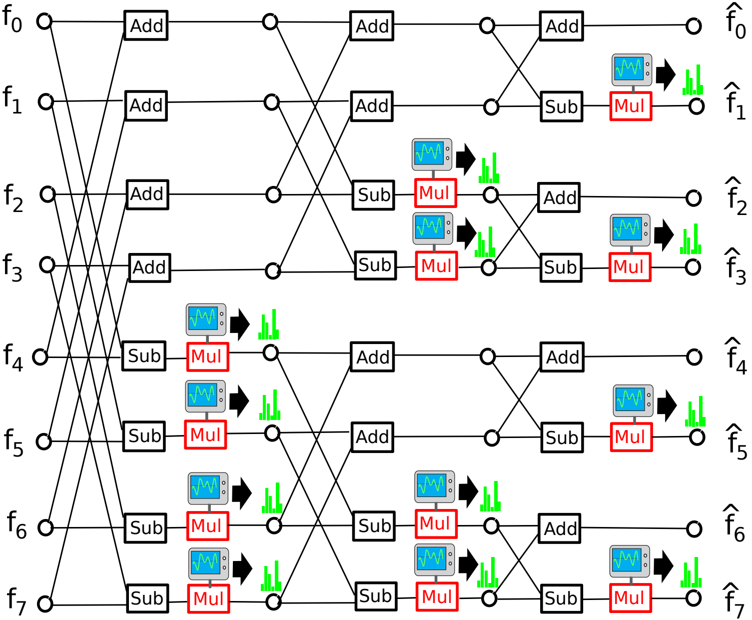


$$proba(r) = \sum_{x+y=r} proba(x)prob(y)$$

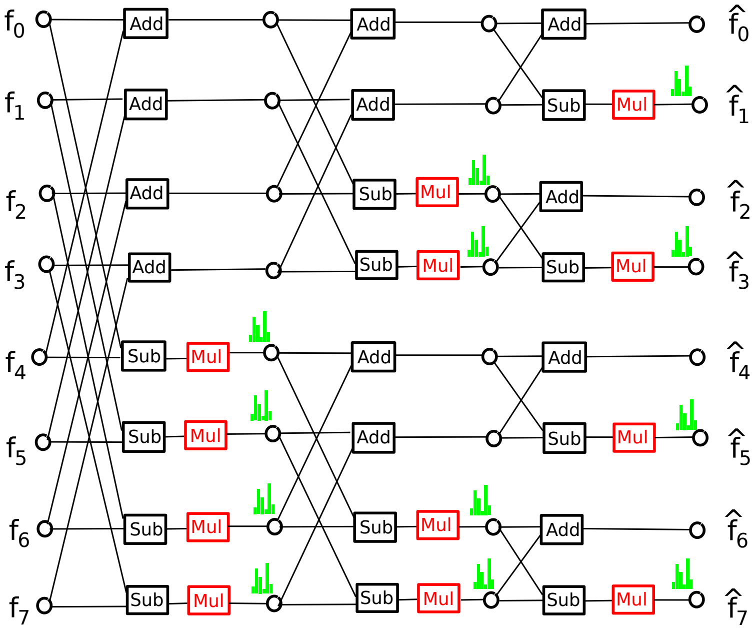
Belief-propagation on NTT



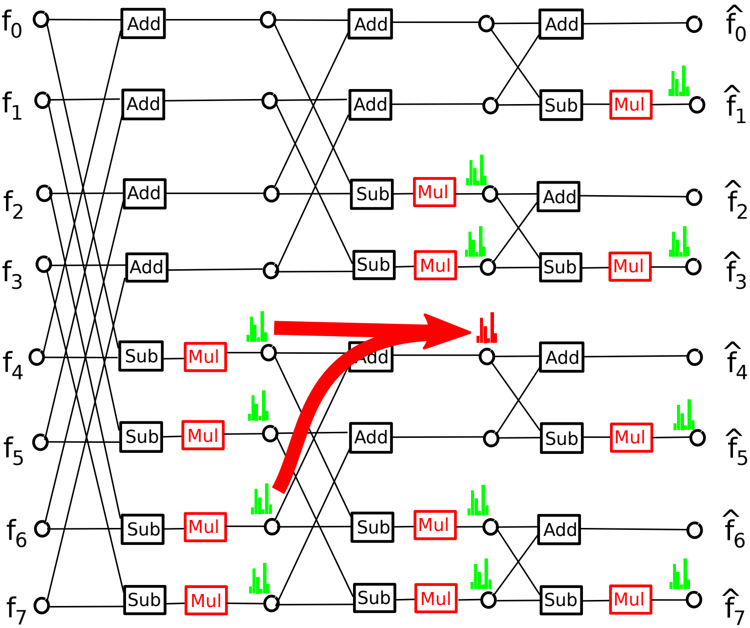
Belief-propagation on NTT



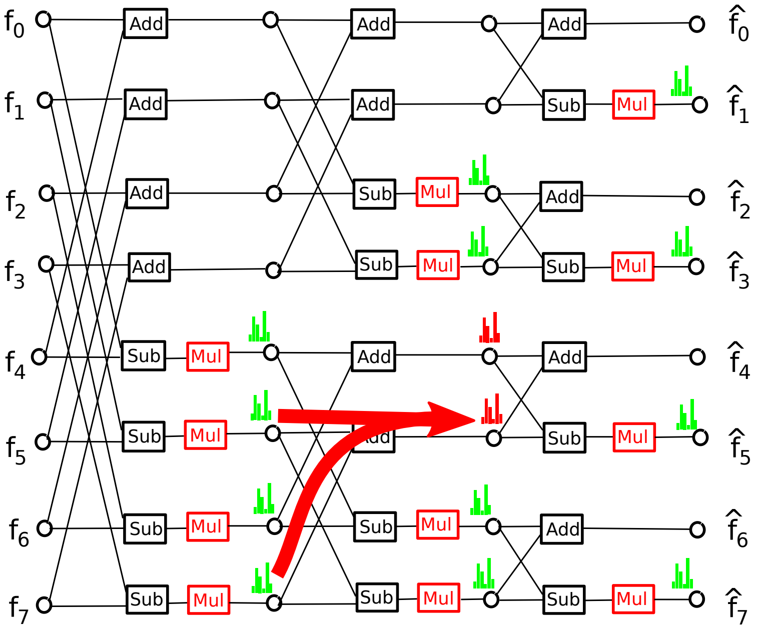
Belief-propagation on NTT



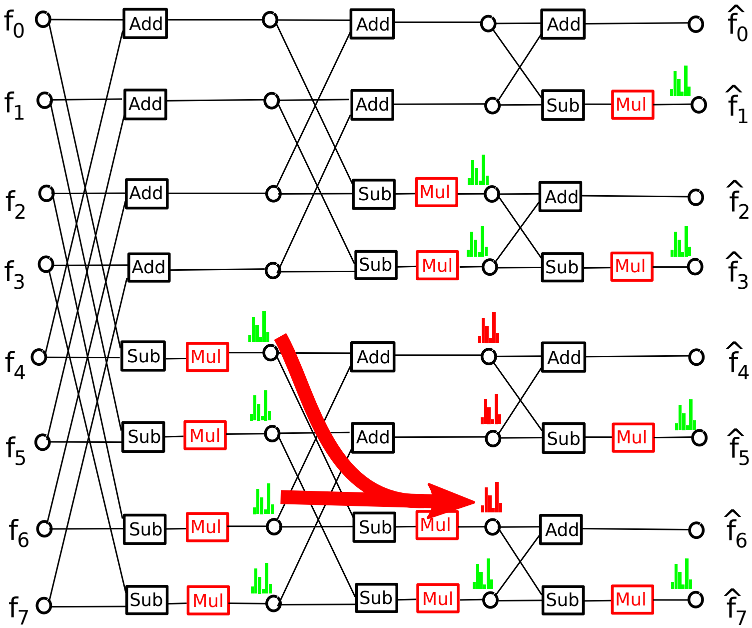
Belief-propagation on NTT



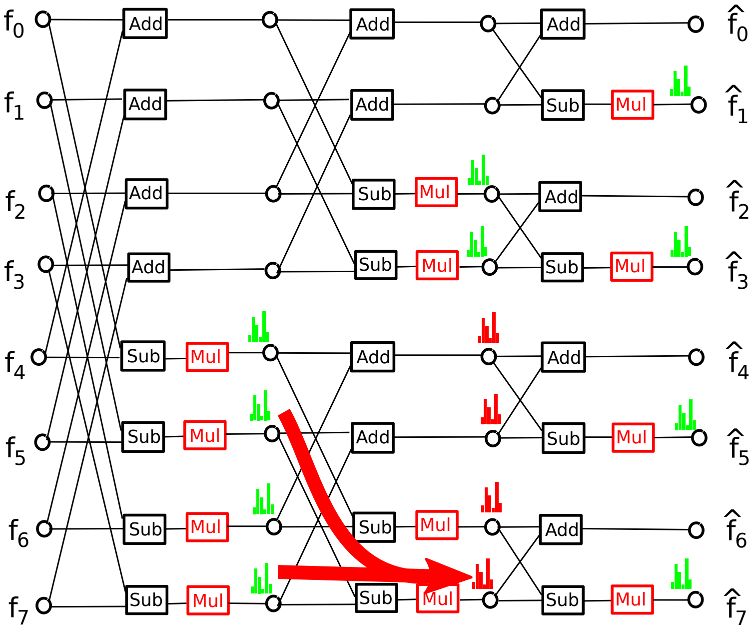
Belief-propagation on NTT



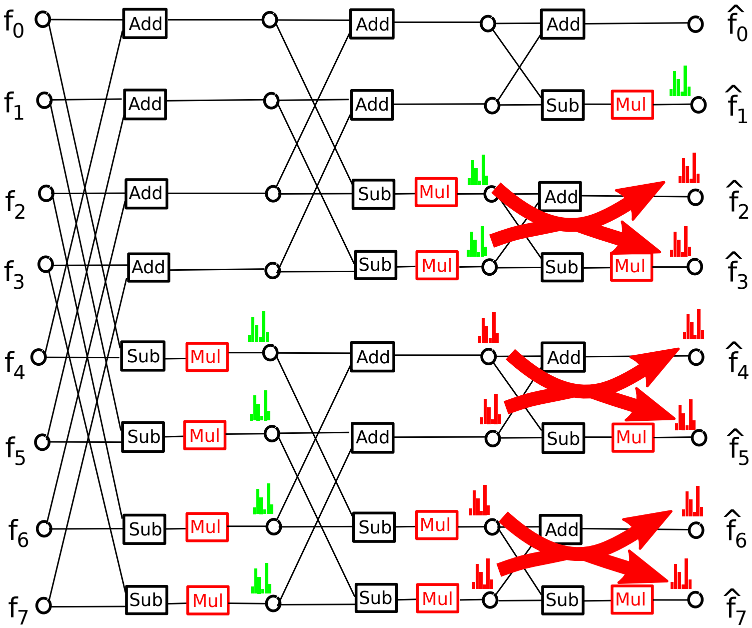
Belief-propagation on NTT



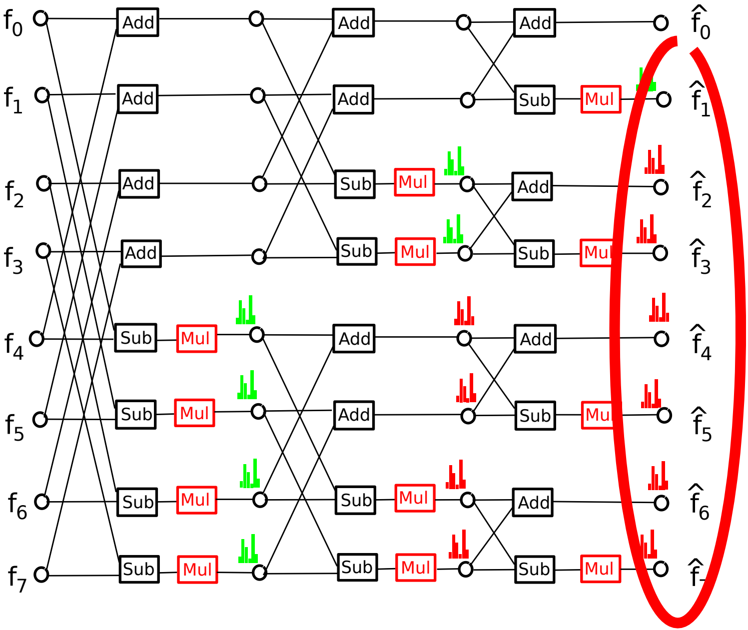
Belief-propagation on NTT



Belief-propagation on NTT



Belief-propagation on NTT



Outline

NTT randomisation : State of the art

Multiplicative mask	$\mathbf{f}' = \alpha \times \mathbf{f}$ with a random $\alpha \in \mathbb{Z}_q$ $\widehat{\mathbf{f}'} = NTT(\mathbf{f}')$ $\widehat{\mathbf{f}} = \alpha^{-1} \times \widehat{\mathbf{f}'}$
Additive mask	random split $\mathbf{f} = \mathbf{f}' + \mathbf{f}''$ $\widehat{\mathbf{f}'} = NTT(\mathbf{f}')$ $\widehat{\mathbf{f}''} = NTT(\mathbf{f}'')$ $\widehat{\mathbf{f}} = \widehat{\mathbf{f}'} + \widehat{\mathbf{f}''}$
Shifting	$\mathbf{f}' = X^r \times \mathbf{f} \mod (X^n - 1)$ $\widehat{\mathbf{f}'} = NTT(\mathbf{f}')$ $\widehat{\mathbf{f}} = (\omega^{-ir}\widehat{\mathbf{f}'}_i)_{i=0,\ldots,n-1}$
Shuffling	At each level of NTT, perform butterfly operations in random order

Proposed (virtually free) randomisation

- Randomisation by mixing HL and OE NTT.
- Randomising the root of unity ω.
- Randomising with multiplicative mask ω^i.
- Random reduction (Barrett/Montgomery).

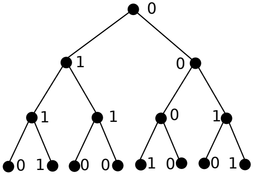
Randomization mixing High-Low and Odd-Even NTT (1/2)

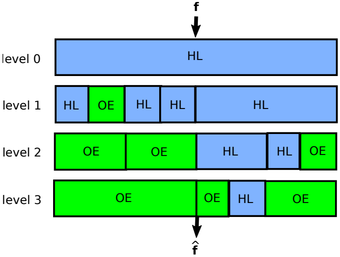
NTT-RandHLOE(F, ω, n)

if $n = 1$ **then**
  return $\widehat{F} = [f_0]$
else
  $r \xleftarrow{\mathcal{U}} \{0, 1\}$
  if $r = 0$ **then**
    // Apply High-Low splitting formula
    // and recursion
    ...
  else
    // Apply Odd-Even splitting formula
    // and recursion
    ...

Randomization mixing High-Low and Odd-Even NTT (2/2)

Random bits. (with $n = 16 = 2^4$)





Level of randomisation: $n/2$

NTT with random reduction (1/2)

p is an $\ell - 1$ bit integer.
$p' = p^{-1} \pmod{2^\ell}$

Montgomery Multiplication (MM)

1: $z \leftarrow x \times y$
2: $s \leftarrow p^{-1} \times z \pmod{2^\ell}$
3: $r \leftarrow (z - s \times p)/2^\ell$

$$r \equiv (x \times y) \times 2^{-\ell} \mod p$$

NTT with random reduction (1/2)

p is an $\ell - 1$ bit integer.
$p' = p^{-1} \pmod{2^\ell}$

Montgomery Multiplication (MM)

1: $z \leftarrow x \times y$
2: $s \leftarrow p^{-1} \times z \pmod{2^\ell}$
3: $r \leftarrow (z - s \times p)/2^\ell$

$r \equiv (x \times y) \times 2^{-\ell} \mod p$

$p' = \lfloor 2^{2\ell}/p \rfloor$

Barrett Multiplication (BM)

1: $z \leftarrow x \times y$
2: $s \leftarrow \lfloor \lfloor z/2^{\ell-1} \rfloor p'/2^{\ell+1} \rfloor$
3: $r \leftarrow z - s \times p$

$r \equiv (x \times y) \mod p$

NTT with random reduction (2/2)

NTT-RandRed(F, ω, n)

if $n = 1$ **then**
  return $\widehat{F} = [f_0]$
else
  $r \xleftarrow{\mathcal{U}} \{0, 1\}$
  if $r = 0$ **then**
    // High-Low butterfly with BM
    // and recursion
    ...
  else
    // High-Low butterfly with MM
    // and recursion
    ...

- Level of random ($n = 2^t$) :

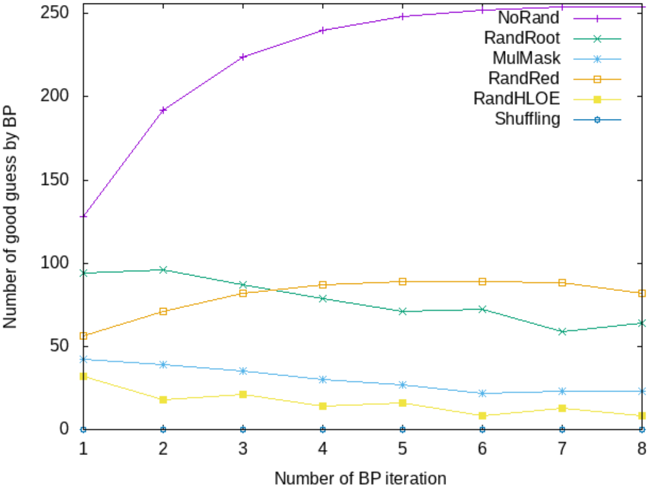
$$1+2+2^2+\ldots+2^{t-2} = \frac{n}{2} - 1$$

- Multiplicative masks are "small" :

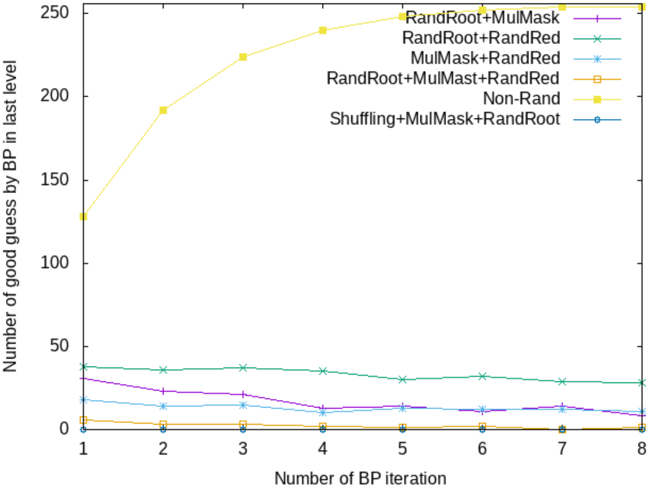
$$\hat{f}_j \times (2^{-\ell})^i, i \in \{0, \ldots, t-2\}$$

Outline

Simulation results : only the last level ($n = 256$)



Simulation results : combined randomisation ($n = 256$)



Conclusion

Level of randomization:

	Formula	$n = 256$
Shuffling	$(\frac{n}{2}!)^{\log_2(n)}$	2^{5729}
RandHLOE	$n/2$	2^7
RandRoot	$\sim \frac{2^n}{n}$	2^{247}
RandMulMask	$\sim n^{n/2}$	2^{1024}
RandRed	$n/2$	2^7

Simulation results show that:

- All tested randomisation are effective against BP.
- Combined randomisations are better.

Some remaining questions:

- Is any brute-force or trade-off attack on randomisation effective for BP ?
- The effect of randomization on the first phase of the attack: template attack on modular multiplication ?

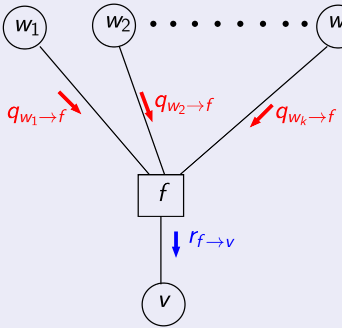
Thank you for you attention

Any question ?

Belief propagation - general setting

Graph of factor nodes □ and variable nodes ○.

Information : factor to variable

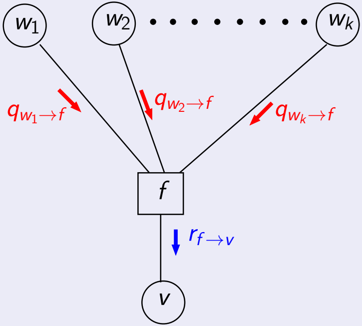


$$r_{f \to v}(x) = \sum_{x_i} f(x, x_1, .., x_k) \prod_{i=1}^{k} q_{v_i \to f}(x_i)$$

Belief propagation - general setting

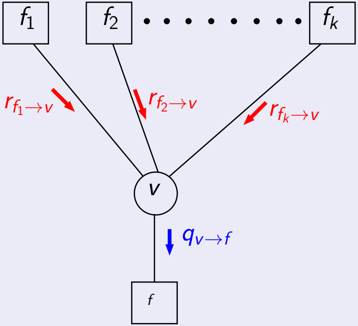
Graph of factor nodes □ and variable nodes ○.

Information : factor to variable



$$r_{f\to v}(x) = \sum_{x_i} f(x, x_1, .., x_k) \prod_{i=1}^{k} q_{v_i\to f}(x_i)$$

Information : variable to factor



$$q_{v\to f}(x) = \prod_{i=1}^{k} r_{f_i\to v}(x)$$

Toy example : BP on NTT for $n = 16$ and $q = 257$

- G = Good. The value with highest probability corresponds to real value computed.

Belief Probagation iteration 1 :

init	**B-B-B-B-B-B-B-B-B-B-B-B-B-B-B-B**
level 1	**B-B-B-B-B-B-B-B-G-G-G-G-G-G-G-G**
level 2	**B-B-B-B-G-G-G-G-B-B-B-B-G-G-G-G**
level 3	**B-B-G-G-B-B-G-G-B-B-G-G-B-B-G-G**
level 4	**B-B-B-B-B-B-B-B-B-B-B-B-B-B-B-B**

Toy example : BP on NTT for $n = 16$ and $q = 257$

- G = Good. The value with highest probability corresponds to real value computed.

Belief Probagation iteration 1 :

init	B-B-B-B-B-B-B-B-B-B-B-B-B-B-B-B
level 1	B-B-B-B-B-B-B-B-G-G-G-G-G-G-G-G
level 2	B-B-B-B-G-G-G-G-B-B-B-B-G-G-G-G
level 3	B-B-G-G-B-B-G-G-B-B-G-G-B-B-G-G
level 4	B-B-B-B-B-B-B-B-B-B-B-B-B-B-B-B

After two iterations of BP:

init	B-B-B-B-B-B-B-B-B-B-B-B-B-B-B-B
level 1	B-B-B-B-B-B-B-B-G-G-G-G-G-G-G-G
level 2	B-B-B-B-G-G-G-G-G-G-G-G-G-G-G-G
level 3	B-B-G-G-G-G-G-G-B-B-G-G-G-G-G-G
level 4	B-B-G-G-B-B-G-G-B-B-G-G-B-B-G-G

Toy example : BP on NTT for $n = 16$ and $q = 257$

- G = Good. The value with highest probability corresponds to real value computed.

Belief Probagation iteration 1 :

init	B-B-B-B-B-B-B-B-B-B-B-B-B-B-B-B
level 1	B-B-B-B-B-B-B-B-G-G-G-G-G-G-G-G
level 2	B-B-B-B-G-G-G-G-B-B-B-B-G-G-G-G
level 3	B-B-G-G-B-B-G-G-B-B-G-G-B-B-G-G
level 4	B-B-B-B-B-B-B-B-B-B-B-B-B-B-B-B

After two iterations of BP:

init	B-B-B-B-B-B-B-B-B-B-B-B-B-B-B-B
level 1	B-B-B-B-B-B-B-B-G-G-G-G-G-G-G-G
level 2	B-B-B-B-G-G-G-G-G-G-G-G-G-G-G-G
level 3	B-B-G-G-G-G-G-G-B-B-G-G-G-G-G-G
level 4	B-B-G-G-B-B-G-G-B-B-G-G-B-B-G-G

After four iterations of BP:

init	B-B-B-B-B-B-B-B-B-B-B-B-B-B-B-B
level 1	B-B-B-B-B-B-B-B-G-G-G-G-G-G-G-G
level 2	B-B-B-B-G-G-G-G-G-G-G-G-G-G-G-G
level 3	B-B-G-G-G-G-G-G-G-G-G-G-G-G-G-G
level 4	B-B-G-G-G-G-G-G-G-G-G-G-G-G-G-G

Afterwards, there is no improvement.