Efficient Arithmetic of Finite Field Extensions

Édouard Rousseau

July 12, 2021
PhD Defense

Introduction

What are finite fields?

- In mathematics, we study **sets of numbers**:
  - The set of natural numbers $\mathbb{N}$: $0, 1, 2, 3, \dots$
  - The set of integers $\mathbb{Z}$: $\dots, -2, -1, 0, 1, 2, \dots$
  - The set of rational fractions $\mathbb{Q}$: $0, 1, \frac{1}{2}, \frac{1}{3}, -\frac{2}{7}, \dots$
  - The set of real numbers $\mathbb{R}$: $0, 1, \frac{1}{2}, -\frac{2}{7}, \sqrt{2}, \pi, \dots$
- and **operations** between these numbers:
  - $1 + 2$ in $\mathbb{N}$
  - $3 - (-2)$ in $\mathbb{Z}$
  - $5 \times \frac{2}{3}$ in $\mathbb{Q}$
  - $\sqrt{2}/3$ in $\mathbb{R}$
- A **field** is a set of numbers with operations $+, -, \times, /$
- It is called **finite** when it contains only a finite number of elements

ARITHMETIC OF EXTENSIONS

- The simplest example of **finite field** is $\mathbb{F}_p = \mathbb{Z}/p\mathbb{Z} = \{0, 1, \ldots, p-1\}$, where all the operations are taken modulo a prime number p.
- $\mathbb{F}_p$ has p elements
  - There exists exactly one finite field of size p^k for all $k \geq 1$
  - The field of size p^k, $\mathbb{F}_{p^k}$, is an **extension** of $\mathbb{F}_p$
  - We have $\mathbb{F}_p \subset \mathbb{F}_{p^k}$
- We are interested in **computer algebra**
  - Particularly in the arithmetic of $\mathbb{F}_{p^k}$, *i.e.* how to perform operations in $\mathbb{F}_{p^k}$ **efficiently, on a computer**

Applications of finite fields

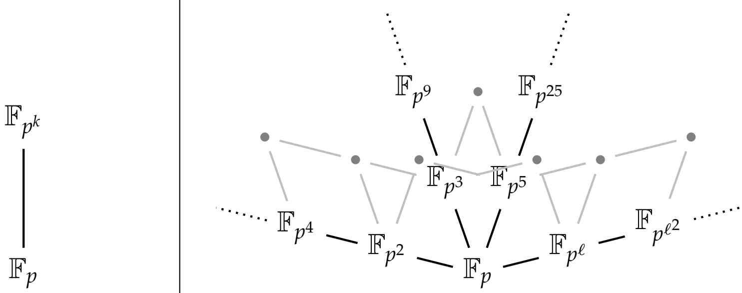
Finite fields are widely used in many areas:

- number theory
- algebraic geometry
- coding theory
- cryptography

Goals

- Improve the arithmetic in finite field extensions
- Two directions of study

single extension | **many** extensions



$\rightsquigarrow$ efficient **operations** in one given field

$\rightsquigarrow$ efficient **morphisms** between fields

Contributions

Published in the *International Symposium on Symbolic and Algebraic Computation* (ISSAC):

- *Lattices of compatibly embedded finite fields in Nemo/Flint*, Luca De Feo, Hugues Randriambololona, and É. R., 2018
- *Standard lattices of compatibly embedded finite fields*, Luca De Feo, Hugues Randriambololona and É. R., 2019

Published in the *International Workshop on the Arithmetic of Finite Fields* (WAIFI):

- *Trisymmetric multiplication formulae in finite fields*, Hugues Randriambololona and É. R., 2020

Single extension

Finite field arithmetic

Notation: $\mathbb{F}_{p^k}$ denotes *the* finite field with p^k elements

$$\mathbb{F}_{p^k} \cong \mathbb{F}_p[X]/(P(X))$$

- $P \in \mathbb{F}_p[X]$ is an **irreducible** polynomial of degree k

Some possible **representations**:

- **Zech's logarithm**: elements are represented as generator powers
- **normal** basis: $(\alpha, \alpha^\sigma, \dots, \alpha^{\sigma^{k-1}})$
- **monomial** basis: $(1, \bar{X}, \dots, \bar{X}^{k-1})$

Finite field arithmetic

Notation: $\mathbb{F}_{p^k}$ denotes *the* finite field with p^k elements

$$\mathbb{F}_{p^k} \cong \mathbb{F}_p[X]/(P(X))$$

- $P \in \mathbb{F}_p[X]$ is an **irreducible** polynomial of degree k

Some possible **representations**:

- **Zech's logarithm**: elements are represented as generator powers
  - fast, but only possible for small fields
- **normal** basis: $(\alpha, \alpha^\sigma, \ldots, \alpha^{\sigma^{k-1}})$
- **monomial** basis: $(1, \bar{X}, \ldots, \bar{X}^{k-1})$

Finite field arithmetic

Notation: $\mathbb{F}_{p^k}$ denotes *the* finite field with p^k elements

$$\mathbb{F}_{p^k} \cong \mathbb{F}_p[X]/(P(X))$$

- $P \in \mathbb{F}_p[X]$ is an **irreducible** polynomial of degree k

Some possible **representations**:

- **Zech's logarithm**: elements are represented as generator powers
  - fast, but only possible for small fields
- **normal** basis: $(\alpha, \alpha^{\sigma}, \dots, \alpha^{\sigma^{k-1}})$
  - fast Frobenius evaluation but slow multiplication
- **monomial** basis: $(1, \bar{X}, \dots, \bar{X}^{k-1})$

FINITE FIELD ARITHMETIC

Notation: $\mathbb{F}_{p^k}$ denotes *the* finite field with p^k elements

$$\mathbb{F}_{p^k} \cong \mathbb{F}_p[X]/(P(X))$$

- $P \in \mathbb{F}_p[X]$ is an **irreducible** polynomial of degree k

Some possible **representations**:

- **Zech's logarithm**: elements are represented as generator powers
  - fast, but only possible for small fields
- **normal** basis: $(\alpha, \alpha^\sigma, \dots, \alpha^{\sigma^{k-1}})$
  - fast Frobenius evaluation but slow multiplication
- **monomial** basis: $(1, \bar{X}, \dots, \bar{X}^{k-1})$
  - commonly used representation, easy to construct
  - multiplication slower than addition

MOTIVATION

- Computations in an extension $\mathbb{F}_{p^k}$

Motivation

- Computations in an extension $\mathbb{F}_{p^k}$
  - multiplications: **expensive** ☹
  - additions, scalar multiplications: **cheap** ☺

Motivation

- Computations in an extension $\mathbb{F}_{p^k}$
  - multiplications: **expensive** ☹
  - additions, scalar multiplications: **cheap** ☺
- we want to study/reduce the cost of multiplication

Motivation

- Computations in an extension $\mathbb{F}_{p^k}$
  - multiplications: **expensive** ☹
  - additions, scalar multiplications: **cheap** ☺
- we want to study/reduce the cost of multiplication
- A lot of litterature on the subject

MOTIVATION

- Computations in an extension $\mathbb{F}_{p^k}$
  - multiplications: **expensive** ☹
  - additions, scalar multiplications: **cheap** ☺
- we want to study/reduce the cost of multiplication
- A lot of litterature on the subject
  - **Karatsuba** (1962)

Motivation

- Computations in an extension $\mathbb{F}_{p^k}$
  - multiplications: **expensive** ☹
  - additions, scalar multiplications: **cheap** ☺
- we want to study/reduce the cost of multiplication
- A lot of litterature on the subject
  - **Karatsuba** (1962)
  - Toom-Cook (1963), **evaluation-interpolation** techniques

Motivation

- Computations in an extension $\mathbb{F}_{p^k}$
  - multiplications: **expensive** ☹
  - additions, scalar multiplications: **cheap** ☺
- we want to study/reduce the cost of multiplication
- A lot of litterature on the subject
  - **Karatsuba** (1962)
  - Toom-Cook (1963), **evaluation-interpolation** techniques
  - **Schönhage-Strassen** (1971)

MOTIVATION

- Computations in an extension $\mathbb{F}_{p^k}$
  - multiplications: **expensive** ☹
  - additions, scalar multiplications: **cheap** ☺
- we want to study/reduce the cost of multiplication
- A lot of litterature on the subject
  - **Karatsuba** (1962)
  - Toom-Cook (1963), **evaluation-interpolation** techniques
  - **Schönhage-Strassen** (1971)
  - …

Motivation

- Computations in an extension $\mathbb{F}_{p^k}$
  - multiplications: **expensive** ☹
  - additions, scalar multiplications: **cheap** ☺
- we want to study/reduce the cost of multiplication
- A lot of litterature on the subject
  - **Karatsuba** (1962)
  - Toom-Cook (1963), **evaluation-interpolation** techniques
  - **Schönhage-Strassen** (1971)
  - …
  - $O(k \log k)$ algorithm [Harvey, Van Der Hoeven '19]

MODELS OF COMPLEXITY

$\mathcal{A}$ an $\mathbb{F}_p$-algebra

- **algebraic** complexity: we count all operations $+, \times$ in $\mathbb{F}_p$
- **bilinear** complexity: we count only the multiplications
  - nice results with polynomials: **Karatsuba's algorithm**
  - and with matrices: **Strassen's algorithm**

When $\mathcal{A} = \mathbb{F}_{p^k}$:

- theoretical interest
- links with coding theory
- links with algebraic geometry

BILINEAR COMPLEXITY: INTUITION

- $\mathbb{F}_{p^k}$ an extension of $\mathbb{F}_p$
- **bilinear complexity:** number of subproducts in $\mathbb{F}_p$ needed to compute a product in $\mathbb{F}_{p^k}$

Karatsuba:

$$(a_0 + a_1 X)(b_0 + b_1 X) =$$

$$a_0 b_0 + (a_0 b_1 + a_1 b_0) X + a_1 b_1 X^2$$

Bilinear complexity: intuition

- $\mathbb{F}_{p^k}$ an extension of $\mathbb{F}_p$
- **bilinear complexity:** number of subproducts in $\mathbb{F}_p$ needed to compute a product in $\mathbb{F}_{p^k}$

Karatsuba:

$$(a_0 + a_1 X)(b_0 + b_1 X) =$$

$$\mathbf{a_0 b_0} + (\mathbf{a_0 b_1} + \mathbf{a_1 b_0})X + \mathbf{a_1 b_1} X^2$$

Bilinear complexity: intuition

- $\mathbb{F}_{p^k}$ an extension of $\mathbb{F}_p$
- **bilinear complexity:** number of subproducts in $\mathbb{F}_p$ needed to compute a product in $\mathbb{F}_{p^k}$

Karatsuba:

$$(a_0 + a_1X)(b_0 + b_1X) =$$

$$c_0 + (c_2 - c_1 - c_0)X + c_1X^2$$

with

$$\begin{cases} c_0 = a_0b_0 \\ c_1 = a_1b_1 \\ c_2 = (a_0 + a_1)(b_0 + b_1) \end{cases}$$

BILINEAR COMPLEXITY: INTUITION

- $\mathbb{F}_{p^k}$ an extension of $\mathbb{F}_p$
- **bilinear complexity:** number of subproducts in $\mathbb{F}_p$ needed to compute a product in $\mathbb{F}_{p^k}$

Karatsuba:

$$(a_0 + a_1 X)(b_0 + b_1 X) =$$

$$\mathbf{c_0} + (\mathbf{c_2} - \mathbf{c_1} - \mathbf{c_0})X + \mathbf{c_1} X^2$$

with

$$\begin{cases} c_0 &= a_0 b_0 \\ c_1 &= a_1 b_1 \\ c_2 &= (a_0 + a_1)(b_0 + b_1) \end{cases}$$

Bilinear complexity: intuition

- $\mathbb{F}_{p^k}$ an extension of $\mathbb{F}_p$
- **bilinear complexity:** number of subproducts in $\mathbb{F}_p$ needed to compute a product in $\mathbb{F}_{p^k}$

Karatsuba:

$$(a_0 + a_1 X)(b_0 + b_1 X) =$$

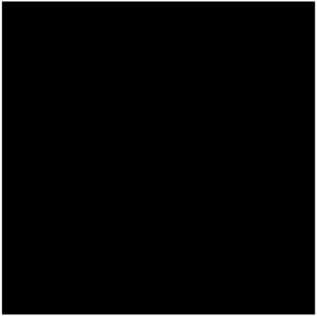
$$\mathbf{c_0} + (\mathbf{c_2} - \mathbf{c_1} - \mathbf{c_0})X + \mathbf{c_1} X^2$$

with

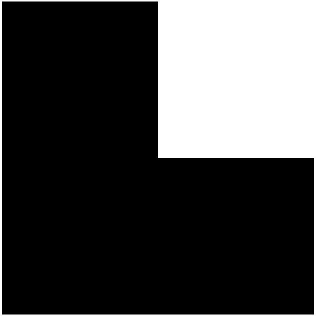
$$\begin{cases} c_0 &= a_0 b_0 \\ c_1 &= a_1 b_1 \\ c_2 &= (a_0 + a_1)(b_0 + b_1) \end{cases}$$

- ☹ **Hard** to compute the bilinear complexity of a product: unkwown even for the 3×3 matrix product

Complexity of Karatsuba's algorithm

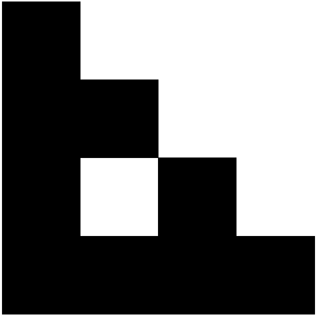


Complexity of Karatsuba's algorithm



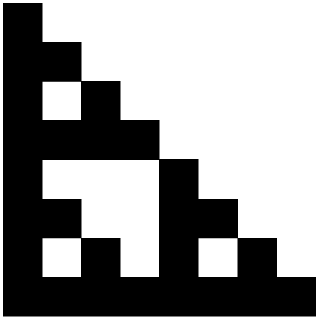
- Degree 2: 3 **multiplications instead of** 4

Complexity of Karatsuba's algorithm



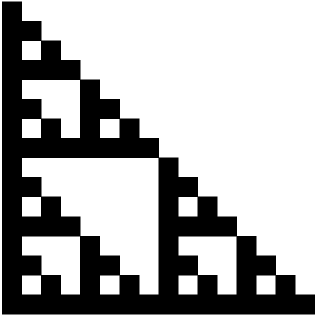
- Degree 2: 3 **multiplications instead of** 4
- Higher degrees: reccursive strategy

Complexity of Karatsuba's algorithm



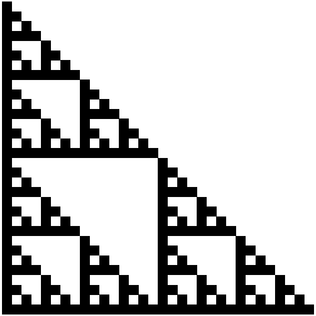
- Degree 2: 3 **multiplications instead of** 4
- Higher degrees: reccursive strategy
- Asymptotically: $O(n^{1.58})$ instead of $O(n^2)$

Complexity of Karatsuba's algorithm



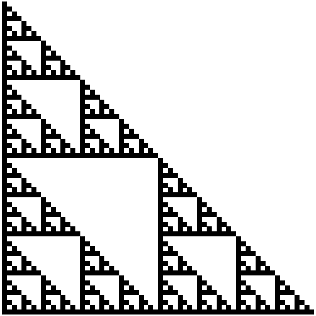
- Degree 2: 3 **multiplications instead of** 4
- Higher degrees: reccursive strategy
- Asymptotically: $O(n^{1.58})$ instead of $O(n^2)$

COMPLEXITY OF KARATSUBA'S ALGORITHM



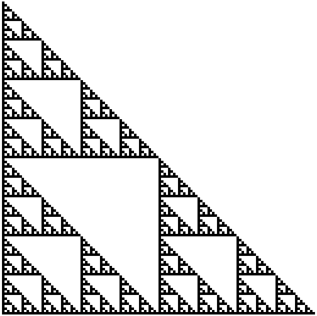
- Degree 2: 3 **multiplications instead of** 4
- Higher degrees: reccursive strategy
- Asymptotically: $O(n^{1.58})$ instead of $O(n^2)$

Complexity of Karatsuba's algorithm



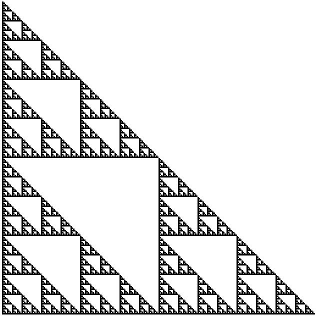
- Degree 2: 3 **multiplications instead of** 4
- Higher degrees: reccursive strategy
- Asymptotically: $O(n^{1.58})$ instead of $O(n^2)$

Complexity of Karatsuba's algorithm



- Degree 2: 3 **multiplications instead of** 4
- Higher degrees: reccursive strategy
- Asymptotically: $O(n^{1.58})$ instead of $O(n^2)$

Complexity of Karatsuba's algorithm



- Degree 2: 3 **multiplications instead of** 4
- Higher degrees: reccursive strategy
- Asymptotically: $O(n^{1.58})$ instead of $O(n^2)$

Bilinear complexity: definition

Definition

The **bilinear complexity** of the product in $\mathbb{F}_{p^k}$ is the minimal integer $r \in \mathbb{N}$ such that you can write, for all $x, y \in \mathbb{F}_{p^k}$

$$xy = \sum_{j=1}^{r} \varphi_j(x)\psi_j(y) \cdot \alpha_j$$

with φ_j, ψ_j linear forms and α_j elements of $\mathbb{F}_{p^k}$.

Bilinear complexity: definition

Definition

The **bilinear complexity** of the product in $\mathbb{F}_{p^k}$ is the minimal integer $r \in \mathbb{N}$ such that you can write, for all $x, y \in \mathbb{F}_{p^k}$

$$xy = \sum_{j=1}^{r} \varphi_{\mathbf{j}}(\mathbf{x})\psi_{\mathbf{j}}(\mathbf{y}) \cdot \alpha_j$$

with φ_j, ψ_j linear forms and α_j elements of $\mathbb{F}_{p^k}$.

- $\varphi_j(x)$: linear combination of the coordinates x_i of x
- $\psi_j(y)$: linear combination of the coordinates y_i of y

Bilinear complexity: definition

Definition

The **bilinear complexity** of the product in $\mathbb{F}_{p^k}$ is the minimal integer $r \in \mathbb{N}$ such that you can write, for all $x, y \in \mathbb{F}_{p^k}$

$$xy = \sum_{j=1}^{r} \varphi_{\mathbf{j}}(\mathbf{x})\psi_{\mathbf{j}}(\mathbf{y}) \cdot \alpha_j$$

with φ_j, ψ_j linear forms and α_j elements of $\mathbb{F}_{p^k}$.

- $\varphi_j(x)$: linear combination of the coordinates x_i of x
- $\psi_j(y)$: linear combination of the coordinates y_i of y

Notations and questions

- $\mu_p(k)$ = bilinear complexity of the product in $\mathbb{F}_{p^k}$

Two independent questions:

- What is the asymptotic behaviour of $\mu_p(k)$?

- Can we find values $\mu_p(k)$ for small k?

Notations and questions

- $\mu_p(k)$ = bilinear complexity of the product in $\mathbb{F}_{p^k}$

Two independent questions:

- What is the asymptotic behaviour of $\mu_p(k)$?
  - $\mu_p(k)$ is **linear** in k
- Can we find values $\mu_p(k)$ for small k?

NOTATIONS AND QUESTIONS

- $\mu_p(k)$ = bilinear complexity of the product in $\mathbb{F}_{p^k}$

Two independent questions:

- What is the asymptotic behaviour of $\mu_p(k)$?
  - $\mu_p(k)$ is **linear** in k
  - **Evaluation-interpolation** techniques:
- Can we find values $\mu_p(k)$ for small k?

NOTATIONS AND QUESTIONS

- $\mu_p(k) =$ bilinear complexity of the product in $\mathbb{F}_{p^k}$

Two independent questions:

- What is the asymptotic behaviour of $\mu_p(k)$?
  - $\mu_p(k)$ is **linear** in k
  - **Evaluation-interpolation** techniques:
    - [Chudnovsky-Chudnovsky '87]
    - [Shparlinski-Tsfasman-Vladut '92]
    - [Ballet '99]
    - [Randriambololona '12]
    - …
- Can we find values $\mu_p(k)$ for small k?

Notations and questions

- $\mu_p(k) =$ bilinear complexity of the product in $\mathbb{F}_{p^k}$

Two independent questions:

- What is the asymptotic behaviour of $\mu_p(k)$?
  - $\mu_p(k)$ is **linear** in k
  - **Evaluation-interpolation** techniques:
    - [Chudnovsky-Chudnovsky '87]
    - [Shparlinski-Tsfasman-Vladut '92]
    - [Ballet '99]
    - [Randriambololona '12]
    - ...
- Can we find values $\mu_p(k)$ for small k?
  - Clever exhaustive search [BDEZ '12] [Covanov '18]

Symmetric decompositions

Classic decompositions	**Symmetric** decompositions
$xy = \sum_{j=1}^{r} \varphi_j(x)\psi_j(y) \cdot \alpha_j$	$yx = xy = \sum_{j=1}^{r} \varphi_j(x)\varphi_j(y) \cdot \alpha_j$

Symmetric decompositions

Classic decompositions	**Symmetric** decompositions
$xy = \sum_{j=1}^{r} \varphi_j(x)\psi_j(y) \cdot \alpha_j$	$yx = xy = \sum_{j=1}^{r} \varphi_{\mathbf{j}}(x)\varphi_{\mathbf{j}}(y) \cdot \alpha_j$

Symmetric decompositions

Classic decompositions	**Symmetric** decompositions
$xy = \sum_{j=1}^{r} \varphi_j(x)\psi_j(y) \cdot \alpha_j$	$yx = xy = \sum_{j=1}^{r} \varphi_{\mathbf{j}}(x)\varphi_{\mathbf{j}}(y) \cdot \alpha_j$

Notation: for $\mathbb{F}_{p^k}$, we note $\mu_p^{\mathrm{sym}}(k)$ the minimal length r in a **symmetric** decomposition

Symmetric decompositions

Classic decompositions	**Symmetric** decompositions
$xy = \sum_{j=1}^{r} \varphi_j(x)\psi_j(y) \cdot \alpha_j$	$yx = xy = \sum_{j=1}^{r} \boldsymbol{\varphi_j}(x)\boldsymbol{\varphi_j}(y) \cdot \alpha_j$

Notation: for $\mathbb{F}_{p^k}$, we note $\mu_p^{\text{sym}}(k)$ the minimal length r in a **symmetric** decomposition

- **Asymptotics:** $\mu_p^{\text{sym}}(k)$ is **linear** in k

Symmetric decompositions

Classic decompositions	**Symmetric** decompositions
$xy = \sum_{j=1}^{r} \varphi_j(x)\psi_j(y) \cdot \alpha_j$	$yx = xy = \sum_{j=1}^{r} \varphi_{\mathbf{j}}(x)\varphi_{\mathbf{j}}(y) \cdot \alpha_j$

Notation: for $\mathbb{F}_{p^k}$, we note $\mu_p^{\mathrm{sym}}(k)$ the minimal length r in a **symmetric** decomposition

- **Asymptotics:** $\mu_p^{\mathrm{sym}}(k)$ is **linear** in k
- **Small values: smaller** search space $\rightsquigarrow$ **faster** algorithms

Even more symmetry

- every linear form $\varphi \in (\mathbb{F}_{p^k})^\vee$ can be written $x \mapsto \mathrm{Tr}(\alpha x)$ for some $\alpha \in \mathbb{F}_{p^k}$, with Tr the trace of $\mathbb{F}_{p^k}/\mathbb{F}_p$
- we can rewrite the formula

$$xy = \sum_{j=1}^{r} \varphi_j(x)\varphi_j(y) \cdot \beta_j$$

Even more symmetry

- every linear form $\varphi \in (\mathbb{F}_{p^k})^\vee$ can be written $x \mapsto \mathrm{Tr}(\alpha x)$ for some $\alpha \in \mathbb{F}_{p^k}$, with Tr the trace of $\mathbb{F}_{p^k}/\mathbb{F}_p$
- we can rewrite the formula

$$xy = \sum_{j=1}^{r} \mathrm{Tr}(\alpha_j x)\,\mathrm{Tr}(\alpha_j y) \cdot \beta_j$$

EVEN MORE SYMMETRY

- every linear form $\varphi \in (\mathbb{F}_{p^k})^\vee$ can be written $x \mapsto \mathrm{Tr}(\alpha x)$ for some $\alpha \in \mathbb{F}_{p^k}$, with Tr the trace of $\mathbb{F}_{p^k}/\mathbb{F}_p$
- we can rewrite the formula, and even ask $\beta_j = \lambda_j \alpha_j$

$$xy = \sum_{j=1}^{r} \lambda_j \,\mathrm{Tr}(\alpha_j x)\,\mathrm{Tr}(\alpha_j y) \cdot \alpha_j$$

with $\lambda_j \in \mathbb{F}_p$ scalars

EVEN MORE SYMMETRY

- every linear form $\varphi \in (\mathbb{F}_{p^k})^\vee$ can be written $x \mapsto \mathrm{Tr}(\alpha x)$ for some $\alpha \in \mathbb{F}_{p^k}$, with Tr the trace of $\mathbb{F}_{p^k}/\mathbb{F}_p$
- we can rewrite the formula, and even ask $\beta_j = \lambda_j \alpha_j$

$$xy = \sum_{j=1}^{r} \lambda_j \,\mathrm{Tr}(\alpha_j x)\,\mathrm{Tr}(\alpha_j y) \cdot \alpha_j$$

with $\lambda_j \in \mathbb{F}_p$ scalars

- we call these formulae **trisymmetric** decompositions

Even more symmetry

- every linear form $\varphi \in (\mathbb{F}_{p^k})^\vee$ can be written $x \mapsto \mathrm{Tr}(\alpha x)$ for some $\alpha \in \mathbb{F}_{p^k}$, with Tr the trace of $\mathbb{F}_{p^k}/\mathbb{F}_p$
- we can rewrite the formula, and even ask $\beta_j = \lambda_j \alpha_j$

$$xy = \sum_{j=1}^{r} \lambda_j \,\mathrm{Tr}(\alpha_j x)\,\mathrm{Tr}(\alpha_j y) \cdot \alpha_j$$

with $\lambda_j \in \mathbb{F}_p$ scalars

- we call these formulae **trisymmetric** decompositions
- we note $\mu_p^{\mathrm{tri}}(k)$ the minimal r in such formulae

Example of trisymmetric decomposition

- $\mathbb{F}_{3^2} \cong \mathbb{F}_3[z]/(z^2 - z - 1) \cong \mathbb{F}_3(\zeta)$
- $x, y \in \mathbb{F}_{3^2}$, $x = x_0 + x_1\zeta$ and $y = y_0 + y_1\zeta$

Example of trisymmetric decomposition

- $\mathbb{F}_{3^2} \cong \mathbb{F}_3[z]/(z^2 - z - 1) \cong \mathbb{F}_3(\zeta)$
- $x, y \in \mathbb{F}_{3^2}$, $x = x_0 + x_1\zeta$ and $y = y_0 + y_1\zeta$

$$(x_0 + x_1\zeta)(y_0 + y_1\zeta) = (x_0y_0 + x_1y_1) + (x_0y_1 + x_1y_0 + x_1y_1)\zeta$$

Example of trisymmetric decomposition

- $\mathbb{F}_{3^2} \cong \mathbb{F}_3[z]/(z^2 - z - 1) \cong \mathbb{F}_3(\zeta)$
- $x, y \in \mathbb{F}_{3^2}$, $x = x_0 + x_1\zeta$ and $y = y_0 + y_1\zeta$

$$(x_0 + x_1\zeta)(y_0 + y_1\zeta) = (x_0y_0 + x_1y_1) + (x_0y_1 + x_1y_0 + x_1y_1)\zeta$$

$$\begin{aligned} xy \quad = \quad & -\operatorname{Tr}(1 \times x)\operatorname{Tr}(1 \times y) \cdot 1 - \operatorname{Tr}(\zeta \times x)\operatorname{Tr}(\zeta \times y) \cdot \zeta \\ & + \operatorname{Tr}((\zeta - 1) \times x)\operatorname{Tr}((\zeta - 1) \times y) \cdot (\zeta - 1) \end{aligned}$$

Example of trisymmetric decomposition

- $\mathbb{F}_{3^2} \cong \mathbb{F}_3[z]/(z^2 - z - 1) \cong \mathbb{F}_3(\zeta)$
- $x, y \in \mathbb{F}_{3^2}$, $x = x_0 + x_1\zeta$ and $y = y_0 + y_1\zeta$

$$(x_0 + x_1\zeta)(y_0 + y_1\zeta) = (x_0y_0 + x_1y_1) + (x_0y_1 + x_1y_0 + x_1y_1)\zeta$$

$$\begin{aligned} xy \quad = \quad & -\operatorname{Tr}(1 \times x)\operatorname{Tr}(1 \times y) \cdot 1 - \operatorname{Tr}(\zeta \times x)\operatorname{Tr}(\zeta \times y) \cdot \zeta \\ & + \operatorname{Tr}((\zeta - 1) \times x)\operatorname{Tr}((\zeta - 1) \times y) \cdot (\zeta - 1) \end{aligned}$$

with

$$\begin{cases} \operatorname{Tr}(x)\operatorname{Tr}(y) & = \quad (x_0 - x_1)(y_0 - y_1) \\ \operatorname{Tr}((\zeta - 1)x)\operatorname{Tr}((\zeta - 1)y) & = \quad (x_0 + x_1)(y_0 + y_1) \\ \operatorname{Tr}(\zeta x)\operatorname{Tr}(\zeta y) & = \quad x_0y_0 \end{cases}$$

About trisymmetric decompositions

Link with other decompositions:

$$\mu_p(k) \leq \mu_p^{\mathrm{sym}}(k) \leq \mu_p^{\mathrm{tri}}(k)$$

About trisymmetric decompositions

Link with other decompositions:

$$\mu_p(k) \underset{?}{<} \mu_p^{\mathrm{sym}}(k) \underset{?}{<} \mu_p^{\mathrm{tri}}(k)$$

About trisymmetric decompositions

Link with other decompositions:

$$\mu_p(k) \underset{?}{<} \mu_p^{\mathrm{sym}}(k) \underset{?}{<} \mu_p^{\mathrm{tri}}(k)$$

Proposition (Randriambololona, '14)

Tri-symmetric decompositions always exist, except for $p = 2, m \geq 3$.

About trisymmetric decompositions

Link with other decompositions:

$$\mu_p(k) \underset{?}{<} \mu_p^{\text{sym}}(k) \underset{?}{<} \mu_p^{\text{tri}}(k)$$

Proposition (Randriambololona, '14)

Tri-symmetric decompositions always exist, except for $p = 2, m \geq 3$.

Results from [Randriambololona, R. '20]:

- **Asymptotics:** linearity in k can be obtained for **symmetric multilinear** decompositions in $\mathbb{F}_{p^k}$

About trisymmetric decompositions

Link with other decompositions:

$$\mu_p(k) \underset{?}{<} \mu_p^{\mathrm{sym}}(k) \underset{?}{<} \mu_p^{\mathrm{tri}}(k)$$

Proposition (Randriambololona, '14)

Tri-symmetric decompositions always exist, except for $p = 2, m \geq 3$.

Results from [Randriambololona, R. '20]:

- **Asymptotics:** linearity in k can be obtained for **symmetric multilinear** decompositions in $\mathbb{F}_{p^k}$
  - Corollary: $\mu_p^{\mathrm{tri}}(k)$ is also **linear** in k

About trisymmetric decompositions

Link with other decompositions:

$$\mu_p(k) \underset{?}{<} \mu_p^{\text{sym}}(k) \underset{?}{<} \mu_p^{\text{tri}}(k)$$

Proposition (Randriambololona, '14)

Tri-symmetric decompositions always exist, except for $p = 2, m \geq 3$.

Results from [Randriambololona, R. '20]:

- **Asymptotics:** linearity in k can be obtained for **symmetric multilinear** decompositions in $\mathbb{F}_{p^k}$
  - Corollary: $\mu_p^{\text{tri}}(k)$ is also **linear** in k
- **Small values:** usual algorithms do not work

About trisymmetric decompositions

Link with other decompositions:

$$\mu_p(k) \underset{?}{<} \mu_p^{\mathrm{sym}}(k) \underset{?}{<} \mu_p^{\mathrm{tri}}(k)$$

Proposition (Randriambololona, '14)

Tri-symmetric decompositions always exist, except for $p = 2, m \geq 3$.

Results from [Randriambololona, R. '20]:

- **Asymptotics:** linearity in k can be obtained for **symmetric multilinear** decompositions in $\mathbb{F}_{p^k}$
  - Corollary: $\mu_p^{\mathrm{tri}}(k)$ is also **linear** in k
- **Small values:** usual algorithms do not work
  - We provide an *ad hoc* exhaustive search algorithm

PARTIAL CONCLUSION

Results:

- Linearity of the **symmetric multilinear** complexity
- Linearity of the **trisymmetric** complexity
- New algorithm to find trisymmetric decompositions

Partial conclusion

Results:

- Linearity of the **symmetric multilinear** complexity
- Linearity of the **trisymmetric** complexity
- New algorithm to find trisymmetric decompositions

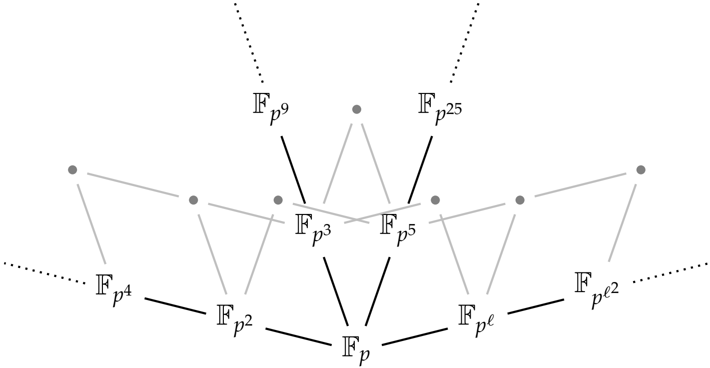
Future work:

- Find better bounds for the linearity of μ_p^{tri}
- Find algorithms exploiting the symmetries in the trisymmetric decompositions

Many extensions

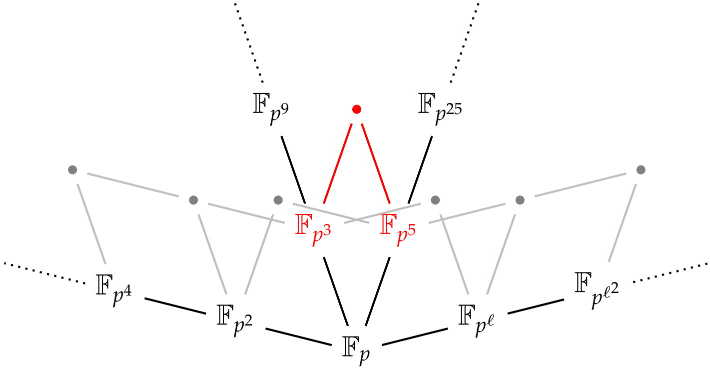
Context

- Use of Computer Algebra System (CAS)
- Use of many extensions of a prime finite field $\mathbb{F}_p$
- Computations in $\bar{\mathbb{F}}_p$.



Context

- Use of Computer Algebra System (CAS)
- Use of many extensions of a prime finite field $\mathbb{F}_p$
- Computations in $\bar{\mathbb{F}}_p$.



Embeddings

- When $k \mid l$, we know $\mathbb{F}_{p^k} \hookrightarrow \mathbb{F}_{p^l}$
  - How to compute an embedding **efficiently**?
  - There are several embeddings, **how to choose**?
- Naive algorithm: if $\mathbb{F}_{p^k} = \mathbb{F}_p[x]/(P(x))$, find a root ρ of P in $\mathbb{F}_{p^l}$ and map $\bar{x}$ to ρ. Complexity strictly larger than $\tilde{O}(k^2)$.
- Lots of other solutions in the litterature:
  - [Lenstra '91]
  - [Allombert '02]
  - [Rains '96]
  - [Narayanan '18]

Compatibility

- $\mathbb{F}_{p^k}, \mathbb{F}_{p^l}, \mathbb{F}_{p^m}$ three finite fields with $k \mid l \mid m$
- $f : \mathbb{F}_{p^k} \hookrightarrow \mathbb{F}_{p^l}, g : \mathbb{F}_{p^l} \hookrightarrow \mathbb{F}_{p^m}, h : \mathbb{F}_{p^k} \hookrightarrow \mathbb{F}_{p^m}$ embeddings

Compatibility:

$$\begin{tikzcd}
\mathbb{F}_{p^m} & \\
& \mathbb{F}_{p^l} \arrow[ul, "g"'] \\
\mathbb{F}_{p^k} \arrow[uu, "h"] \arrow[ur, "f"'] &
\end{tikzcd}$$

```
In:  p = 17; Fp = GF(p); FpX.<x> = Fp[]
     # We create finite fields of degree 12, 24, 48
     P12, P24 = x^12 + x + 2, x^24 + x^2 + 2*x + 7
     P48 = x^48 + x^2 + 2*x + 6
     GFp12 = FiniteField(p^12, 'x12', modulus=P12)
     GFp24 = FiniteField(p^24, 'x24', modulus=P24)
     GFp48 = FiniteField(p^48, 'x48', modulus=P48)
     # We (naively) compute the roots we need
     a = P12.any_root(GFp24) # Image of 'x12' in GFp24
     b = P24.any_root(GFp48) # Image of 'x24' in GFp48
     c = P12.any_root(GFp48) # Image of 'x12' in GFp48
     a # We print 'a'
Out: 6*x24^23 + 15*x24^22 + ... + 12*x24 + 16
```

Compatibility

- $\mathbb{F}_{p^k}, \mathbb{F}_{p^l}, \mathbb{F}_{p^m}$ three finite fields with $k \mid l \mid m$
- $f : \mathbb{F}_{p^k} \hookrightarrow \mathbb{F}_{p^l}$, $g : \mathbb{F}_{p^l} \hookrightarrow \mathbb{F}_{p^m}$, $h : \mathbb{F}_{p^k} \hookrightarrow \mathbb{F}_{p^m}$ embeddings

Compatibility:

$$\begin{tikzcd} \mathbb{F}_{p^m} & \\ & \mathbb{F}_{p^l} \arrow[ul, "g"'] \\ \mathbb{F}_{p^k} \arrow[uu, "h"] \arrow[ur, "f"'] & \end{tikzcd}$$

$$g \circ f \stackrel{?}{=} h$$

```
In:  p = 17; Fp = GF(p); FpX.<x> = Fp[]
     # We create finite fields of degree 12, 24, 48
     P12, P24 = x^12 + x + 2, x^24 + x^2 + 2*x + 7
     P48 = x^48 + x^2 + 2*x + 6
     GFp12 = FiniteField(p^12, 'x12', modulus=P12)
     GFp24 = FiniteField(p^24, 'x24', modulus=P24)
     GFp48 = FiniteField(p^48, 'x48', modulus=P48)
     # We (naively) compute the roots we need
     a = P12.any_root(GFp24) # Image of 'x12' in GFp24
     b = P24.any_root(GFp48) # Image of 'x24' in GFp48
     c = P12.any_root(GFp48) # Image of 'x12' in GFp48
     a # We print 'a'
Out: 6*x24^23 + 15*x24^22 + ... + 12*x24 + 16

     # We map 'x24' to 'b'
In:  c == a.polynomial()(b)
```

Compatibility

- $\mathbb{F}_{p^k}, \mathbb{F}_{p^l}, \mathbb{F}_{p^m}$ three finite fields with $k \mid l \mid m$
- $f : \mathbb{F}_{p^k} \hookrightarrow \mathbb{F}_{p^l}$, $g : \mathbb{F}_{p^l} \hookrightarrow \mathbb{F}_{p^m}$, $h : \mathbb{F}_{p^k} \hookrightarrow \mathbb{F}_{p^m}$ embeddings

Compatibility:

$$\begin{tikzcd} \mathbb{F}_{p^m} & \\ & \mathbb{F}_{p^l} \arrow[ul, "g"'] \\ \mathbb{F}_{p^k} \arrow[uu, "h"] \arrow[ur, "f"'] & \end{tikzcd}$$

$$g \circ f \stackrel{?}{=} h$$

```
In:  p = 17; Fp = GF(p); FpX.<x> = Fp[]
     # We create finite fields of degree 12, 24, 48
     P12, P24 = x^12 + x + 2, x^24 + x^2 + 2*x + 7
     P48 = x^48 + x^2 + 2*x + 6
     GFp12 = FiniteField(p^12, 'x12', modulus=P12)
     GFp24 = FiniteField(p^24, 'x24', modulus=P24)
     GFp48 = FiniteField(p^48, 'x48', modulus=P48)
     # We (naively) compute the roots we need
     a = P12.any_root(GFp24) # Image of 'x12' in GFp24
     b = P24.any_root(GFp48) # Image of 'x24' in GFp48
     c = P12.any_root(GFp48) # Image of 'x12' in GFp48
     a # We print 'a'
Out: 6*x24^23 + 15*x24^22 + ... + 12*x24 + 16

     # We map 'x24' to 'b'
In:  c == a.polynomial()(b)

Out: False
```

ENSURING COMPATIBILITY: CONWAY POLYNOMIALS

Definition (l-th Conway polynomials C_l)

- degree l, irreducible, monic
- primitive (*i.e.* its roots generate $\mathbb{F}_{p^l}^\times$)
- *norm-compatible* (*i.e.* $C_k\left(X^{\frac{p^l-1}{p^k-1}}\right) = 0 \mod C_l$ if $k \mid l$)

ENSURING COMPATIBILITY: CONWAY POLYNOMIALS

Definition (l-th Conway polynomials C_l)

- degree l, irreducible, monic
- primitive (*i.e.* its roots generate $\mathbb{F}_{p^l}^\times$)
- *norm-compatible* (*i.e.* $C_k\left(X^{\frac{p^l-1}{p^k-1}}\right) = 0 \mod C_l$ if $k \mid l$)

- **Standard polynomials**

Ensuring compatibility: Conway polynomials

Definition (l-th Conway polynomials C_l)

- degree l, irreducible, monic
- primitive (*i.e.* its roots generate $\mathbb{F}_{p^l}^{\times}$)
- *norm-compatible* (*i.e.* $C_k\left(X^{\frac{p^l-1}{p^k-1}}\right) = 0 \mod C_l$ if $k \mid l$)

- **Standard polynomials**
- Compatible embeddings: $\bar{X} \mapsto \bar{Y}^{\frac{p^l-1}{p^k-1}}$ $\tilde{O}(l^2)$

Ensuring compatibility: Conway polynomials

Definition (l-th Conway polynomials C_l)

- degree l, irreducible, monic
- primitive (*i.e.* its roots generate $\mathbb{F}_{p^l}^\times$)
- *norm-compatible* (*i.e.* $C_k\left(X^{\frac{p^l-1}{p^k-1}}\right) = 0 \mod C_l$ if $k \mid l$)

- **Standard polynomials**
- Compatible embeddings: $\bar{X} \mapsto \bar{Y}^{\frac{p^l-1}{p^k-1}}$ $\tilde{O}(l^2)$
- Hard to compute (exponential complexity)

Ensuring compatibility: Bosma, Cannon and Steel

- Framework originally used in MAGMA
- Based on the **naive embedding algorithm**
- Allows user-defined finite fields
- Computations made on the fly

COMMON SUBFIELD

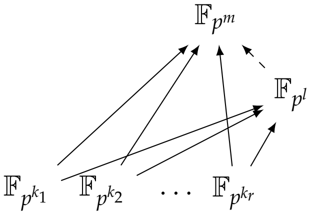
- Generalization of the naive algorithm

$$\begin{tikzcd} & \mathbb{F}_{p^m} \\ & & \mathbb{F}_{p^l} \arrow[ul, dashed] \\ \mathbb{F}_{p^k} \arrow[uur, "h"] \arrow[urr, "f"'] \end{tikzcd}$$

- Consider α such that $\mathbb{F}_{p^l} = \mathbb{F}_p(\alpha)$
- Take ρ a root of $h(\mathrm{minpoly}_{\mathbb{F}_{p^k}}(\alpha))$
- Map $\alpha \mapsto \rho$

We obtain $h = g \circ f$

SEVERAL SUBFIELDS



- Consider α such that $\mathbb{F}_{p^l} = \mathbb{F}_p(\alpha)$
- Take ρ a root of $\gcd_i(h_i(\mathrm{minpoly}_{\mathbb{F}_{p^{k_i}}}(\alpha)))$
- Map $\alpha \mapsto \rho$
- **This gives an embedding compatible with all subfields**

Implicit isomorphisms

From implicit isomorphisms come compatibility conditions

$$
\begin{tikzcd}
& \mathbb{F}_{p^{24}} & \\
& \mathbb{F}_{p^{12}} \arrow[u, dashed] & \\
\mathbb{F}_{p^4} \arrow[ur] \arrow[uur, dashed, bend left] & & \mathbb{F}_{p^6} \arrow[ul] \arrow[uul, dashed, bend right] \\
\cup & & \cup \\
\mathbb{F}_{p^2} & \cong & \mathbb{F}_{p^2}
\end{tikzcd}
$$

Implicit isomorphisms

From implicit isomorphisms come compatibility conditions

$$
\begin{tikzcd}
 & \mathbb{F}_{p^{24}} & \\
 & \mathbb{F}_{p^{12}} \arrow[u, dashed] & \\
\mathbb{F}_{p^4} \arrow[ur] \arrow[uur, dashed, bend left] & & \mathbb{F}_{p^6} \arrow[ul] \arrow[uul, dashed, bend right] \\
\mathbb{F}_{p^2} \arrow[u, phantom, "\cup"] & \cong & \mathbb{F}_{p^2} \arrow[u, phantom, "\cup"]
\end{tikzcd}
\qquad
\begin{tikzcd}
 & \mathbb{F}_{p^{12}} & \\
\mathbb{F}_{p^4} \arrow[ur, dashed] & & \mathbb{F}_{p^6} \arrow[ul]
\end{tikzcd}
$$

Implicit isomorphisms

From implicit isomorphisms come compatibility conditions

$$
\begin{tikzcd}
 & \mathbb{F}_{p^{24}} & \\
 & \mathbb{F}_{p^{12}} \arrow[u, dashed] & \\
\mathbb{F}_{p^4} \arrow[ur] \arrow[uur, dashed, bend left] & & \mathbb{F}_{p^6} \arrow[ul] \arrow[uul, dashed, bend right] \\
\cup & & \cup \\
\mathbb{F}_{p^2} & \cong & \mathbb{F}_{p^2}
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
 & \mathbb{F}_{p^{12}} & \\
\mathbb{F}_{p^4} \arrow[ur, dashed] & & \mathbb{F}_{p^6} \arrow[ul] \\
 & \mathbb{F}_{p^4} \cap \mathbb{F}_{p^6} = \mathbb{F}_{p^2} \arrow[ul, "?"] \arrow[ur, "?"'] &
\end{tikzcd}
$$

Computing the intersections

An example of what can happen with the intersections:

$$\begin{tikzcd}
 & & & \mathbb{F}_{p^{120}} & \\
 & & \mathbb{F}_{p^{60}} \arrow[ur, dashed] & & \mathbb{F}_{p^{40}} \arrow[ul] \\
 & \mathbb{F}_{p^{30}} \arrow[ur] & & & \\
\mathbb{F}_{p^{6}} \arrow[ur] & & & &
\end{tikzcd}$$

Computing the intersections

An example of what can happen with the intersections:

$$
\begin{tikzcd}
 & & & \mathbb{F}_{p^{120}} & \\
 & & \mathbb{F}_{p^{60}} \arrow[ur, dashed] & & \mathbb{F}_{p^{40}} \arrow[ul] \\
 & \mathbb{F}_{p^{30}} \arrow[ur] & & \mathbb{F}_{p^{20}} \arrow[ul, dashed] \arrow[ur] & \\
\mathbb{F}_{p^{6}} \arrow[ur] & & & &
\end{tikzcd}
$$

Computing the intersections

An example of what can happen with the intersections:

$$\begin{tikzcd}
 & & & \mathbb{F}_{p^{120}} & \\
 & & \mathbb{F}_{p^{60}} \arrow[ur, dashed] & & \mathbb{F}_{p^{40}} \arrow[ul] \\
 & \mathbb{F}_{p^{30}} \arrow[ur] & & \mathbb{F}_{p^{20}} \arrow[ul, dashed] \arrow[ur] & \\
\mathbb{F}_{p^{6}} \arrow[ur] & & \mathbb{F}_{p^{10}} \arrow[ul, dashed] \arrow[ur] & &
\end{tikzcd}$$

Computing the intersections

An example of what can happen with the intersections:

$$
\begin{tikzcd}
 & & & \mathbb{F}_{p^{120}} & \\
 & & \mathbb{F}_{p^{60}} \arrow[ur, dashed] & & \mathbb{F}_{p^{40}} \arrow[ul] \\
 & \mathbb{F}_{p^{30}} \arrow[ur] & & \mathbb{F}_{p^{20}} \arrow[ul, dashed] \arrow[ur] & \\
 \mathbb{F}_{p^{6}} \arrow[ur] & & \mathbb{F}_{p^{10}} \arrow[ul, dashed] \arrow[ur] & & \\
 & \mathbb{F}_{p^{2}} \arrow[ul] \arrow[ur] & & &
\end{tikzcd}
$$

COMPUTING THE INTERSECTIONS

An example of what can happen with the intersections:

$$
\begin{tikzcd}
 & & & \mathbb{F}_{p^{120}} & \\
 & & \mathbb{F}_{p^{60}} \arrow[ur, dashed] & & \mathbb{F}_{p^{40}} \arrow[ul] \\
 & \mathbb{F}_{p^{30}} \arrow[ur] & & \mathbb{F}_{p^{20}} \arrow[ul, dashed] \arrow[ur] & \\
\mathbb{F}_{p^{6}} \arrow[ur] & & \mathbb{F}_{p^{10}} \arrow[ul] \arrow[ur] & & \\
 & \mathbb{F}_{p^{2}} \arrow[ul] \arrow[ur] & & &
\end{tikzcd}
$$

Computing the intersections

An example of what can happen with the intersections:

$$
\begin{tikzcd}
 & & & \mathbb{F}_{p^{120}} & \\
 & & \mathbb{F}_{p^{60}} \arrow[ur, dashed] & & \mathbb{F}_{p^{40}} \arrow[ul] \\
 & \mathbb{F}_{p^{30}} \arrow[ur] & & \mathbb{F}_{p^{20}} \arrow[ul] \arrow[ur] & \\
\mathbb{F}_{p^{6}} \arrow[ur] & & \mathbb{F}_{p^{10}} \arrow[ul] \arrow[ur] & & \\
 & \mathbb{F}_{p^{2}} \arrow[ul] \arrow[ur] & & &
\end{tikzcd}
$$

Computing the intersections

An example of what can happen with the intersections:

$$\begin{tikzcd}
 & & & \mathbb{F}_{p^{120}} & \\
 & & \mathbb{F}_{p^{60}} \arrow[ur] & & \mathbb{F}_{p^{40}} \arrow[ul] \\
 & \mathbb{F}_{p^{30}} \arrow[ur] & & \mathbb{F}_{p^{20}} \arrow[ul] \arrow[ur] & \\
\mathbb{F}_{p^{6}} \arrow[ur] & & \mathbb{F}_{p^{10}} \arrow[ul] \arrow[ur] & & \\
 & \mathbb{F}_{p^{2}} \arrow[ul] \arrow[ur] & & &
\end{tikzcd}$$

RESULTS

- Following [De Feo, Randriambololona, R. '18], Bosma-Canon-Steel framework is now part of the free Computer Algebra System **Nemo**
- It is practical but
  - based on the naive embedding algorithm $\rightsquigarrow$ **superquadratic** complexity
  - adding an extension is quadratic in the size of the lattice

Goals:

- Change the embedding algorithm
- Lessen the cost of adding an extension

IDEAS

- Plugging Allombert's embedding algorithm in Bosma, Cannon, and Steel
- Generalizing Bosma, Cannon, and Steel
- Generalizing Conway polynomials

Bring the best of both worlds!

ALLOMBERT'S EMBEDDING ALGORITHM

- Based on **Kummer theory**
- For $k \mid (p-1)$, we work in $\mathbb{F}_{p^k}$, and study

$$\sigma(x) = \zeta_k x \tag{H90}$$

where $(\zeta_k)^k = 1$ and $\zeta_k \in \mathbb{F}_p \subset \mathbb{F}_{p^k}$

- When $k \mid l$ and $(\zeta_l)^{l/k} = \zeta_k$, from $\alpha_k \in \mathbb{F}_{p^k}, \alpha_l \in \mathbb{F}_{p^l}$ solutions of (H90), we can deduce an **embedding** of the form

$$\alpha_k \mapsto \kappa_{k,l}(\alpha_l)^{l/k}$$

with $\kappa_{k,l} \in \mathbb{F}_p$ a constant

ALLOMBERT AND BOSMA, CANON, AND STEEL

- Need to store one constant $\kappa_{k,l}$ for each pair $(\mathbb{F}_{p^k}, \mathbb{F}_{p^l})$
- The constant $\kappa_{k,l}$ depends on α_k and α_l

We would like to:

- get rid of the constants $\kappa_{k,l}$ (*e.g.* have $\kappa_{k,l} = 1$)
- equivalently, get "standard" solutions of (H90)
  - select solutions α_k, α_l that always define the same embedding
  - such that the constants $\kappa_{k,l}$ are well understood

Standard solutions

Let $k \mid l \mid p-1$, $(\zeta_l)^{l/k} = \zeta_k$

- $\alpha_k \in \mathbb{F}_{p^k}$ and $\alpha_l \in \mathbb{F}_{p^l}$ solutions of (H90) for ζ_k and ζ_l
- $(\forall k \mid l \mid p-1,\ \kappa_{k,l} = 1)$ implies $(\alpha_k)^k = (\alpha_l)^l = \zeta_{p-1}$
- We can use this property to define "standard solutions"

Standard solutions

Let $k \mid l \mid p-1$, $(\zeta_l)^{l/k} = \zeta_k$

- $\alpha_k \in \mathbb{F}_{p^k}$ and $\alpha_l \in \mathbb{F}_{p^l}$ solutions of (H90) for ζ_k and ζ_l
- $(\forall k \mid l \mid p-1,\ \kappa_{k,l} = 1)$ implies $(\alpha_k)^k = (\alpha_l)^l = \zeta_{p-1}$
- We can use this property to define "standard solutions"

Definition (Standard solution)

Let $k \mid p-1$ and $\alpha_k \in \mathbb{F}_{p^k}$ a solution of (H90) for $\zeta_k = (\zeta_{p-1})^{\frac{p-1}{k}}$, α_k is **standard** if $(\alpha_k)^k = \zeta_{p-1}$.

Standard solutions

Let $k \mid l \mid p-1$, $(\zeta_l)^{l/k} = \zeta_k$

- $\alpha_k \in \mathbb{F}_{p^k}$ and $\alpha_l \in \mathbb{F}_{p^l}$ solutions of (H90) for ζ_k and ζ_l
- $(\forall k \mid l \mid p-1,\ \kappa_{k,l} = 1)$ implies $(\alpha_k)^k = (\alpha_l)^l = \zeta_{p-1}$
- We can use this property to define "standard solutions"

Definition (Standard solution)

Let $k \mid p-1$ and $\alpha_k \in \mathbb{F}_{p^k}$ a solution of (H90) for $\zeta_k = (\zeta_{p-1})^{\frac{p-1}{k}}$, α_k is **standard** if $(\alpha_k)^k = \zeta_{p-1}$.

Definition (Standard polynomial)

All standard solutions α_k define the same irreducible polynomial of degree k, we call it the **standard polynomial** of degree k.

Standard embeddings

Let $k \mid l \mid p-1$, $(\zeta_l)^{l/k} = \zeta_k$

- α_k and α_l **standard solutions** of (H90) for ζ_k and ζ_l

STANDARD EMBEDDINGS

Let $k \mid l \mid p-1$, $(\zeta_l)^{l/k} = \zeta_k$

- α_k and α_l **standard solutions** of (H90) for ζ_k and ζ_l
  - $\kappa_{k,l} = 1$

Standard embeddings

Let $k \mid l \mid p-1$, $(\zeta_l)^{l/k} = \zeta_k$

- α_k and α_l **standard solutions** of (H90) for ζ_k and ζ_l
  - $\kappa_{k,l} = 1$
- The embedding

$$\alpha_k \mapsto (\alpha_l)^{l/k}$$

  is **standard** too (only depends on ζ_{p-1}).

WHAT HAPPENS WHEN $k \nmid p-1$?

Let $p \nmid k$ and $k \nmid p-1$

- no k-th root of unity ζ_k in $\mathbb{F}_p$
  - **add them!** Consider $A_k = \mathbb{F}_{p^k} \otimes \mathbb{F}_p(\zeta_k)$ instead of $\mathbb{F}_{p^k}$

$$(\sigma \otimes 1)(x) = (1 \otimes \zeta_k)x \tag{H90'}$$

- Allombert's algorithm still works!

If $k \mid l$ and $(\zeta_l)^{l/k} = \zeta_k$

- Still possible to find **standard solutions** α_k, α_l of (H90′)
- $\kappa_{k,l} \neq 1$ but easy to compute
- **Standard embedding** from α_k and α_l

SCHEME OF OUR WORK

$$
\begin{tikzcd}
& \mathbb{F}_{p^m} & \\
& & \mathbb{F}_{p^l} \arrow[ul, dashed] \\
\mathbb{F}_{p^k} \arrow[uur, dashed] \arrow[urr, dashed] & &
\end{tikzcd}
$$

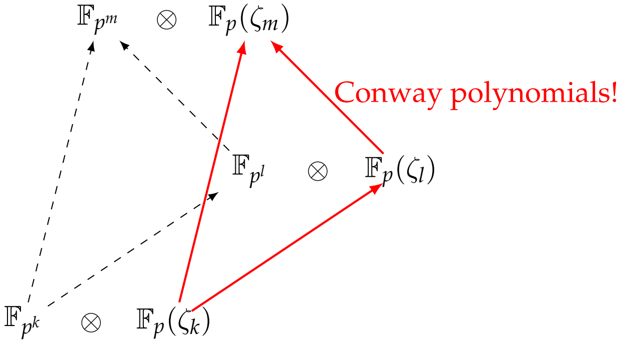
SCHEME OF OUR WORK

$$
\begin{tikzcd}
 & \mathbb{F}_{p^m} & \otimes & \mathbb{F}_p(\zeta_m) & & \\
 & & & \mathbb{F}_{p^l} \arrow[ull, dashed] & \otimes & \mathbb{F}_p(\zeta_l) \\
\mathbb{F}_{p^k} \arrow[uur, dashed] \arrow[urrr, dashed] & \otimes & \mathbb{F}_p(\zeta_k) & & &
\end{tikzcd}
$$

Scheme of our work

$$
\begin{tikzcd}
 & \mathbb{F}_{p^m} & \otimes & \mathbb{F}_p(\zeta_m) & & \\
 & & & \mathbb{F}_{p^l} \arrow[ull, dashed] & \otimes & \mathbb{F}_p(\zeta_l) \arrow[ull] \\
\mathbb{F}_{p^k} \arrow[uur, dashed] \arrow[urrr, dashed] & \otimes & \mathbb{F}_p(\zeta_k) \arrow[uur] \arrow[urrr] & & &
\end{tikzcd}
$$

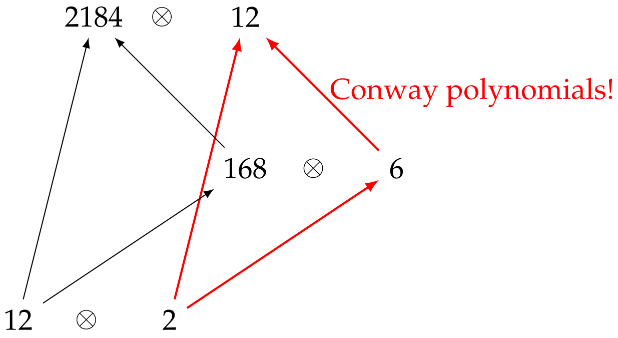
SCHEME OF OUR WORK



Scheme of our work

$$
\begin{tikzcd}
 & \mathbb{F}_{p^m} & \otimes & \mathbb{F}_p(\zeta_m) & & \\
 & & & \mathbb{F}_{p^l} \arrow[ull] & \otimes & \mathbb{F}_p(\zeta_l) \arrow[ull, red, "\text{Conway polynomials!}"'] \\
\mathbb{F}_{p^k} \arrow[uur] \arrow[urrr] & \otimes & \mathbb{F}_p(\zeta_k) \arrow[uur, red] \arrow[urrr, red] & & &
\end{tikzcd}
$$

Scheme of our work



Example of degrees involved in the case $p = 5$.

Compatibility and complexity

Results from [De Feo, Randriambololona, R. '19]:

Proposition (Compatibility)

Let $k \mid l \mid m$ and $f : \mathbb{F}_{p^k} \hookrightarrow \mathbb{F}_{p^l}$, $g : \mathbb{F}_{p^l} \hookrightarrow \mathbb{F}_{p^m}$, $h : \mathbb{F}_{p^k} \hookrightarrow \mathbb{F}_{p^m}$ the standard embeddings. Then we have $g \circ f = h$.

Proposition (Complexity)

Given a collection of Conway polynomials of degree up to d, for any $k \mid l \mid p^i - 1, i \leq d$

- *Computing a standard solution α_k takes $\tilde{O}(k^2)$*
- *Given α_k and α_l, computing the standard embedding $f : \mathbb{F}_{p^k} \hookrightarrow \mathbb{F}_{p^l}$ takes $\tilde{O}(l^2)$*

IMPLEMENTATION

Implementation using Flint/C and Nemo/Julia.

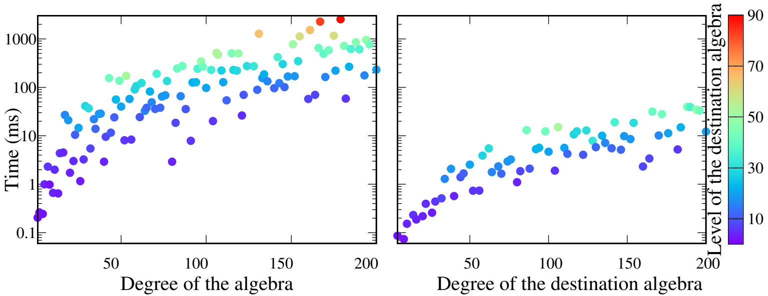


Figure: Timings for computing α_k (left, logscale), and for computing $\mathbb{F}_{p^2} \hookrightarrow \mathbb{F}_{p^k}$ (right, logscale) for $p = 3$.

Conclusion, open problems

- We implicitly assume that we have **compatible roots** ζ (*i.e.* $\zeta_k = (\zeta_l)^{l/k}$ for $k \mid l$)
  - In practice, this is done using **Conway polynomials**
- With Conway polynomials up to degree d, we can compute embeddings to finite fields up to any degree $k \mid p^i - 1, i \leq d$
  - quasi-quadratic complexity

Open problems:

- Make this work less standard, but more practical
- Can we replace the Conway polynomials by other polynomials?

Thank you!