Implementing Fast Carryless Multiplication

Joris van der Hoeven, Robin Larrieu and Grégoire Lecerf

CNRS & École polytechnique

MACIS 2017
Nov. 15, Vienna, Austria

Outline

Introduction
- Carryless multiplication
- State of the art

Presentation of the algorithm

Implementation details

Perspectives

Carryless multiplication

Multiplication in $\mathbb{F}_2[X]$, large degree (typically $\geqslant 10^6$).
Fast algorithms for such sizes use FFT multiplication.

Carryless multiplication

Multiplication in $\mathbb{F}_2[X]$, large degree (typically $\geqslant 10^6$).
Fast algorithms for such sizes use FFT multiplication.

Problem

Not many evaluation points in $\mathbb{F}_2 \Rightarrow$ work in an extension field.
How to minimize the corresponding overhead?

State of the art

1. Triadic Schönhage-Strassen algorithm (GF2X – Brent, Gaudry, Thomé, Zimmermann)

State of the art

1. Triadic Schönhage-Strassen algorithm (GF2X – Brent, Gaudry, Thomé, Zimmermann)
2. FFT over $\mathbb{F}_{2^{60}}$ (Harvey, van der Hoeven, Lecerf – 2016)

State of the art

1. Triadic Schönhage-Strassen algorithm (GF2X – Brent, Gaudry, Thomé, Zimmermann)
2. FFT over $\mathbb{F}_{2^{60}}$ (Harvey, van der Hoeven, Lecerf – 2016)
3. Additive FFT over $\mathbb{F}_{2^{128}}$ or $\mathbb{F}_{2^{256}}$ (Chen, Cheng, Kuo, Li, Yang – 2017)

State of the art

1. Triadic Schönhage-Strassen algorithm (GF2X – Brent, Gaudry, Thomé, Zimmermann)
2. FFT over $\mathbb{F}_{2^{60}}$ (Harvey, van der Hoeven, Lecerf – 2016)
3. Additive FFT over $\mathbb{F}_{2^{128}}$ or $\mathbb{F}_{2^{256}}$ (Chen, Cheng, Kuo, Li, Yang – 2017)

This work[1]

Improvement of strategy n.° 2 using the ideas from the *Frobenius FFT* algorithm (van der Hoeven, Larrieu – 2017). Achieves a speedup by a factor 2.

[1] Source code available from revision 10681 of our SVN server (`https://gforge.inria.fr/projects/mmx/`) in the JUSTINLINE library

Why $\mathbb{F}_{2^{60}}$?

Efficient arithmetic in $\mathbb{F}_{2^{60}}$

- Slightly smaller than a machine word
- $\mu(X) := \frac{X^{61}-1}{X-1}$ irreducible over $\mathbb{F}_2$

Efficient FFT

Roots of unity with order

$$2^{60} - 1 = 3^2 \times 5^2 \times 7 \times 11 \times 13 \times 31 \times 41 \times 61 \times 151 \times 1321$$

Why $\mathbb{F}_{2^{60}}$?

Efficient arithmetic in $\mathbb{F}_{2^{60}}$

- Slightly smaller than a machine word
- $\mu(X) := \frac{X^{61}-1}{X-1}$ irreducible over $\mathbb{F}_2$

Efficient FFT

Roots of unity with order

$$2^{60} - 1 = 3^2 \times 5^2 \times 7 \times 11 \times 13 \times 31 \times 41 \times 61 \times 151 \times 1321$$

Bonus

- 61 divides $2^{60} - 1$. (Fermat's theorem)
- 2 generates $(\mathbb{Z}/61\mathbb{Z})^\times$ ($\Leftrightarrow \mu(X)$ irreducible)

Outline

Introduction

Presentation of the algorithm

- Kronecker segmentation vs. Frobenius FFT
- Our variant of the Frobenius FFT
- Frobenius encoding

Implementation details

Perspectives

Kronecker segmentation vs. Frobenius FFT

Naive strategy $\mathbb{F}_2[X]_{<n} \hookrightarrow \mathbb{F}_{2^{60}}[X]_{<n}$

$$\begin{array}{l} A \in \mathbb{F}_2[X] \longrightarrow \tilde{A} \in \mathbb{F}_{2^{60}}[X] \\ B \in \mathbb{F}_2[X] \longrightarrow \tilde{B} \in \mathbb{F}_{2^{60}}[X] \end{array} \longrightarrow \tilde{AB} \in \mathbb{F}_{2^{60}}[X] \longrightarrow AB \in \mathbb{F}_2[X]$$

Kronecker segmentation vs. Frobenius FFT

Naive strategy $\mathbb{F}_2[X]_{<n} \hookrightarrow \mathbb{F}_{2^{60}}[X]_{<n}$

$$\begin{array}{l} A \in \mathbb{F}_2[X] \longrightarrow \tilde{A} \in \mathbb{F}_{2^{60}}[X] \\ B \in \mathbb{F}_2[X] \longrightarrow \tilde{B} \in \mathbb{F}_{2^{60}}[X] \end{array} \begin{array}{l} \longrightarrow \\ \longrightarrow \end{array} \tilde{AB} \in \mathbb{F}_{2^{60}}[X] \longrightarrow AB \in \mathbb{F}_2[X]$$

Kronecker segmentation $\mathbb{F}_2[X]_{<n} \hookrightarrow \mathbb{F}_2[X]_{<30}[Z]_{<n/30} \hookrightarrow \mathbb{F}_{2^{60}}[Z]_{<n/30}$

$$\begin{array}{l} A \in \mathbb{F}_2[X] \longrightarrow \tilde{A} \in \mathbb{F}_2[X]_{<30}[Z] \\ B \in \mathbb{F}_2[X] \longrightarrow \tilde{B} \in \mathbb{F}_2[X]_{<30}[Z] \end{array} \begin{array}{l} \longrightarrow \\ \longrightarrow \end{array} \tilde{AB} \in \mathbb{F}_2[X]_{<60}[Z] \longrightarrow AB \in \mathbb{F}_2[X]$$

$$\boxed{Z = X^{30}}$$

Kronecker segmentation vs. Frobenius FFT

Naive strategy $\mathbb{F}_2[X]_{<n} \hookrightarrow \mathbb{F}_{2^{60}}[X]_{<n}$

$$\begin{array}{l} A \in \mathbb{F}_2[X] \longrightarrow \tilde{A} \in \mathbb{F}_{2^{60}}[X] \\ B \in \mathbb{F}_2[X] \longrightarrow \tilde{B} \in \mathbb{F}_{2^{60}}[X] \end{array} \begin{array}{l} \longrightarrow \\ \longrightarrow \end{array} \tilde{AB} \in \mathbb{F}_{2^{60}}[X] \longrightarrow AB \in \mathbb{F}_2[X]$$

Kronecker segmentation $\mathbb{F}_2[X]_{<n} \hookrightarrow \mathbb{F}_2[X]_{<30}[Z]_{<n/30} \hookrightarrow \mathbb{F}_{2^{60}}[Z]_{<n/30}$

$$\begin{array}{l} A \in \mathbb{F}_2[X] \longrightarrow \tilde{A} \in \mathbb{F}_2[X]_{<30}[Z] \\ B \in \mathbb{F}_2[X] \longrightarrow \tilde{B} \in \mathbb{F}_2[X]_{<30}[Z] \end{array} \begin{array}{l} \longrightarrow \\ \longrightarrow \end{array} \tilde{AB} \in \mathbb{F}_2[X]_{<60}[Z] \longrightarrow AB \in \mathbb{F}_2[X]$$

$\boxed{Z = X^{30}}$

Frobenius FFT

For ω a root of unity, $\phi : x \to x^2$ acts on $\{1, \omega, \omega^2, \omega^3, \dots\}$. The naive DFT $A \to [A(1), A(\omega), A(\omega^2), \dots]$ causes redundant computation:

$$A \in \mathbb{F}_2[X], x \in \mathbb{F}_{2^{60}} \Rightarrow A(x^2) = A(x)^2$$

Our variant of the Frobenius FFT

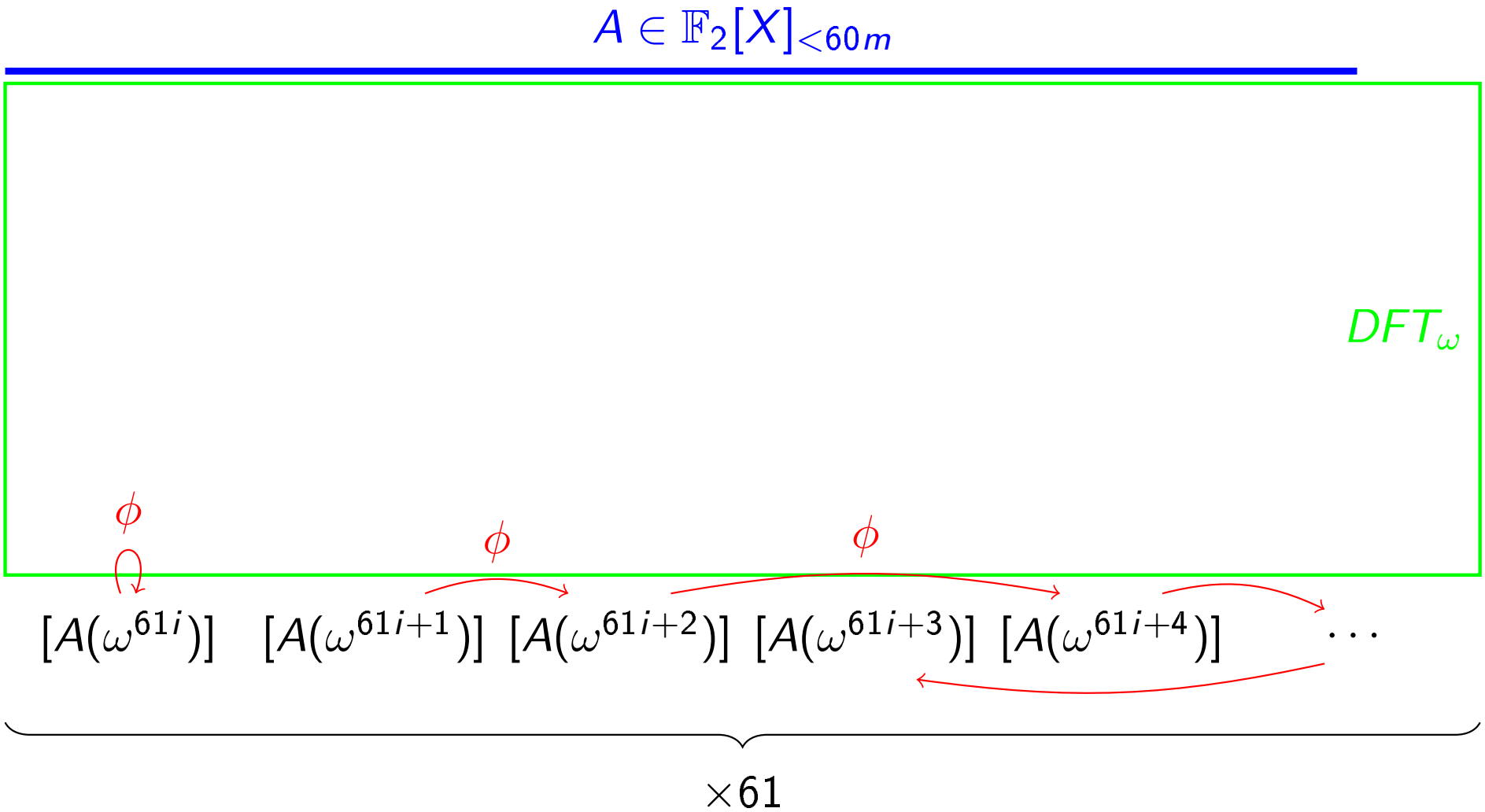
$$A \in \mathbb{F}_2[X]_{<60m}$$

DFT_ω

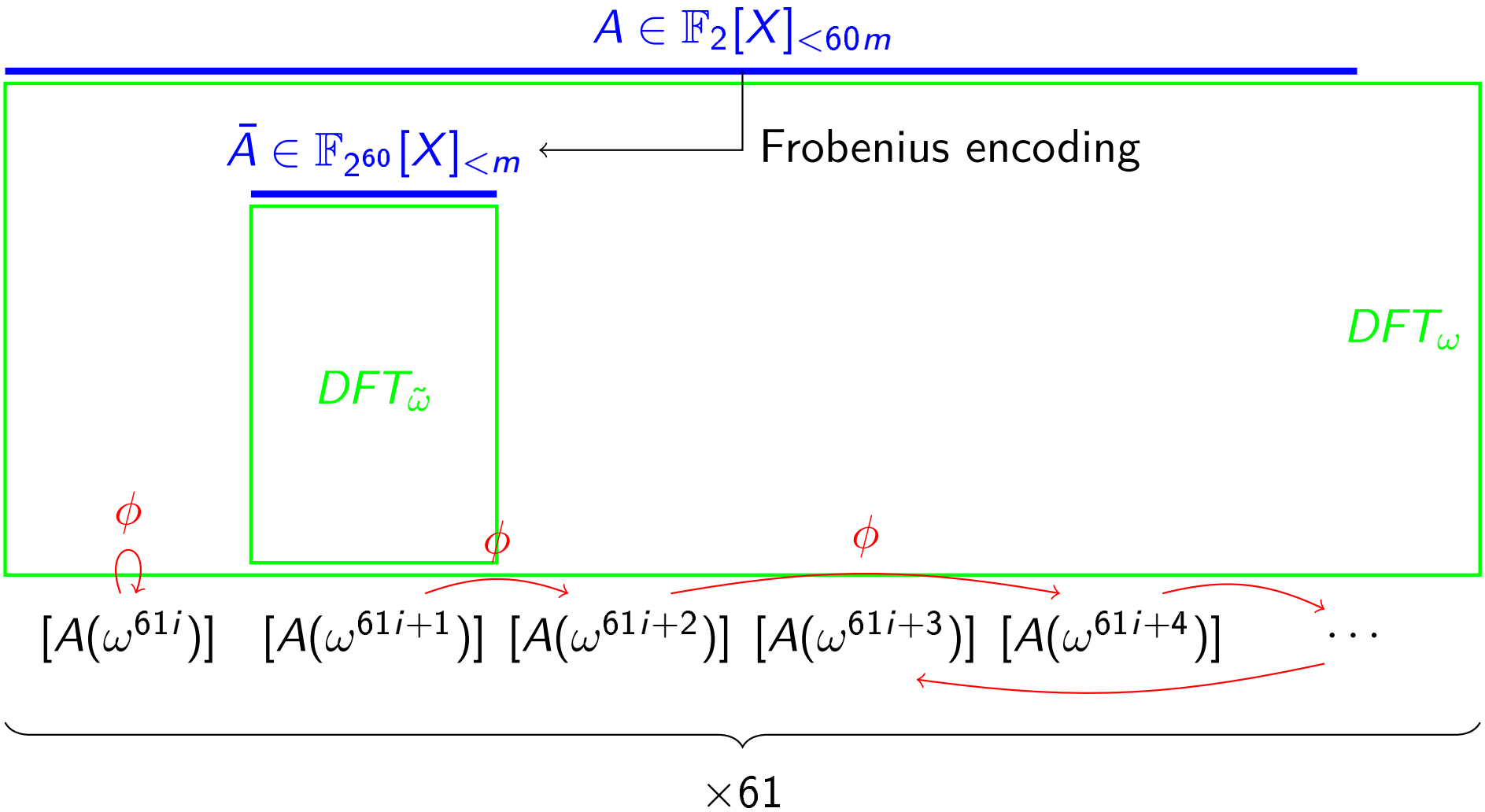
$$[A(\omega^{61i})] \quad [A(\omega^{61i+1})] \; [A(\omega^{61i+2})] \; [A(\omega^{61i+3})] \; [A(\omega^{61i+4})] \quad \cdots$$

$\times 61$

Our variant of the Frobenius FFT



Our variant of the Frobenius FFT



Multiplication algorithm

$$A \in \mathbb{F}_2[X]_{<a}$$
$$\downarrow \text{Frobenius Encoding}$$
$$\bar{A} \in \mathbb{F}_{2^{60}}[X]_{<a/60}$$
$$\downarrow DFT_{\tilde{\omega}}$$
$$E_\omega(A) \in \mathbb{F}_{2^{60}}^m$$

$a + b < 60m$
$61m$ divides $2^{60} - 1$

Multiplication algorithm

$$A \in \mathbb{F}_2[X]_{<a} \qquad\qquad B \in \mathbb{F}_2[X]_{<b}$$
$$\downarrow \text{Frobenius Encoding} \qquad\qquad \text{Frobenius Encoding} \downarrow$$
$$\bar{A} \in \mathbb{F}_{2^{60}}[X]_{<a/60} \qquad\qquad \bar{B} \in \mathbb{F}_{2^{60}}[X]_{<b/60}$$
$$\downarrow DFT_{\tilde{\omega}} \qquad\qquad DFT_{\tilde{\omega}} \downarrow$$
$$E_\omega(A) \in \mathbb{F}_{2^{60}}^m \qquad\qquad E_\omega(B) \in \mathbb{F}_{2^{60}}^m$$

$a + b < 60m$
$61m$ divides $2^{60} - 1$

Multiplication algorithm

$A \in \mathbb{F}_2[X]_{<a}$ $\qquad$ $B \in \mathbb{F}_2[X]_{<b}$

$\downarrow$ Frobenius Encoding $\qquad$ Frobenius Encoding $\downarrow$

$\bar{A} \in \mathbb{F}_{2^{60}}[X]_{<a/60}$ $\qquad$ $\bar{B} \in \mathbb{F}_{2^{60}}[X]_{<b/60}$

$\downarrow DFT_{\tilde{\omega}}$ $\qquad$ $DFT_{\tilde{\omega}} \downarrow$

$E_\omega(A) \in \mathbb{F}_{2^{60}}^m$ $\qquad$ $E_\omega(B) \in \mathbb{F}_{2^{60}}^m$

pointwise product

$E_\omega(AB) \in \mathbb{F}_{2^{60}}^m$

$\downarrow DFT_{\tilde{\omega}}^{-1}$

$\bar{AB} \in \mathbb{F}_{2^{60}}[X]_{<m}$

$\downarrow$ Frobenius Decoding

$AB \in \mathbb{F}_2[X]_{<a+b}$

$a + b < 60m$
$61m$ divides $2^{60} - 1$

Frobenius encoding

Cooley-Tukey FFT

$$A(\omega^{61i+1}) = \sum_{k<m} \omega^k \left(\sum_{l<61} a_{k+ml} \omega^{ml} \right) \omega^{61ki}$$

Frobenius encoding

Cooley-Tukey FFT

$$A(\omega^{61i+1}) = \sum_{k<m} \omega^k \left(\sum_{l<61} a_{k+ml} \omega^{ml} \right) \omega^{61ki}$$

- $\theta := \omega^m, \tilde{\omega} := \omega^{61}$
- $\tilde{A}_k := \sum_{l<61} a_{k+ml} X^l \in \mathbb{F}_2[X]_{<60} \quad (A \in \mathbb{F}_2[X]_{<60m})$
- $\bar{A} = \sum_{k<m} \omega^k \tilde{A}_k(\theta) Z^k \in \mathbb{F}_{2^{60}}[Z]_{<m}$

Frobenius encoding

Cooley-Tukey FFT

$$A(\omega^{61i+1}) = \sum_{k<m} \omega^k \left(\sum_{l<61} a_{k+ml} \omega^{ml} \right) \omega^{61ki}$$

- $\theta := \omega^m, \tilde{\omega} := \omega^{61}$
- $\tilde{A}_k := \sum_{l<61} a_{k+ml} X^l \in \mathbb{F}_2[X]_{<60} \quad (A \in \mathbb{F}_2[X]_{<60m})$
- $\bar{A} = \sum_{k<m} \omega^k \tilde{A}_k(\theta) Z^k \in \mathbb{F}_{2^{60}}[Z]_{<m}$

Technical assumption

Assume ω chosen such that $\theta = z \bmod \mu(z)$ with $\mu(z) := \frac{z^{61}-1}{z-1}$

Outline

Introduction

Presentation of the algorithm

Implementation details
- Data Representation
- Frobenius encoding
- Timings

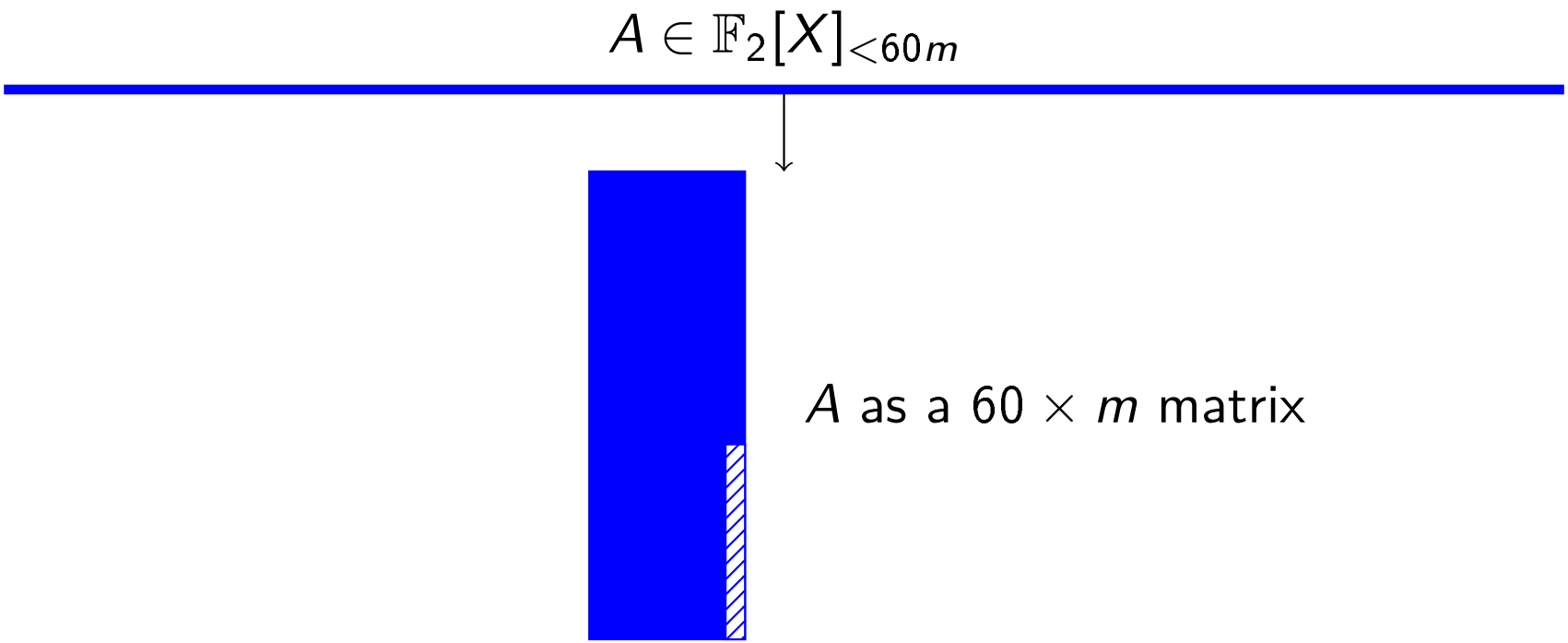
Perspectives

Data Representation

- Polynomials over $\mathbb{F}_2$ in packed representation.
- Elements of $\mathbb{F}_{2^{60}}$ on one machine word; polynomials over $\mathbb{F}_{2^{60}}$ as an array of words.
- Matrices over $\mathbb{F}_2$ in packed column representation.

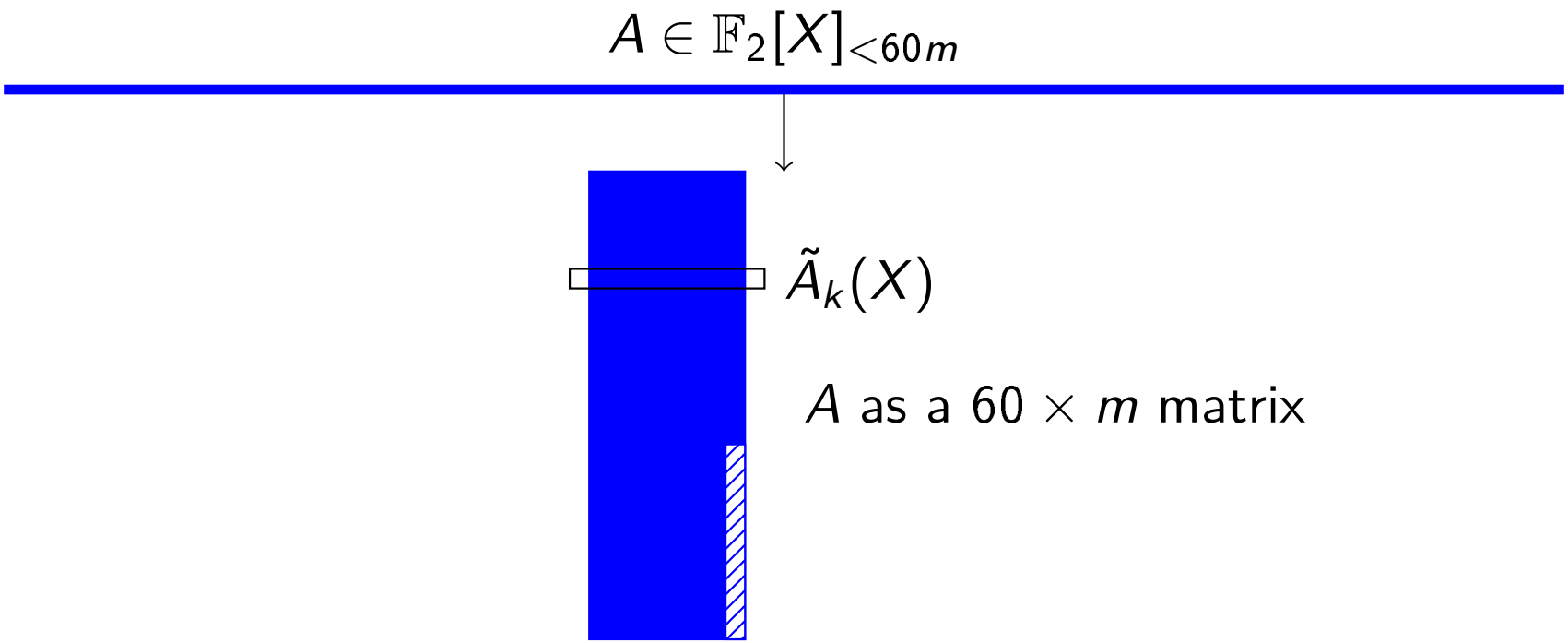
Data Representation

- Polynomials over $\mathbb{F}_2$ in packed representation.
- Elements of $\mathbb{F}_{2^{60}}$ on one machine word; polynomials over $\mathbb{F}_{2^{60}}$ as an array of words.
- Matrices over $\mathbb{F}_2$ in packed column representation.



Data Representation

- Polynomials over $\mathbb{F}_2$ in packed representation.
- Elements of $\mathbb{F}_{2^{60}}$ on one machine word; polynomials over $\mathbb{F}_{2^{60}}$ as an array of words.
- Matrices over $\mathbb{F}_2$ in packed column representation.



Frobenius encoding

- See A as a $60 \times m$ matrix; add 4 columns for alignment.
- Transpose the $64 \times m$ matrix ($\Rightarrow [\tilde{A}_k(\theta)]_{k<m}$) .
- Multiply by the twiddle factors $\omega^k (\Rightarrow \bar{A})$.

Frobenius encoding

- See A as a $60 \times m$ matrix; add 4 columns for alignment.
- Transpose the $64 \times m$ matrix ($\Rightarrow [\tilde{A}_k(\theta)]_{k<m}$) .
- Multiply by the twiddle factors $\omega^k (\Rightarrow \bar{A})$.

Matrix transposition

Exploit the AVX2 instruction set

- Reduction $(64 \times m) \rightarrow (64 \times 256) \rightarrow (8 \times 8)$
- Transpose 4 packed 8×8 matrices at once using vector instruction.

Frobenius encoding

- See A as a $60 \times m$ matrix; add 4 columns for alignment.
- Transpose the $64 \times m$ matrix ($\Rightarrow [\tilde{A}_k(\theta)]_{k<m}$) .
- Multiply by the twiddle factors $\omega^k (\Rightarrow \bar{A})$.

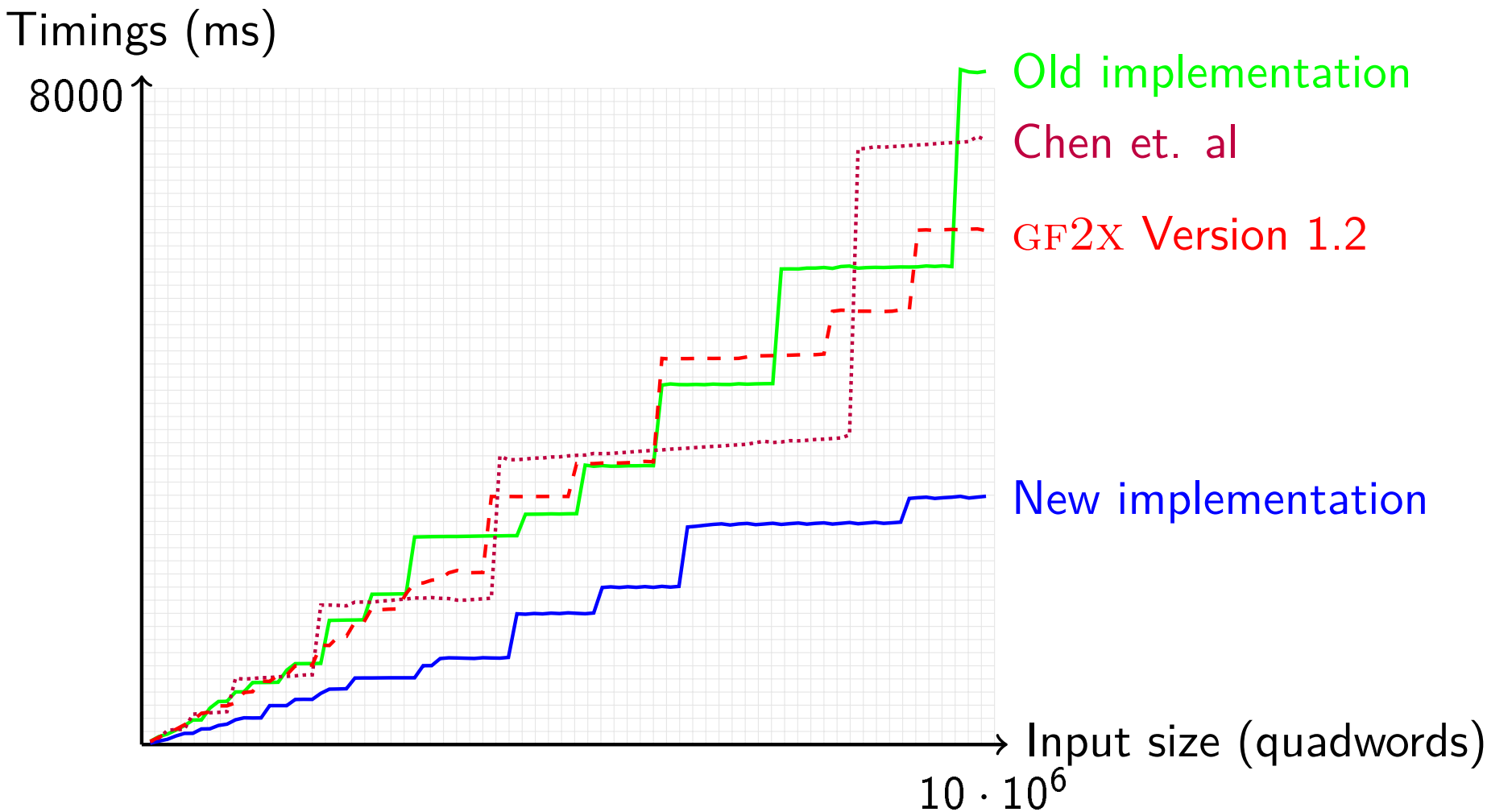
Matrix transposition

Exploit the AVX2 instruction set

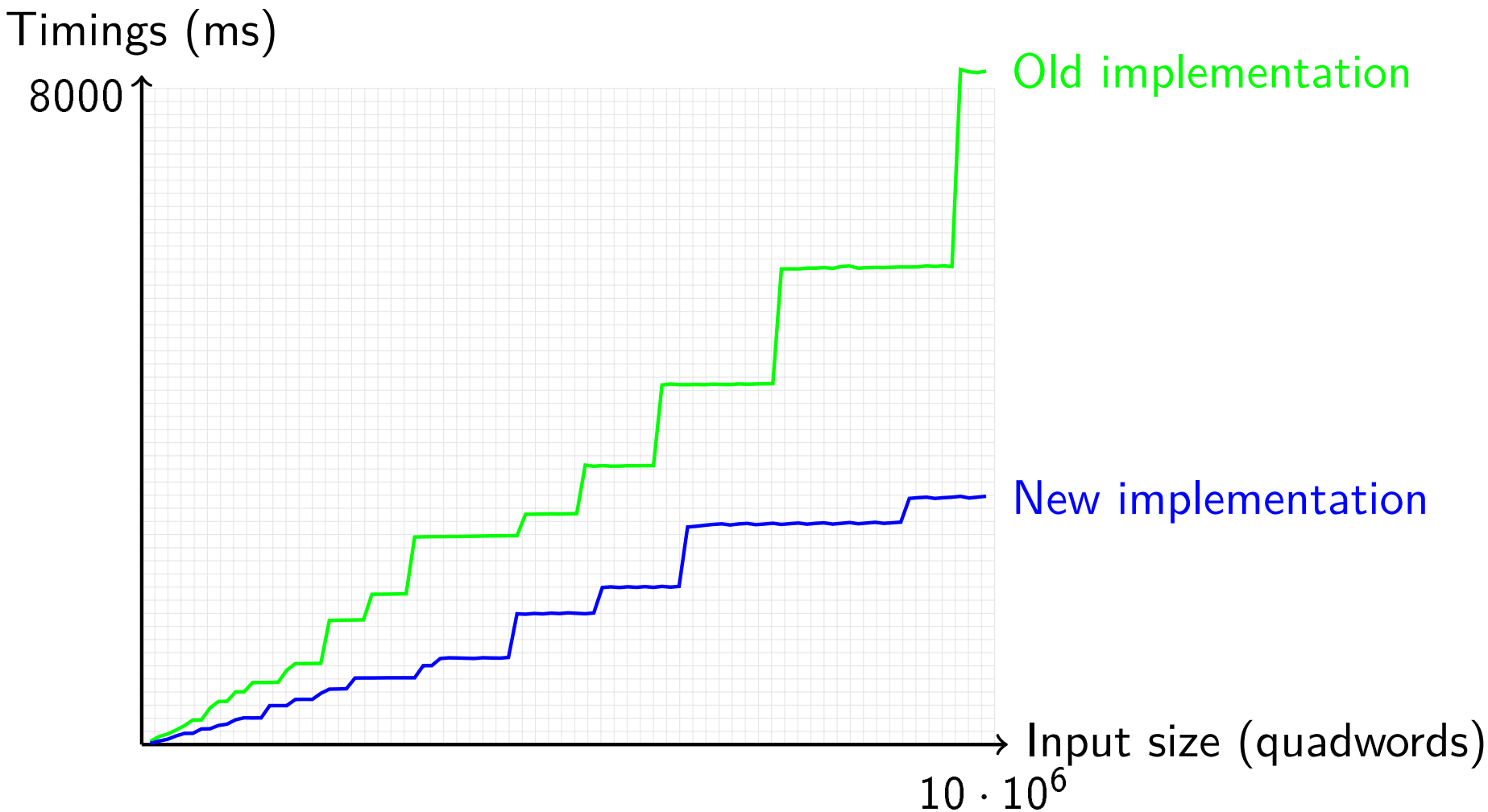
- Reduction $(64 \times m) \rightarrow (64 \times 256) \rightarrow (8 \times 8)$
- Transpose 4 packed 8×8 matrices at once using vector instruction.

Finally, call the efficient FFT over $\mathbb{F}_{2^{60}}$ on $\bar{A}$.

Timings


Timings (ms)
8000
Old implementation
Chen et. al
GF2X Version 1.2
New implementation
Input size (quadwords)
$10 \cdot 10^6$

Timings


Timings (ms)
8000
Old implementation
New implementation
Input size (quadwords)
$10 \cdot 10^6$

Outline

Introduction

Presentation of the algorithm

Implementation details

Perspectives

Perspectives

Better use of vector instructions

- Vectorize the FFT routine over $\mathbb{F}_{2^{60}}$.
- Support for the new AVX-512.

Perspectives

Better use of vector instructions

- Vectorize the FFT routine over $\mathbb{F}_{2^{60}}$.
- Support for the new AVX-512.

Others

- Use the Truncated Fourier Transform (reduce the staircase effect)
- Generalization for other finite fields

Questions?

Thank you for your attention