



HAL
open science

Finding Optimal Formulae for Bilinear Maps

Razvan Barbulescu, Jérémie Detrey, Nicolas Estibals, Paul Zimmermann

► **To cite this version:**

Razvan Barbulescu, Jérémie Detrey, Nicolas Estibals, Paul Zimmermann. Finding Optimal Formulae for Bilinear Maps. AriC Seminar, Mar 2012, Lyon, France. hal-01413162

HAL Id: hal-01413162

<https://inria.hal.science/hal-01413162v1>

Submitted on 9 Dec 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Arén[^]H[^]H[^]HRIC seminar, LIP — March 22, 2012

Finding Optimal Formulae for Bilinear Maps

(slides courtesy of Nicolas Estibals)

Jérémie Detrey

CAMEL project-team, LORIA
INRIA Nancy – Grand Est
Jeremie.Detrey@loria.fr

Joint work with:

Răzvan Bărbulescu

Nicolas Estibals

Paul Zimmermann



A bit of history

- ▶ Multiplication is an **expensive** arithmetic operation
- ▶ Well-studied problem
 - **Karatsuba** (1962)
 - Toom–Cook (1963), **evaluation-interpolation** schemes
 - **CRT**-based algorithms
 - **Schönhage–Strassen** algorithm (1971)
 - ...
- ▶ *Five-, six-, and seven-term Karatsuba-like formulae*, P. Montgomery (2005)
 - **ad-hoc** formulae
 - **exhaustive search** for five-term multiplication
 - **non-exhaustive search** for six- and seven-term multiplications
 - (January 2011) start a task group to reproduce this work

Outline of the talk

- ▶ Formulae for polynomial multiplication
- ▶ Enumerating formulae
- ▶ Further improvements and heuristics
- ▶ Some results

Outline of the talk

- ▶ Formulae for polynomial multiplication
- ▶ Enumerating formulae
- ▶ Further improvements and heuristics
- ▶ Some results

Example: a 2-term polynomial times a 3-term one

► We want to compute

$$\begin{aligned}C(X) &= (a_1 \cdot X + a_0) \times (b_2 \cdot X^2 + b_1 \cdot X + b_0) \\ &= a_1 b_2 \cdot X^3 + (a_1 b_1 + a_0 b_2) \cdot X^2 + (a_0 b_1 + a_1 b_0) \cdot X + a_0 b_0\end{aligned}$$

Example: a 2-term polynomial times a 3-term one

- ▶ We want to compute

$$\begin{aligned}C(X) &= (a_1 \cdot X + a_0) \times (b_2 \cdot X^2 + b_1 \cdot X + b_0) \\ &= a_1 b_2 \cdot X^3 + (a_1 b_1 + a_0 b_2) \cdot X^2 + (a_0 b_1 + a_1 b_0) \cdot X + a_0 b_0\end{aligned}$$

- ▶ Only 5 products required instead of 6
 - use Karatsuba's trick

$$C(X) = a_1 b_2 \cdot X^3 + (a_1 b_1 + a_0 b_2) \cdot X^2 + ((a_0 + a_1)(b_0 + b_1) - a_1 b_1 - a_0 b_0) \cdot X + a_0 b_0$$

Example: a 2-term polynomial times a 3-term one

- ▶ We want to compute

$$\begin{aligned}C(X) &= (a_1 \cdot X + a_0) \times (b_2 \cdot X^2 + b_1 \cdot X + b_0) \\ &= a_1 b_2 \cdot X^3 + (a_1 b_1 + a_0 b_2) \cdot X^2 + (a_0 b_1 + a_1 b_0) \cdot X + a_0 b_0\end{aligned}$$

- ▶ Only 5 products required instead of 6
 - use Karatsuba's trick

$$C(X) = a_1 b_2 \cdot X^3 + (a_1 b_1 + a_0 b_2) \cdot X^2 + ((a_0 + a_1)(b_0 + b_1) - a_1 b_1 - a_0 b_0) \cdot X + a_0 b_0$$

- compute the products

$$\begin{aligned}g_0 &= a_0 \cdot b_0, \\ g_1 &= a_0 \cdot b_2, \\ g_2 &= a_1 \cdot b_1, \\ g_3 &= a_1 \cdot b_2, \text{ and} \\ g_4 &= (a_0 + a_1) \cdot (b_0 + b_1).\end{aligned}$$

- reconstruct the result

$$C(X) = g_3 \cdot X^3 + (g_1 + g_2) \cdot X^2 + (g_4 - g_2 - g_0) \cdot X + g_0$$

General form of a multiplication formula

► We want to compute, over a given field K (or any K -algebra \mathbf{K}),

$$(a_{n-1} \cdot X^{n-1} + \cdots + a_0) \times (b_{m-1} \cdot X^{m-1} + \cdots + b_0) = c_{n+m-2} \cdot X^{n+m-2} + \cdots + c_0$$

General form of a multiplication formula

- ▶ We want to compute, over a given field K (or any K -algebra \mathbf{K}),

$$(a_{n-1} \cdot X^{n-1} + \cdots + a_0) \times (b_{m-1} \cdot X^{m-1} + \cdots + b_0) = c_{n+m-2} \cdot X^{n+m-2} + \cdots + c_0$$

- ▶ All formulae for multiplication can be expressed as:

General form of a multiplication formula

- ▶ We want to compute, over a given field K (or any K -algebra \mathbf{K}),

$$(a_{n-1} \cdot X^{n-1} + \cdots + a_0) \times (b_{m-1} \cdot X^{m-1} + \cdots + b_0) = c_{n+m-2} \cdot X^{n+m-2} + \cdots + c_0$$

- ▶ All formulae for multiplication can be expressed as:

- compute some linear combinations of the a_i 's

$$a'_j = \sum \alpha_{i,j} \cdot a_i$$

- compute some linear combinations of the b_i 's

$$b'_j = \sum \beta_{i,j} \cdot b_i$$

General form of a multiplication formula

- ▶ We want to compute, over a given field K (or any K -algebra \mathbf{K}),

$$(a_{n-1} \cdot X^{n-1} + \cdots + a_0) \times (b_{m-1} \cdot X^{m-1} + \cdots + b_0) = c_{n+m-2} \cdot X^{n+m-2} + \cdots + c_0$$

- ▶ All formulae for multiplication can be expressed as:

- compute some linear combinations of the a_i 's

$$a'_j = \sum \alpha_{i,j} \cdot a_i$$

- compute some linear combinations of the b_i 's

$$b'_j = \sum \beta_{i,j} \cdot b_i$$

- perform some products

$$g_j = a'_j \cdot b'_j$$

General form of a multiplication formula

- ▶ We want to compute, over a given field K (or any K -algebra \mathbf{K}),

$$(a_{n-1} \cdot X^{n-1} + \cdots + a_0) \times (b_{m-1} \cdot X^{m-1} + \cdots + b_0) = c_{n+m-2} \cdot X^{n+m-2} + \cdots + c_0$$

- ▶ All formulae for multiplication can be expressed as:

- compute some linear combinations of the a_i 's

$$a'_j = \sum \alpha_{i,j} \cdot a_i$$

- compute some linear combinations of the b_i 's

$$b'_j = \sum \beta_{i,j} \cdot b_i$$

- perform some products

$$g_j = a'_j \cdot b'_j$$

- reconstruct the result by linear combinations of the products

$$c_k = \sum \gamma_{j,k} \cdot g_j$$

General form of a multiplication formula

- ▶ We want to compute, over a given field K (or any K -algebra \mathbf{K}),

$$(a_{n-1} \cdot X^{n-1} + \cdots + a_0) \times (b_{m-1} \cdot X^{m-1} + \cdots + b_0) = c_{n+m-2} \cdot X^{n+m-2} + \cdots + c_0$$

- ▶ All formulae for multiplication can be expressed as:

- compute some linear combinations of the a_i 's

$$a'_j = \sum \alpha_{i,j} \cdot a_i, \quad \text{with } \alpha_{i,j} \in K$$

- compute some linear combinations of the b_i 's

$$b'_j = \sum \beta_{i,j} \cdot b_i, \quad \text{with } \beta_{i,j} \in K$$

- perform some products

$$g_j = a'_j \cdot b'_j$$

- reconstruct the result by linear combinations of the products

$$c_k = \sum \gamma_{j,k} \cdot g_j, \quad \text{with } \gamma_{j,k} \in K$$

General form of a multiplication formula

- We want to compute, over a given field K (or any K -algebra \mathbf{K}),

$$(a_{n-1} \cdot X^{n-1} + \cdots + a_0) \times (b_{m-1} \cdot X^{m-1} + \cdots + b_0) = c_{n+m-2} \cdot X^{n+m-2} + \cdots + c_0$$

- All formulae for multiplication can be expressed as:

- compute some linear combinations of the a_i 's

$$a'_j = \sum \alpha_{i,j} \cdot a_i, \quad \text{with } \alpha_{i,j} \in K$$

- compute some linear combinations of the b_i 's

$$b'_j = \sum \beta_{i,j} \cdot b_i, \quad \text{with } \beta_{i,j} \in K$$

- perform some products

$$g_j = a'_j \cdot b'_j, \quad \text{with } a'_j, b'_j \in \mathbf{K}$$

- reconstruct the result by linear combinations of the products

$$c_k = \sum \gamma_{j,k} \cdot g_j, \quad \text{with } \gamma_{j,k} \in K$$

General form of a multiplication formula

- ▶ We want to compute, over a given field K (or any K -algebra \mathbf{K}),

$$(a_{n-1} \cdot X^{n-1} + \cdots + a_0) \times (b_{m-1} \cdot X^{m-1} + \cdots + b_0) = c_{n+m-2} \cdot X^{n+m-2} + \cdots + c_0$$

- ▶ All formulae for multiplication can be expressed as:

- compute some linear combinations of the a_i 's

$$a'_j = \sum \alpha_{i,j} \cdot a_i, \quad \text{with } \alpha_{i,j} \in K$$

- compute some linear combinations of the b_i 's

$$b'_j = \sum \beta_{i,j} \cdot b_i, \quad \text{with } \beta_{i,j} \in K$$

- perform some products

$$g_j = a'_j \cdot b'_j, \quad \text{with } a'_j, b'_j \in \mathbf{K}$$

- reconstruct the result by linear combinations of the products

$$c_k = \sum \gamma_{j,k} \cdot g_j, \quad \text{with } \gamma_{j,k} \in K$$

- ▶ This is also valid for any bilinear map

$$F : \quad K^n \quad \times \quad K^m \quad \longrightarrow \quad K^\ell \\ ((a_0, \dots, a_{n-1}) , (b_0, \dots, b_{m-1})) \longmapsto (c_0, \dots, c_{\ell-1})$$

Outline of the talk

- ▶ Formulae for polynomial multiplication
- ▶ Enumerating formulae
- ▶ Further improvements and heuristics
- ▶ Some results

Formal framework

$$F : K^n \times K^m \rightarrow K^\ell$$

Formal framework

$$F : K^n \times K^m \rightarrow K^\ell$$

- ▶ Represent the products and the coefficients of the result as elements of the nm -dimensional K -vector space

$$V = \text{Span } \mathcal{V}, \quad \text{with basis } \mathcal{V} = \{a_i b_j\}_{0 \leq i < n, 0 \leq j < m},$$

where the $a_i b_j$'s are seen as formal elements

Formal framework

$$F : K^n \times K^m \rightarrow K^\ell$$

- ▶ Represent the products and the coefficients of the result as elements of the nm -dimensional K -vector space

$$V = \text{Span } \mathcal{V}, \quad \text{with basis } \mathcal{V} = \{a_i b_j\}_{0 \leq i < n, 0 \leq j < m},$$

where the $a_i b_j$'s are seen as formal elements

- ▶ **Targets:** the coefficients of the result form a set

$$\mathcal{T} = \{c_i\}_{0 \leq i < \ell} \subset V$$

Formal framework

$$F : K^n \times K^m \rightarrow K^\ell$$

- ▶ Represent the products and the coefficients of the result as elements of the nm -dimensional K -vector space

$$V = \text{Span } \mathcal{V}, \quad \text{with basis } \mathcal{V} = \{a_i b_j\}_{0 \leq i < n, 0 \leq j < m},$$

where the $a_i b_j$'s are seen as formal elements

- ▶ **Targets:** the coefficients of the result form a set

$$\mathcal{T} = \{c_i\}_{0 \leq i < \ell} \subset V$$

- ▶ **Generators:** the set $\mathcal{G} \subset V$ of the **potential products** to use in a formula

$$\mathcal{G} = \{(\sum \alpha_i a_i) \cdot (\sum \beta_j b_j) \mid \forall i, \alpha_i \in K \text{ and } \forall j, \beta_j \in K\} \setminus \{0\}$$

Formal framework

$$F : K^n \times K^m \rightarrow K^\ell$$

- ▶ Represent the products and the coefficients of the result as elements of the nm -dimensional K -vector space

$$V = \text{Span } \mathcal{V}, \quad \text{with basis } \mathcal{V} = \{a_i b_j\}_{0 \leq i < n, 0 \leq j < m},$$

where the $a_i b_j$'s are seen as formal elements

- ▶ **Targets:** the coefficients of the result form a set

$$\mathcal{T} = \{c_i\}_{0 \leq i < \ell} \subset V$$

- ▶ **Generators:** the set $\mathcal{G} \subset V$ of the potential products to use in a formula
We only consider products modulo a nonzero scalar factor

$$\mathcal{G} = \{(\sum \alpha_i a_i) \cdot (\sum \beta_j b_j) \mid \forall i, \alpha_i \in K \text{ and } \forall j, \beta_j \in K\} \setminus \{0\} / \sim$$

where $g \sim g'$ when $\exists \lambda \in K, \lambda \neq 0$ such that $g = \lambda g'$

Example (cont'd)

Consider the previous example: 2×3 -term polynomial product in $\mathbb{F}_2[X]$

Example (cont'd)

Consider the previous example: 2×3 -term polynomial product in $\mathbb{F}_2[X]$

► V is a 6-dimensional vector space with basis

$$\mathcal{V} = \{a_0b_0, a_0b_1, a_0b_2, a_1b_0, a_1b_1, a_1b_2\}$$

Example (cont'd)

Consider the previous example: 2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ V is a 6-dimensional vector space with basis

$$\mathcal{V} = \{a_0b_0, a_0b_1, a_0b_2, a_1b_0, a_1b_1, a_1b_2\}$$

- ▶ The set of targets is

$$\mathcal{T} = \{a_1b_2, a_1b_1 + a_0b_2, a_0b_1 + a_1b_0, a_0b_0\}$$

Example (cont'd)

Consider the previous example: 2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ V is a 6-dimensional vector space with basis

$$\mathcal{V} = \{a_0b_0, a_0b_1, a_0b_2, a_1b_0, a_1b_1, a_1b_2\}$$

- ▶ The set of targets is

$$\mathcal{T} = \{a_1b_2, a_1b_1 + a_0b_2, a_0b_1 + a_1b_0, a_0b_0\}$$

- ▶ The set of generators contains 21 products:

$$\mathcal{G} = \left\{ \begin{array}{lll} a_0 \cdot b_0, & a_1 \cdot b_0, & (a_0 + a_1) \cdot b_0, \\ a_0 \cdot b_1, & a_1 \cdot b_1, & (a_0 + a_1) \cdot b_1, \\ a_0 \cdot (b_0 + b_1), & a_1 \cdot (b_0 + b_1), & (a_0 + a_1) \cdot (b_0 + b_1), \\ a_0 \cdot b_2, & a_1 \cdot b_2, & (a_0 + a_1) \cdot b_2, \\ a_0 \cdot (b_0 + b_2), & a_1 \cdot (b_0 + b_2), & (a_0 + a_1) \cdot (b_0 + b_2), \\ a_0 \cdot (b_1 + b_2), & a_1 \cdot (b_1 + b_2), & (a_0 + a_1) \cdot (b_1 + b_2), \\ a_0 \cdot (b_0 + b_1 + b_2), & a_1 \cdot (b_0 + b_1 + b_2), & (a_0 + a_1) \cdot (b_0 + b_1 + b_2) \end{array} \right\}$$

Naive algorithm

- ▶ Goal: find the optimal formulae (*i.e.*, with a minimum number of products)
 - enumerate the subsets $\mathcal{W} \subset \mathcal{G}$ of exactly k products which yield a valid formula
 - starting with $k = \text{rk}(\mathcal{T})$, increase k until a solution is found

Naive algorithm

- ▶ Goal: find the optimal formulae (*i.e.*, with a minimum number of products)
 - enumerate the subsets $\mathcal{W} \subset \mathcal{G}$ of exactly k products which yield a valid formula
 - starting with $k = \text{rk}(\mathcal{T})$, increase k until a solution is found
- ▶ Look for \mathcal{W} such that
 - \mathcal{W} is a set of k generators:

$$\mathcal{W} \subset \mathcal{G} \quad \text{and} \quad \#\mathcal{W} = k$$

- \mathcal{W} linearly generates the coefficients of the results:

$$\mathcal{T} \subset \text{Span } \mathcal{W}$$

Naive algorithm

- ▶ Goal: find the optimal formulae (*i.e.*, with a minimum number of products)
 - enumerate the subsets $\mathcal{W} \subset \mathcal{G}$ of exactly k products which yield a valid formula
 - starting with $k = \text{rk}(\mathcal{T})$, increase k until a solution is found
- ▶ Look for \mathcal{W} such that
 - \mathcal{W} is a set of k generators:

$$\mathcal{W} \subset \mathcal{G} \quad \text{and} \quad \#\mathcal{W} = k$$

- \mathcal{W} linearly generates the coefficients of the results:

$$\mathcal{T} \subset \text{Span } \mathcal{W}$$

- ▶ Naive approach:
 - enumerate the $\binom{\#\mathcal{G}}{k}$ subsets \mathcal{W} of size k
 - for each subset, test whether it generates \mathcal{T} or not

Naive algorithm

- ▶ Goal: find the optimal formulae (i.e., with a minimum number of products)
 - enumerate the subsets $\mathcal{W} \subset \mathcal{G}$ of exactly k products which yield a valid formula
 - starting with $k = \text{rk}(\mathcal{T})$, increase k until a solution is found

- ▶ Look for \mathcal{W} such that
 - \mathcal{W} is a set of k generators:

$$\mathcal{W} \subset \mathcal{G} \quad \text{and} \quad \#\mathcal{W} = k$$

- \mathcal{W} linearly generates the coefficients of the results:

$$\mathcal{T} \subset \text{Span } \mathcal{W}$$

- ▶ Naive approach:
 - enumerate the $\binom{\#\mathcal{G}}{k}$ subsets \mathcal{W} of size k
 - for each subset, test whether it generates \mathcal{T} or not

- ▶ \mathcal{G} has to be finite:
 - look for formulae over finite fields: $K = \mathbb{F}_q$ (typically, $q = 2, 3, 4$)
 - restrict to a finite subset of the generators \mathcal{G} (but search not exhaustive)

Naive algorithm

- ▶ Goal: find the optimal formulae (i.e., with a minimum number of products)
 - enumerate the subsets $\mathcal{W} \subset \mathcal{G}$ of exactly k products which yield a valid formula
 - starting with $k = \text{rk}(\mathcal{T})$, increase k until a solution is found

- ▶ Look for \mathcal{W} such that
 - \mathcal{W} is a set of k generators:

$$\mathcal{W} \subset \mathcal{G} \quad \text{and} \quad \#\mathcal{W} = k$$

- \mathcal{W} linearly generates the coefficients of the results:

$$\mathcal{T} \subset \text{Span } \mathcal{W}$$

- ▶ Naive approach:
 - enumerate the $\binom{\#\mathcal{G}}{k}$ subsets \mathcal{W} of size k
 - for each subset, test whether it generates \mathcal{T} or not

- ▶ \mathcal{G} has to be finite:
 - look for formulae over finite fields: $K = \mathbb{F}_q$ (typically, $q = 2, 3, 4$)
 - restrict to a finite subset of the generators \mathcal{G} (but search not exhaustive)

- ▶ Drawback: Distinct subsets may span the same subspace

Improved algorithm: subspaces instead of subsets

- ▶ Look instead for subspaces W of V such that
 - W can be generated by products only: $\text{Span}(W \cap \mathcal{G}) = W$
 - only k products are required: $\dim W = k$
 - W contains the target space spanned by the target vectors: $T = \text{Span } \mathcal{T} \subset W$

Improved algorithm: subspaces instead of subsets

- ▶ Look instead for **subspaces** W of V such that
 - W can be **generated by products** only: $\text{Span}(W \cap \mathcal{G}) = W$
 - only **k products** are required: $\dim W = k$
 - W contains the **target space** spanned by the target vectors: $T = \text{Span } \mathcal{T} \subset W$

- ▶ **Algorithm:**
 - 1: **procedure** expand_subspace(W)
 - 2: **if** $\dim W = k$ **and** $T \subset W$ **then**
 - 3: W is a solution
 - 4: **else if** $\dim W < k$ **then**
 - 5: **for each** $g \in \mathcal{G} \setminus W$ **do**
 - 6: expand_subspace($W \oplus \text{Span}(g)$)
 - 7: **end procedure**
 - 8: expand_subspace($\{\mathbf{0}\}$)

Improved algorithm: subspaces instead of subsets

- ▶ Look instead for subspaces W of V such that
 - W can be generated by products only: $\text{Span}(W \cap \mathcal{G}) = W$
 - only k products are required: $\dim W = k$
 - W contains the target space spanned by the target vectors: $T = \text{Span } \mathcal{T} \subset W$
- ▶ Algorithm:
 - 1: **procedure** expand_subspace(W)
 - 2: **if** $\dim W = k$ **and** $T \subset W$ **then**
 - 3: W is a solution
 - 4: **else if** $\dim W < k$ **then**
 - 5: **for each** $g \in \mathcal{G} \setminus W$ **do**
 - 6: expand_subspace($W \oplus \text{Span}(g)$)
 - 7: **end procedure**
 - 8: expand_subspace($\{\mathbf{0}\}$)
- ▶ Several formulae can correspond to a single solution subspace W
 - each basis of W comprising only elements of \mathcal{G} gives a k -product formula

Improved algorithm: incomplete basis

- ▶ We already know part of W !
 - target space T is a subspace of every solution space W
 - find each W by constructing $\mathcal{W}' \subset \mathcal{G}$ such that $W = T \oplus \text{Span } \mathcal{W}'$

Improved algorithm: incomplete basis

- ▶ We already know *part of W* !
 - target space T is a *subspace* of *every* solution space W
 - find each W by *constructing* $\mathcal{W}' \subset \mathcal{G}$ such that $W = T \oplus \text{Span } \mathcal{W}'$

- ▶ *Modified algorithm:*
 - 1: **procedure** `expand_subspace(W)`
 - 2: **if** $\dim W = k$ **and** $T \subset W$ **then**
 - 3: W is a solution
 - 4: **else if** $\dim W < k$ **then**
 - 5: **for each** $g \in \mathcal{G} \setminus W$ **do**
 - 6: `expand_subspace($W \oplus \text{Span}(g)$)`
 - 7: **end procedure**
 - 8: `expand_subspace($\{\mathbf{0}\}$)`

Improved algorithm: incomplete basis

- ▶ We already know **part of W** !
 - target space T is a **subspace** of **every** solution space W
 - find each W by **constructing** $\mathcal{W}' \subset \mathcal{G}$ such that $W = T \oplus \text{Span } \mathcal{W}'$

- ▶ **Modified algorithm:**
 - 1: **procedure** `expand_subspace(W)`
 - 2: **if** $\dim W = k$ **and** $T \subset W$ **then**
 - 3: W is a solution
 - 4: **else if** $\dim W < k$ **then**
 - 5: **for each** $g \in \mathcal{G} \setminus W$ **do**
 - 6: `expand_subspace($W \oplus \text{Span}(g)$)`
 - 7: **end procedure**
 - 8: `expand_subspace(T)`

Improved algorithm: incomplete basis

- ▶ We already know part of W !
 - target space T is a subspace of every solution space W
 - find each W by constructing $\mathcal{W}' \subset \mathcal{G}$ such that $W = T \oplus \text{Span } \mathcal{W}'$

- ▶ Modified algorithm:
 - 1: **procedure** expand_subspace(W)
 - 2: **if** $\dim W = k$ ~~and $T \in W$~~ **then**
 - 3: W is a solution
 - 4: **else if** $\dim W < k$ **then**
 - 5: **for each** $g \in \mathcal{G} \setminus W$ **do**
 - 6: expand_subspace($W \oplus \text{Span}(g)$)
 - 7: **end procedure**
 - 8: expand_subspace(T)

Improved algorithm: incomplete basis

- ▶ We already know **part of W** !
 - target space T is a **subspace** of **every** solution space W
 - find each W by **constructing** $\mathcal{W}' \subset \mathcal{G}$ such that $W = T \oplus \text{Span } \mathcal{W}'$

- ▶ **Modified algorithm:**
 - 1: **procedure** `expand_subspace(W)`
 - 2: **if** $\dim W = k$ **and** $\text{rk}(W \cap \mathcal{G}) = k$ **then**
 - 3: W is a solution
 - 4: **else if** $\dim W < k$ **then**
 - 5: **for each** $g \in \mathcal{G} \setminus W$ **do**
 - 6: `expand_subspace($W \oplus \text{Span}(g)$)`
 - 7: **end procedure**
 - 8: `expand_subspace(T)`

Improved algorithm: incomplete basis

- ▶ We already know **part of W** !
 - target space T is a **subspace** of **every** solution space W
 - find each W by **constructing** $\mathcal{W}' \subset \mathcal{G}$ such that $W = T \oplus \text{Span } \mathcal{W}'$

- ▶ **Modified algorithm:**

```
1: procedure expand_subspace( $W$ )
2:   if  $\dim W = k$  and  $\text{rk}(W \cap \mathcal{G}) = k$  then
3:      $W$  is a solution
4:   else if  $\dim W < k$  then
5:     for each  $g \in \mathcal{G} \setminus W$  do
6:       expand_subspace( $W \oplus \text{Span}(g)$ )
7:   end procedure
8: expand_subspace( $T$ )
```

- ▶ Complexity now depends on

$$\binom{\#\mathcal{G}}{k - \text{rk } \mathcal{T}}$$

Example (cont'd)

2×3 -term polynomial product in $\mathbb{F}_2[X]$

Example (cont'd)

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ Targets: $\mathcal{T} = \{a_1b_2, a_1b_1 + a_0b_2, a_0b_1 + a_1b_0, a_0b_0\}$
 - at least $\text{rk}(\mathcal{T}) = 4$ products required

Example (cont'd)

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ Targets: $\mathcal{T} = \{a_1b_2, a_1b_1 + a_0b_2, a_0b_1 + a_1b_0, a_0b_0\}$
 - at least $\text{rk}(\mathcal{T}) = 4$ products required

- ▶ Attempt with $k = 4$:
 - $W = \mathcal{T}$
 - $\mathcal{T} \cap \mathcal{G} = \{ a_0 \cdot b_0, a_1 \cdot b_2, (a_0 + a_1) \cdot (b_0 + b_1 + b_2) \}$

Example (cont'd)

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ Targets: $\mathcal{T} = \{a_1b_2, a_1b_1 + a_0b_2, a_0b_1 + a_1b_0, a_0b_0\}$
 - at least $\text{rk}(\mathcal{T}) = 4$ products required

- ▶ Attempt with $k = 4$:
 - $W = \mathcal{T}$
 - $\mathcal{T} \cap \mathcal{G} = \{a_0 \cdot b_0, a_1 \cdot b_2, (a_0 + a_1) \cdot (b_0 + b_1 + b_2)\}$
 - $\text{rk}(\mathcal{T} \cap \mathcal{G}) = 3 < k$
 - **no solutions** with $k = 4$ products only

Example (cont'd)

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ Targets: $\mathcal{T} = \{a_1b_2, a_1b_1 + a_0b_2, a_0b_1 + a_1b_0, a_0b_0\}$
 - at least $\text{rk}(\mathcal{T}) = 4$ products required
- ▶ Attempt with $k = 4$:
 - $W = \mathcal{T}$
 - $\mathcal{T} \cap \mathcal{G} = \{a_0 \cdot b_0, a_1 \cdot b_2, (a_0 + a_1) \cdot (b_0 + b_1 + b_2)\}$
 - $\text{rk}(\mathcal{T} \cap \mathcal{G}) = 3 < k$
 - **no solutions** with $k = 4$ products only
- ▶ Attempt with $k = 5$:
 - Try with $W = \mathcal{T} \oplus \text{Span}\{a_0b_1\}$
 - $W \cap \mathcal{G} = \left\{ \begin{array}{lll} a_0 \cdot b_0, & a_1 \cdot b_0, & (a_0 + a_1) \cdot b_0, \\ a_0 \cdot b_1, & a_1 \cdot b_2, & (a_0 + a_1) \cdot (b_1 + b_2), \\ a_0 \cdot (b_0 + b_1), & a_1 \cdot (b_0 + b_2), & (a_0 + a_1) \cdot (b_0 + b_1 + b_2) \end{array} \right\}$

Example (cont'd)

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ Targets: $\mathcal{T} = \{a_1b_2, a_1b_1 + a_0b_2, a_0b_1 + a_1b_0, a_0b_0\}$
 - at least $\text{rk}(\mathcal{T}) = 4$ products required
- ▶ Attempt with $k = 4$:
 - $W = \mathcal{T}$
 - $\mathcal{T} \cap \mathcal{G} = \{a_0 \cdot b_0, a_1 \cdot b_2, (a_0 + a_1) \cdot (b_0 + b_1 + b_2)\}$
 - $\text{rk}(\mathcal{T} \cap \mathcal{G}) = 3 < k$
 - **no solutions** with $k = 4$ products only
- ▶ Attempt with $k = 5$:
 - Try with $W = \mathcal{T} \oplus \text{Span}\{a_0b_1\}$
 - $W \cap \mathcal{G} = \left\{ \begin{array}{lll} a_0 \cdot b_0, & a_1 \cdot b_0, & (a_0 + a_1) \cdot b_0, \\ a_0 \cdot b_1, & a_1 \cdot b_2, & (a_0 + a_1) \cdot (b_1 + b_2), \\ a_0 \cdot (b_0 + b_1), & a_1 \cdot (b_0 + b_2), & (a_0 + a_1) \cdot (b_0 + b_1 + b_2) \end{array} \right\}$
 - $\text{rk}(W \cap \mathcal{G}) = 5 = k$, **W is a solution!**

Example (cont'd)

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ Targets: $\mathcal{T} = \{a_1b_2, a_1b_1 + a_0b_2, a_0b_1 + a_1b_0, a_0b_0\}$
 - at least $\text{rk}(\mathcal{T}) = 4$ products required
- ▶ Attempt with $k = 4$:
 - $W = \mathcal{T}$
 - $\mathcal{T} \cap \mathcal{G} = \{a_0 \cdot b_0, a_1 \cdot b_2, (a_0 + a_1) \cdot (b_0 + b_1 + b_2)\}$
 - $\text{rk}(\mathcal{T} \cap \mathcal{G}) = 3 < k$
 - **no solutions** with $k = 4$ products only
- ▶ Attempt with $k = 5$:
 - Try with $W = \mathcal{T} \oplus \text{Span}\{a_0b_1\}$
 - $W \cap \mathcal{G} = \left\{ \begin{array}{lll} a_0 \cdot b_0, & a_1 \cdot b_0, & (a_0 + a_1) \cdot b_0, \\ a_0 \cdot b_1, & a_1 \cdot b_2, & (a_0 + a_1) \cdot (b_1 + b_2), \\ a_0 \cdot (b_0 + b_1), & a_1 \cdot (b_0 + b_2), & (a_0 + a_1) \cdot (b_0 + b_1 + b_2) \end{array} \right\}$
 - $\text{rk}(W \cap \mathcal{G}) = 5 = k$, **W is a solution!**
 - $\{a_0b_0, a_1b_0, a_0b_1, a_1b_2, (a_0 + a_1)(b_1 + b_2)\}$ is a **basis** of W , and gives a **formula**

Example (cont'd)

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ Targets: $\mathcal{T} = \{a_1b_2, a_1b_1 + a_0b_2, a_0b_1 + a_1b_0, a_0b_0\}$
 - at least $\text{rk}(\mathcal{T}) = 4$ products required
- ▶ Attempt with $k = 4$:
 - $W = \mathcal{T}$
 - $\mathcal{T} \cap \mathcal{G} = \{a_0 \cdot b_0, a_1 \cdot b_2, (a_0 + a_1) \cdot (b_0 + b_1 + b_2)\}$
 - $\text{rk}(\mathcal{T} \cap \mathcal{G}) = 3 < k$
 - **no solutions** with $k = 4$ products only
- ▶ Attempt with $k = 5$:
 - Try with $W = \mathcal{T} \oplus \text{Span}\{a_0b_1\}$
 - $W \cap \mathcal{G} = \left\{ \begin{array}{lll} a_0 \cdot b_0, & a_1 \cdot b_0, & (a_0 + a_1) \cdot b_0, \\ a_0 \cdot b_1, & a_1 \cdot b_2, & (a_0 + a_1) \cdot (b_1 + b_2), \\ a_0 \cdot (b_0 + b_1), & a_1 \cdot (b_0 + b_2), & (a_0 + a_1) \cdot (b_0 + b_1 + b_2) \end{array} \right\}$
 - $\text{rk}(W \cap \mathcal{G}) = 5 = k$, **W is a solution!**
 - $\{a_0b_0, a_1b_0, a_0b_1, a_1b_2, (a_0 + a_1)(b_1 + b_2)\}$ is a **basis** of W , and gives a **formula**
 - there are **3 such solution spaces** which yield a total of **162 formulae**

A generic algorithm

- ▶ This algorithm will find all formulae with a prescribed number of products k
 - as long as we consider all the potential products in \mathcal{G}
 - can be used to prove lower bounds on the number of required products

A generic algorithm

- ▶ This algorithm will find **all formulae** with a prescribed number of products k
 - as long as we consider **all the potential products** in \mathcal{G}
 - can be used to **prove lower bounds** on the number of required products
- ▶ This algorithm works for **every bilinear maps**:
 - short products, middle products, cross products
 - multiplication in complexes, quaternions, field extensions, matrices
 - multiplication of **sparse** polynomials and matrices
 - etc.

A generic algorithm

- ▶ This algorithm will find **all formulae** with a prescribed number of products k
 - as long as we consider **all the potential products** in \mathcal{G}
 - can be used to **prove lower bounds** on the number of required products
- ▶ This algorithm works for **every bilinear maps**:
 - short products, middle products, cross products
 - multiplication in complexes, quaternions, field extensions, matrices
 - multiplication of **sparse** polynomials and matrices
 - etc.
- ▶ Also works for maps where the coefficients are **quadratic forms**
 - simply requires extending the definition of \mathcal{G} :

$$\mathcal{G} = \{(\sum \alpha_j \mathbf{a}_j) \cdot (\sum \alpha'_j \mathbf{a}_j) \mid (\alpha_0, \dots, \alpha_{n-1}) \preceq_{\text{lex}} (\alpha'_0, \dots, \alpha'_{n-1})\} \setminus \{\mathbf{0}\} / \sim$$

A generic algorithm

- ▶ This algorithm will find **all formulae** with a prescribed number of products k
 - as long as we consider **all the potential products** in \mathcal{G}
 - can be used to **prove lower bounds** on the number of required products
- ▶ This algorithm works for **every bilinear maps**:
 - short products, middle products, cross products
 - multiplication in complexes, quaternions, field extensions, matrices
 - multiplication of **sparse** polynomials and matrices
 - etc.
- ▶ Also works for maps where the coefficients are **quadratic forms**
 - simply requires extending the definition of \mathcal{G} :

$$\mathcal{G} = \{ (\sum \alpha_i a_i) \cdot (\sum \alpha'_i a_i) \mid (\alpha_0, \dots, \alpha_{n-1}) \preceq_{\text{lex}} (\alpha'_0, \dots, \alpha'_{n-1}) \} \setminus \{0\} / \sim$$

- example: **squaring of a 2-term polynomial** in $\mathbb{F}_3[X]$

$$\mathcal{G} = \{ a_0 \cdot a_0, \\ a_0 \cdot a_1, \quad a_1 \cdot (a_0 + a_1), \\ a_0 \cdot (a_0 + a_1), \quad a_1 \cdot (a_0 + a_1), \quad (a_0 + a_1) \cdot (a_0 + a_1), \\ a_0 \cdot (a_0 - a_1), \quad a_1 \cdot (a_0 - a_1), \quad (a_0 + a_1) \cdot (a_0 - a_1) \quad (a_0 - a_1) \cdot (a_0 - a_1) \}$$

Outline of the talk

- ▶ Formulae for polynomial multiplication
- ▶ Enumerating formulae
- ▶ Further improvements and heuristics
- ▶ Some results

Symmetric bilinear maps

- ▶ We consider only symmetric bilinear maps
 - same number of a_i 's and b_j 's: $F : K^n \times K^n \rightarrow K^\ell$
 - symmetry: $F(\mathbf{a}, \mathbf{b}) = F(\mathbf{b}, \mathbf{a})$
 - for instance, multiplication of two n -term polynomials

Symmetric bilinear maps

- ▶ We consider only **symmetric bilinear maps**
 - same number of a_i 's and b_j 's: $F : K^n \times K^n \rightarrow K^\ell$
 - **symmetry**: $F(\mathbf{a}, \mathbf{b}) = F(\mathbf{b}, \mathbf{a})$
 - for instance, **multiplication of two n -term polynomials**
- ▶ **Heuristic**: consider only products with **same linear combination** for the a_i 's and b_j 's

$$\mathcal{G}_{\text{sym}} = \{(\sum \alpha_i a_i) \cdot (\sum \alpha_i b_i) \mid \forall i, \alpha_i \in K\} \setminus \{0\} / \sim$$

Symmetric bilinear maps

- ▶ We consider only **symmetric bilinear maps**
 - same number of a_i 's and b_j 's: $F : K^n \times K^n \rightarrow K^\ell$
 - **symmetry**: $F(\mathbf{a}, \mathbf{b}) = F(\mathbf{b}, \mathbf{a})$
 - for instance, **multiplication of two n -term polynomials**
- ▶ **Heuristic**: consider only products with **same linear combination** for the a_i 's and b_j 's

$$\mathcal{G}_{\text{sym}} = \{(\sum \alpha_i a_i) \cdot (\sum \alpha_i b_i) \mid \forall i, \alpha_i \in K\} \setminus \{0\} / \sim$$

- 😊 reduce the number of generators: $\#\mathcal{G}_{\text{sym}} = \sqrt{\#\mathcal{G}}$
- ☹ formulae involving **asymmetric products** will not be found

Symmetric bilinear maps

- ▶ We consider only **symmetric bilinear maps**
 - same number of a_i 's and b_j 's: $F : K^n \times K^n \rightarrow K^\ell$
 - **symmetry**: $F(\mathbf{a}, \mathbf{b}) = F(\mathbf{b}, \mathbf{a})$
 - for instance, **multiplication of two n -term polynomials**
- ▶ **Heuristic**: consider only products with **same linear combination** for the a_i 's and b_j 's

$$\mathcal{G}_{\text{sym}} = \{(\sum \alpha_i a_i) \cdot (\sum \alpha_i b_i) \mid \forall i, \alpha_i \in K\} \setminus \{0\} / \sim$$

- ☺ reduce the number of generators: $\#\mathcal{G}_{\text{sym}} = \sqrt{\#\mathcal{G}}$
 - ☹ formulae involving **asymmetric products** will **not be found**
- ▶ Example: **2 × 2-term polynomial product** in $\mathbb{F}_3[X]$

$$\mathcal{G} = \left\{ \begin{array}{cccc} a_0 \cdot b_0, & a_1 \cdot b_0, & (a_0 + a_1) \cdot b_0, & (a_0 - a_1) \cdot b_0, \\ a_0 \cdot b_1, & a_1 \cdot b_1, & (a_0 + a_1) \cdot b_1, & (a_0 - a_1) \cdot b_1, \\ a_0 \cdot (b_0 + b_1), & a_1 \cdot (b_0 + b_1), & (a_0 + a_1) \cdot (b_0 + b_1), & (a_0 - a_1) \cdot (b_0 + b_1), \\ a_0 \cdot (b_0 - b_1), & a_1 \cdot (b_0 - b_1), & (a_0 + a_1) \cdot (b_0 - b_1), & (a_0 - a_1) \cdot (b_0 - b_1) \end{array} \right\}$$

Symmetric bilinear maps

- ▶ We consider only **symmetric bilinear maps**
 - same number of a_i 's and b_j 's: $F : K^n \times K^n \rightarrow K^\ell$
 - **symmetry**: $F(\mathbf{a}, \mathbf{b}) = F(\mathbf{b}, \mathbf{a})$
 - for instance, **multiplication of two n -term polynomials**

- ▶ **Heuristic**: consider only products with **same linear combination** for the a_i 's and b_j 's

$$\mathcal{G}_{\text{sym}} = \{(\sum \alpha_i a_i) \cdot (\sum \alpha_i b_i) \mid \forall i, \alpha_i \in K\} \setminus \{0\} / \sim$$

- ☺ reduce the number of generators: $\#\mathcal{G}_{\text{sym}} = \sqrt{\#\mathcal{G}}$
 - ☹ formulae involving **asymmetric products** will not be found
- ▶ Example: **2 × 2-term polynomial product** in $\mathbb{F}_3[X]$

$$\mathcal{G}_{\text{sym}} = \left\{ \begin{array}{cccc} a_0 \cdot b_0, & a_1 \cdot b_0, & (a_0 + a_1) \cdot b_0, & (a_0 - a_1) \cdot b_0, \\ a_0 \cdot b_1, & a_1 \cdot b_1, & (a_0 + a_1) \cdot b_1, & (a_0 - a_1) \cdot b_1, \\ a_0 \cdot (b_0 + b_1), & a_1 \cdot (b_0 + b_1), & (a_0 + a_1) \cdot (b_0 + b_1), & (a_0 - a_1) \cdot (b_0 + b_1), \\ a_0 \cdot (b_0 - b_1), & a_1 \cdot (b_0 - b_1), & (a_0 + a_1) \cdot (b_0 - b_1), & (a_0 - a_1) \cdot (b_0 - b_1) \end{array} \right\}$$

Orbits of a group action

Back to bilinear maps $F : K^n \times K^m \rightarrow K^\ell$

Orbits of a group action

Back to bilinear maps $F : K^n \times K^m \rightarrow K^\ell$

- ▶ Consider the vector space $V = \text{Span } \mathcal{V}$, with basis $\mathcal{V} = \{a_i b_j\}_{0 \leq i < n, 0 \leq j < m}$

Orbits of a group action

Back to bilinear maps $F : K^n \times K^m \rightarrow K^\ell$

- ▶ Consider the vector space $V = \text{Span } \mathcal{V}$, with basis $\mathcal{V} = \{a_i b_j\}_{0 \leq i < n, 0 \leq j < m}$
- ▶ Suppose there exists $\sigma \in \text{Aut}(V)$ such that
 - the set of generators $\mathcal{G} \subset V$ is fixed by σ : $\mathcal{G}^\sigma = \mathcal{G}$
 - the target space $T \subset V$ is also fixed by σ : $T^\sigma = T$

Orbits of a group action

Back to bilinear maps $F : K^n \times K^m \rightarrow K^\ell$

- ▶ Consider the vector space $V = \text{Span } \mathcal{V}$, with basis $\mathcal{V} = \{a_i b_j\}_{0 \leq i < n, 0 \leq j < m}$
- ▶ Suppose there exists $\sigma \in \text{Aut}(V)$ such that
 - the set of generators $\mathcal{G} \subset V$ is fixed by σ : $\mathcal{G}^\sigma = \mathcal{G}$
 - the target space $T \subset V$ is also fixed by σ : $T^\sigma = T$
 - for instance, for the $n \times m$ -term polynomial product over K :

$$\sigma : \quad a_i b_j \quad \longmapsto \quad a_{n-1-i} b_{m-1-j}$$

Orbits of a group action

Back to bilinear maps $F : K^n \times K^m \rightarrow K^\ell$

- ▶ Consider the vector space $V = \text{Span } \mathcal{V}$, with basis $\mathcal{V} = \{a_i b_j\}_{0 \leq i < n, 0 \leq j < m}$
- ▶ Suppose there exists $\sigma \in \text{Aut}(V)$ such that
 - the set of generators $\mathcal{G} \subset V$ is fixed by σ : $\mathcal{G}^\sigma = \mathcal{G}$
 - the target space $T \subset V$ is also fixed by σ : $T^\sigma = T$
 - for instance, for the $n \times m$ -term polynomial product over K :

$$\begin{aligned} \sigma : \quad a_i b_j &\longmapsto a_{n-1-i} b_{m-1-j} \\ (\sum \alpha_i a_i) \cdot (\sum \beta_j b_j) &\longmapsto (\sum \alpha_{n-1-i} a_i) \cdot (\sum \beta_{m-1-j} b_j) \end{aligned}$$

Orbits of a group action

Back to bilinear maps $F : K^n \times K^m \rightarrow K^\ell$

- ▶ Consider the vector space $V = \text{Span } \mathcal{V}$, with basis $\mathcal{V} = \{a_i b_j\}_{0 \leq i < n, 0 \leq j < m}$
- ▶ Suppose there exists $\sigma \in \text{Aut}(V)$ such that
 - the set of generators $\mathcal{G} \subset V$ is fixed by σ : $\mathcal{G}^\sigma = \mathcal{G}$
 - the target space $T \subset V$ is also fixed by σ : $T^\sigma = T$
 - for instance, for the $n \times m$ -term polynomial product over K :

$$\begin{aligned} \sigma : \quad a_i b_j &\longmapsto a_{n-1-i} b_{m-1-j} \\ (\sum \alpha_i a_i) \cdot (\sum \beta_j b_j) &\longmapsto (\sum \alpha_{n-1-i} a_i) \cdot (\sum \beta_{m-1-j} b_j) \end{aligned}$$

(corresponds to replacing X by $1/X$)

Orbits of a group action

Back to bilinear maps $F : K^n \times K^m \rightarrow K^\ell$

- ▶ Consider the vector space $V = \text{Span } \mathcal{V}$, with basis $\mathcal{V} = \{a_i b_j\}_{0 \leq i < n, 0 \leq j < m}$
- ▶ Suppose there exists $\sigma \in \text{Aut}(V)$ such that
 - the set of generators $\mathcal{G} \subset V$ is fixed by σ : $\mathcal{G}^\sigma = \mathcal{G}$
 - the target space $T \subset V$ is also fixed by σ : $T^\sigma = T$
 - for instance, for the $n \times m$ -term polynomial product over K :

$$\begin{aligned} \sigma : \quad a_i b_j &\longmapsto a_{n-1-i} b_{m-1-j} \\ (\sum \alpha_i a_i) \cdot (\sum \beta_j b_j) &\longmapsto (\sum \alpha_{n-1-i} a_i) \cdot (\sum \beta_{m-1-j} b_j) \end{aligned}$$

(corresponds to replacing X by $1/X$)

- ▶ Fact: if W is a solution space, then so is W^σ

Orbits of a group action

Back to bilinear maps $F : K^n \times K^m \rightarrow K^\ell$

- ▶ Consider the vector space $V = \text{Span } \mathcal{V}$, with basis $\mathcal{V} = \{a_i b_j\}_{0 \leq i < n, 0 \leq j < m}$
- ▶ Suppose there exists $\sigma \in \text{Aut}(V)$ such that
 - the set of generators $\mathcal{G} \subset V$ is fixed by σ : $\mathcal{G}^\sigma = \mathcal{G}$
 - the target space $T \subset V$ is also fixed by σ : $T^\sigma = T$
 - for instance, for the $n \times m$ -term polynomial product over K :

$$\begin{aligned} \sigma : \quad a_i b_j &\longmapsto a_{n-1-i} b_{m-1-j} \\ (\sum \alpha_i a_i) \cdot (\sum \beta_j b_j) &\longmapsto (\sum \alpha_{n-1-i} a_i) \cdot (\sum \beta_{m-1-j} b_j) \end{aligned}$$

(corresponds to replacing X by $1/X$)

- ▶ Fact: if W is a solution space, then so is W^σ
- ▶ Idea: group generators $g \in \mathcal{G}$ according to their σ -orbit

$$\bar{\mathcal{G}} = \{\bar{g} \mid g \in \mathcal{G}\}, \quad \text{where } \bar{g} = \{g' \in \mathcal{G} \mid g' = \sigma^i(g)\}$$

Orbits of a group action

Back to bilinear maps $F : K^n \times K^m \rightarrow K^\ell$

- ▶ Consider the vector space $V = \text{Span } \mathcal{V}$, with basis $\mathcal{V} = \{a_i b_j\}_{0 \leq i < n, 0 \leq j < m}$
- ▶ Suppose there exists $\sigma \in \text{Aut}(V)$ such that
 - the set of generators $\mathcal{G} \subset V$ is fixed by σ : $\mathcal{G}^\sigma = \mathcal{G}$
 - the target space $T \subset V$ is also fixed by σ : $T^\sigma = T$
 - for instance, for the $n \times m$ -term polynomial product over K :

$$\begin{aligned} \sigma : \quad a_i b_j &\longmapsto a_{n-1-i} b_{m-1-j} \\ (\sum \alpha_i a_i) \cdot (\sum \beta_j b_j) &\longmapsto (\sum \alpha_{n-1-i} a_i) \cdot (\sum \beta_{m-1-j} b_j) \end{aligned}$$

(corresponds to replacing X by $1/X$)

- ▶ Fact: if W is a solution space, then so is W^σ
- ▶ Idea: group generators $g \in \mathcal{G}$ according to their σ -orbit

$$\bar{\mathcal{G}} = \{\bar{g} \mid g \in \mathcal{G}\}, \quad \text{where } \bar{g} = \{g' \in \mathcal{G} \mid g' = \sigma^i(g)\}$$

- ▶ As long as W is fixed by σ , try only one generator per equivalence class

Orbits of a group action

Back to bilinear maps $F : K^n \times K^m \rightarrow K^\ell$

- ▶ Consider the vector space $V = \text{Span } \mathcal{V}$, with basis $\mathcal{V} = \{a_i b_j\}_{0 \leq i < n, 0 \leq j < m}$
- ▶ Suppose there exists $\sigma \in \text{Aut}(V)$ such that
 - the set of generators $\mathcal{G} \subset V$ is fixed by σ : $\mathcal{G}^\sigma = \mathcal{G}$
 - the target space $T \subset V$ is also fixed by σ : $T^\sigma = T$
 - for instance, for the $n \times m$ -term polynomial product over K :

$$\begin{aligned} \sigma : \quad a_i b_j &\longmapsto a_{n-1-i} b_{m-1-j} \\ (\sum \alpha_i a_i) \cdot (\sum \beta_j b_j) &\longmapsto (\sum \alpha_{n-1-i} a_i) \cdot (\sum \beta_{m-1-j} b_j) \end{aligned}$$

(corresponds to replacing X by $1/X$)

- ▶ Fact: if W is a solution space, then so is W^σ
- ▶ Idea: group generators $g \in \mathcal{G}$ according to their σ -orbit

$$\bar{\mathcal{G}} = \{\bar{g} \mid g \in \mathcal{G}\}, \quad \text{where } \bar{g} = \{g' \in \mathcal{G} \mid g' = \sigma^i(g)\}$$

- ▶ As long as W is fixed by σ , try only one generator per equivalence class
- ▶ Heuristic: add whole equivalence classes \bar{g} to W , instead of single generators

Orbits of a group action

Back to bilinear maps $F : K^n \times K^m \rightarrow K^\ell$

- ▶ Consider the vector space $V = \text{Span } \mathcal{V}$, with basis $\mathcal{V} = \{a_i b_j\}_{0 \leq i < n, 0 \leq j < m}$
- ▶ Suppose there exists $\sigma \in \text{Aut}(V)$ such that
 - the set of generators $\mathcal{G} \subset V$ is fixed by σ : $\mathcal{G}^\sigma = \mathcal{G}$
 - the target space $T \subset V$ is also fixed by σ : $T^\sigma = T$
 - for instance, for the $n \times m$ -term polynomial product over K :

$$\begin{aligned} \sigma : \quad a_i b_j &\longmapsto a_{n-1-i} b_{m-1-j} \\ (\sum \alpha_i a_i) \cdot (\sum \beta_j b_j) &\longmapsto (\sum \alpha_{n-1-i} a_i) \cdot (\sum \beta_{m-1-j} b_j) \end{aligned}$$

(corresponds to replacing X by $1/X$)

- ▶ Fact: if W is a solution space, then so is W^σ
- ▶ Idea: group generators $g \in \mathcal{G}$ according to their σ -orbit

$$\bar{\mathcal{G}} = \{\bar{g} \mid g \in \mathcal{G}\}, \quad \text{where } \bar{g} = \{g' \in \mathcal{G} \mid g' = \sigma^i(g)\}$$

- ▶ As long as W is fixed by σ , try only one generator per equivalence class
- ▶ Heuristic: add whole equivalence classes \bar{g} to W , instead of single generators
- ▶ Going further: instead of a single $\sigma \in \text{Aut}(V)$, take a whole subgroup $S \subset \text{Aut}(V)$

Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- \mathcal{G} and T are fixed by $S = \langle \sigma_0, \sigma_1 \rangle \subset \text{Aut}(V)$ with
- $\sigma_0 : a_i b_j \mapsto a_{1-i} b_{2-j}$ (i.e., replace X by $1/X$)
 - $\sigma_1 : a_i b_j \mapsto \left(\sum_{k=i}^1 \binom{k}{i} a_k \right) \cdot \left(\sum_{k=j}^2 \binom{k}{j} b_k \right)$ (i.e., replace X by $1 - X$)

Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ \mathcal{G} and T are fixed by $S = \langle \sigma_0, \sigma_1 \rangle \subset \text{Aut}(V)$ with
 - $\sigma_0 : a_i b_j \mapsto a_{1-i} b_{2-j}$ (i.e., replace X by $1/X$)
 - $\sigma_1 : a_i b_j \mapsto \left(\sum_{k=i}^1 \binom{k}{i} a_k \right) \cdot \left(\sum_{k=j}^2 \binom{k}{j} b_k \right)$ (i.e., replace X by $1 - X$)
- ▶ Remark that $\#S = 6$ (hurray for non-commutativity!)

Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ \mathcal{G} and T are fixed by $S = \langle \sigma_0, \sigma_1 \rangle \subset \text{Aut}(V)$ with
 - $\sigma_0 : a_i b_j \mapsto a_{1-i} b_{2-j}$ (i.e., replace X by $1/X$)
 - $\sigma_1 : a_i b_j \mapsto \left(\sum_{k=i}^1 \binom{k}{i} a_k \right) \cdot \left(\sum_{k=j}^2 \binom{k}{j} b_k \right)$ (i.e., replace X by $1 - X$)
- ▶ Remark that $\#S = 6$ (hurray for non-commutativity!)
- ▶ \mathcal{G} can be partitioned into

Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ \mathcal{G} and T are fixed by $S = \langle \sigma_0, \sigma_1 \rangle \subset \text{Aut}(V)$ with
 - $\sigma_0 : a_i b_j \mapsto a_{1-i} b_{2-j}$ (i.e., replace X by $1/X$)
 - $\sigma_1 : a_i b_j \mapsto \left(\sum_{k=i}^1 \binom{k}{i} a_k \right) \cdot \left(\sum_{k=j}^2 \binom{k}{j} b_k \right)$ (i.e., replace X by $1 - X$)
- ▶ Remark that $\#S = 6$ (hurray for non-commutativity!)
- ▶ \mathcal{G} can be partitioned into
 - 3 orbits of size 3:

$$\begin{aligned}\overline{g_0} &= \{ a_0 \cdot b_0, & a_1 \cdot b_2, & (a_0 + a_1) \cdot (b_0 + b_1 + b_2) \}, \\ \overline{g_1} &= \{ a_0 \cdot b_1, & a_1 \cdot b_1, & (a_0 + a_1) \cdot b_1 \}, \\ \overline{g_2} &= \{ a_0 \cdot (b_0 + b_1), & a_1 \cdot (b_1 + b_2), & (a_0 + a_1) \cdot (b_0 + b_2) \}, \text{ and}\end{aligned}$$

Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ \mathcal{G} and T are fixed by $S = \langle \sigma_0, \sigma_1 \rangle \subset \text{Aut}(V)$ with
 - $\sigma_0 : a_i b_j \mapsto a_{1-i} b_{2-j}$ (i.e., replace X by $1/X$)
 - $\sigma_1 : a_i b_j \mapsto \left(\sum_{k=i}^1 \binom{k}{i} a_k \right) \cdot \left(\sum_{k=j}^2 \binom{k}{j} b_k \right)$ (i.e., replace X by $1 - X$)
- ▶ Remark that $\#S = 6$ (hurray for non-commutativity!)
- ▶ \mathcal{G} can be partitioned into
 - 3 orbits of size 3:

$$\begin{aligned} \overline{g_0} &= \{ a_0 \cdot b_0, & a_1 \cdot b_2, & (a_0 + a_1) \cdot (b_0 + b_1 + b_2) \}, \\ \overline{g_1} &= \{ a_0 \cdot b_1, & a_1 \cdot b_1, & (a_0 + a_1) \cdot b_1 \}, \\ \overline{g_2} &= \{ a_0 \cdot (b_0 + b_1), & a_1 \cdot (b_1 + b_2), & (a_0 + a_1) \cdot (b_0 + b_2) \}, \text{ and} \end{aligned}$$

- 2 orbits of size 6:

$$\begin{aligned} \overline{g_3} &= \{ a_0 \cdot b_2, & a_0 \cdot (b_0 + b_1 + b_2), & (a_0 + a_1) \cdot b_0, \\ & a_1 \cdot b_0, & a_1 \cdot (b_0 + b_1 + b_2), & (a_0 + a_1) \cdot b_2 \}, \\ \overline{g_4} &= \{ a_0 \cdot (b_0 + b_2), & a_0 \cdot (b_1 + b_2), & (a_0 + a_1) \cdot (b_0 + b_1), \\ & a_1 \cdot (b_0 + b_2), & a_1 \cdot (b_0 + b_1), & (a_0 + a_1) \cdot (b_1 + b_2) \} \end{aligned}$$

Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ First, consider adding to T a generator from orbit $\overline{g_1}$:

$$\sigma_0 \curvearrowright (a_0 + a_1) \cdot b_1 \xleftrightarrow{\sigma_1} a_0 \cdot b_1 \xleftrightarrow{\sigma_0} a_1 \cdot b_1 \curvearrowleft \sigma_1$$

Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ First, consider adding to T a generator from orbit $\overline{g_1}$:

$$\sigma_0 \curvearrowright (a_0 + a_1) \cdot b_1 \xleftrightarrow{\sigma_1} a_0 \cdot b_1 \xleftrightarrow{\sigma_0} a_1 \cdot b_1 \curvearrowleft \sigma_1$$

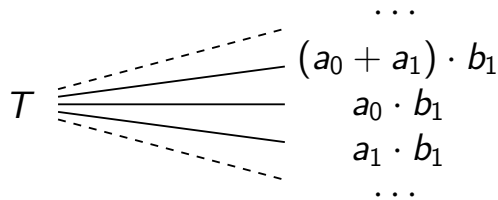
T

Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ First, consider adding to T a generator from orbit $\overline{g_1}$:

$$\sigma_0 \curvearrowright (a_0 + a_1) \cdot b_1 \xleftrightarrow{\sigma_1} a_0 \cdot b_1 \xleftrightarrow{\sigma_0} a_1 \cdot b_1 \curvearrowleft \sigma_1$$

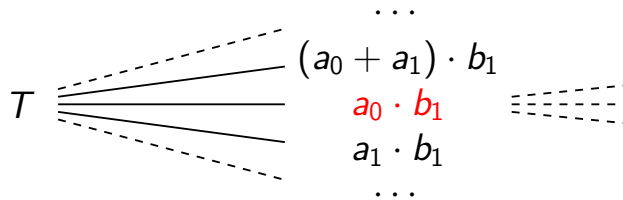


Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ First, consider adding to T a generator from orbit $\overline{g_1}$:

$$\sigma_0 \curvearrowright (a_0 + a_1) \cdot b_1 \xleftrightarrow{\sigma_1} a_0 \cdot b_1 \xleftrightarrow{\sigma_0} a_1 \cdot b_1 \curvearrowleft \sigma_1$$



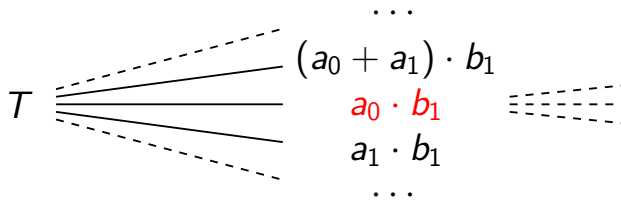
Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ First, consider adding to T a generator from orbit $\overline{g_1}$:

$$\sigma_0 \curvearrowright (a_0 + a_1) \cdot b_1 \xleftrightarrow{\sigma_1} a_0 \cdot b_1 \xleftrightarrow{\sigma_0} a_1 \cdot b_1 \curvearrowleft \sigma_1$$

- ▶ For any solution of the form $W = T \oplus \text{Span}(a_0 \cdot b_1) \oplus \text{Span } \mathcal{W}'$, with $\mathcal{W}' \subset \mathcal{G}$:



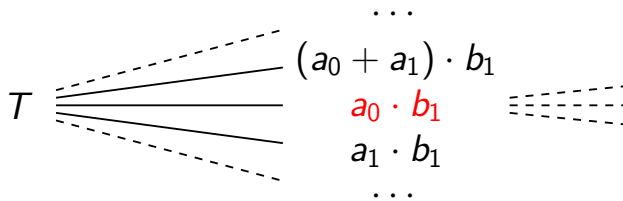
Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ First, consider adding to T a generator from orbit $\overline{g_1}$:

$$\sigma_0 \curvearrowright (a_0 + a_1) \cdot b_1 \xleftrightarrow{\sigma_1} a_0 \cdot b_1 \xleftrightarrow{\sigma_0} a_1 \cdot b_1 \curvearrowleft \sigma_1$$

- ▶ For any solution of the form $W = T \oplus \text{Span}(a_0 \cdot b_1) \oplus \text{Span } \mathcal{W}'$, with $\mathcal{W}' \subset \mathcal{G}$:
 - $W^{\sigma_1} = T \oplus \text{Span}((a_0 + a_1) \cdot b_1) \oplus \mathcal{W}'^{\sigma_1}$, with $\mathcal{W}'^{\sigma_1} \subset \mathcal{G}$, is also a solution



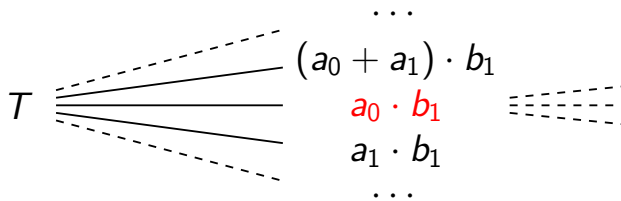
Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ First, consider adding to T a generator from orbit $\overline{g_1}$:

$$\sigma_0 \curvearrowright (a_0 + a_1) \cdot b_1 \xleftrightarrow{\sigma_1} a_0 \cdot b_1 \xleftrightarrow{\sigma_0} a_1 \cdot b_1 \curvearrowleft \sigma_1$$

- ▶ For any solution of the form $W = T \oplus \text{Span}(a_0 \cdot b_1) \oplus \text{Span } \mathcal{W}'$, with $\mathcal{W}' \subset \mathcal{G}$:
 - $W^{\sigma_1} = T \oplus \text{Span}((a_0 + a_1) \cdot b_1) \oplus \mathcal{W}'^{\sigma_1}$, with $\mathcal{W}'^{\sigma_1} \subset \mathcal{G}$, is also a solution
 - thus, no need to enumerate both $(a_0 + a_1) \cdot b_1$ and $a_0 \cdot b_1$



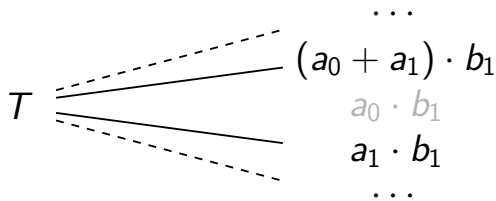
Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ First, consider adding to T a generator from orbit $\overline{g_1}$:

$$\sigma_0 \curvearrowright (a_0 + a_1) \cdot b_1 \xleftrightarrow{\sigma_1} a_0 \cdot b_1 \xleftrightarrow{\sigma_0} a_1 \cdot b_1 \curvearrowleft \sigma_1$$

- ▶ For any solution of the form $W = T \oplus \text{Span}(a_0 \cdot b_1) \oplus \text{Span } \mathcal{W}'$, with $\mathcal{W}' \subset \mathcal{G}$:
 - $W^{\sigma_1} = T \oplus \text{Span}((a_0 + a_1) \cdot b_1) \oplus \mathcal{W}'^{\sigma_1}$, with $\mathcal{W}'^{\sigma_1} \subset \mathcal{G}$, is also a solution
 - thus, no need to enumerate both $(a_0 + a_1) \cdot b_1$ and $a_0 \cdot b_1$



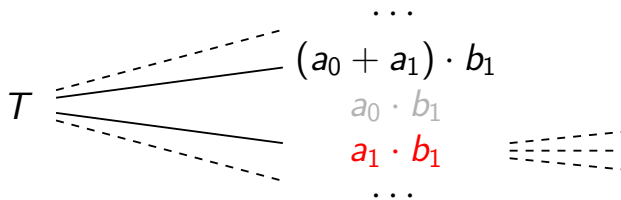
Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ First, consider adding to T a generator from orbit $\overline{g_1}$:

$$\sigma_0 \curvearrowright (a_0 + a_1) \cdot b_1 \xleftrightarrow{\sigma_1} a_0 \cdot b_1 \xleftrightarrow{\sigma_0} a_1 \cdot b_1 \curvearrowleft \sigma_1$$

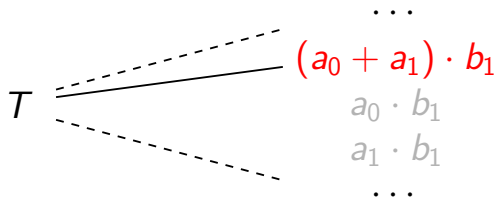
- ▶ For any solution of the form $W = T \oplus \text{Span}(a_0 \cdot b_1) \oplus \text{Span } \mathcal{W}'$, with $\mathcal{W}' \subset \mathcal{G}$:
 - $W^{\sigma_1} = T \oplus \text{Span}((a_0 + a_1) \cdot b_1) \oplus \mathcal{W}'^{\sigma_1}$, with $\mathcal{W}'^{\sigma_1} \subset \mathcal{G}$, is also a solution
 - thus, no need to enumerate both $(a_0 + a_1) \cdot b_1$ and $a_0 \cdot b_1$
- ▶ Similarly, for any solution $W = T \oplus \text{Span}(a_1 \cdot b_1) \oplus \text{Span } \mathcal{W}'$, with $\mathcal{W}' \subset \mathcal{G}$:
 - $W^{\sigma_0\sigma_1} = T \oplus \text{Span}((a_0 + a_1) \cdot b_1) \oplus \mathcal{W}'^{\sigma_0\sigma_1}$, with $\mathcal{W}'^{\sigma_0\sigma_1} \subset \mathcal{G}$, is also a solution



Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ So we add generator $(a_0 + a_1) \cdot b_1$, fixed by σ_0

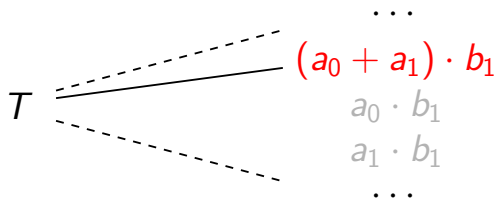


Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ So we add generator $(a_0 + a_1) \cdot b_1$, fixed by σ_0
- ▶ Now, consider adding to $T \oplus \text{Span}((a_0 + a_1) \cdot b_1)$ a generator from orbit $\overline{g_3}$:

$$\begin{array}{ccccc}
 a_0 \cdot b_2 & \xleftrightarrow{\sigma_0} & a_1 \cdot b_0 & \xleftrightarrow{\sigma_1} & a_1 \cdot (b_0 + b_1 + b_2) \\
 \sigma_1 \uparrow \downarrow & & & & \uparrow \downarrow \sigma_0 \\
 (a_0 + a_1) \cdot b_2 & \xleftrightarrow{\sigma_0} & (a_0 + a_1) \cdot b_0 & \xleftrightarrow{\sigma_1} & a_0 \cdot (b_0 + b_1 + b_2)
 \end{array}$$

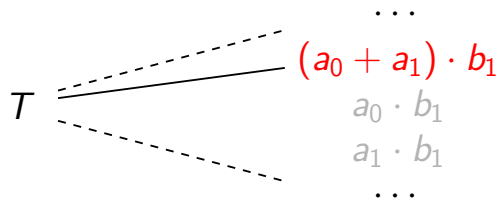


Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ So we add generator $(a_0 + a_1) \cdot b_1$, fixed by σ_0
- ▶ Now, consider adding to $T \oplus \text{Span}((a_0 + a_1) \cdot b_1)$ a generator from orbit $\overline{g_3}$:

$$\begin{array}{ccccc}
 a_0 \cdot b_2 & \xleftrightarrow{\sigma_0} & a_1 \cdot b_0 & \xleftrightarrow{\sigma_1} & a_1 \cdot (b_0 + b_1 + b_2) \\
 \sigma_1 \updownarrow & & & & \updownarrow \sigma_0 \\
 (a_0 + a_1) \cdot b_2 & \xleftrightarrow{\sigma_0} & (a_0 + a_1) \cdot b_0 & \xleftrightarrow{\sigma_1} & a_0 \cdot (b_0 + b_1 + b_2)
 \end{array}$$

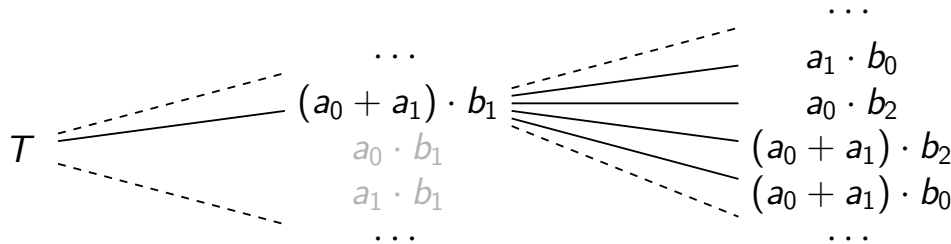


Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ So we add generator $(a_0 + a_1) \cdot b_1$, fixed by σ_0
- ▶ Now, consider adding to $T \oplus \text{Span}((a_0 + a_1) \cdot b_1)$ a generator from orbit $\overline{g_3}$:

$$\begin{array}{ccccc}
 a_0 \cdot b_2 & \xleftrightarrow{\sigma_0} & a_1 \cdot b_0 & \xleftrightarrow{\sigma_1} & a_1 \cdot (b_0 + b_1 + b_2) \\
 \sigma_1 \uparrow \downarrow & & & & \uparrow \downarrow \sigma_0 \\
 (a_0 + a_1) \cdot b_2 & \xleftrightarrow{\sigma_0} & (a_0 + a_1) \cdot b_0 & \xleftrightarrow{\sigma_1} & a_0 \cdot (b_0 + b_1 + b_2)
 \end{array}$$



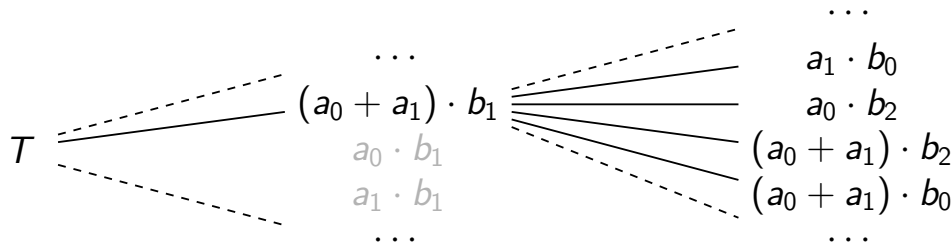
Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ So we add generator $(a_0 + a_1) \cdot b_1$, fixed by σ_0
- ▶ Now, consider adding to $T \oplus \text{Span}((a_0 + a_1) \cdot b_1)$ a generator from orbit $\overline{g_3}$:

$$\begin{array}{ccccc}
 a_0 \cdot b_2 & \xleftrightarrow{\sigma_0} & a_1 \cdot b_0 & \xleftrightarrow{\sigma_1} & a_1 \cdot (b_0 + b_1 + b_2) \\
 \sigma_1 \uparrow \downarrow & & & & \uparrow \downarrow \sigma_0 \\
 (a_0 + a_1) \cdot b_2 & \xleftrightarrow{\sigma_0} & (a_0 + a_1) \cdot b_0 & \xleftrightarrow{\sigma_1} & a_0 \cdot (b_0 + b_1 + b_2)
 \end{array}$$

- ▶ Since $T \oplus \text{Span}((a_0 + a_1) \cdot b_1)$ is fixed by σ_0 , we can still partition $\overline{g_3}$ according to its σ_0 -orbits



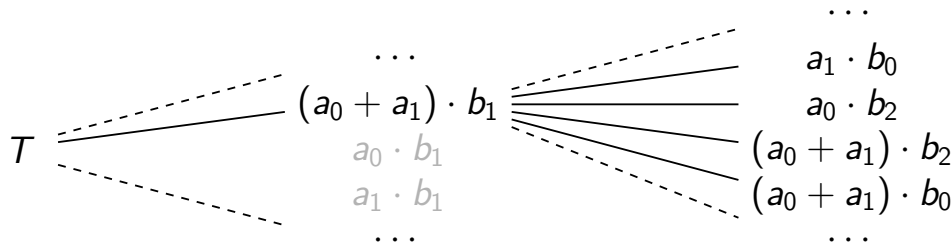
Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ So we add generator $(a_0 + a_1) \cdot b_1$, fixed by σ_0
- ▶ Now, consider adding to $T \oplus \text{Span}((a_0 + a_1) \cdot b_1)$ a generator from orbit $\overline{g_3}$:

$$\begin{array}{ccccc}
 a_0 \cdot b_2 & \xleftrightarrow{\sigma_0} & a_1 \cdot b_0 & \xleftrightarrow{\sigma_1} & a_1 \cdot (b_0 + b_1 + b_2) \\
 \sigma_1 \uparrow \downarrow & & & & \uparrow \downarrow \sigma_0 \\
 (a_0 + a_1) \cdot b_2 & \xleftrightarrow{\sigma_0} & (a_0 + a_1) \cdot b_0 & \xleftrightarrow{\sigma_1} & a_0 \cdot (b_0 + b_1 + b_2)
 \end{array}$$

- ▶ Since $T \oplus \text{Span}((a_0 + a_1) \cdot b_1)$ is fixed by σ_0 , we can still partition $\overline{g_3}$ according to its σ_0 -orbits



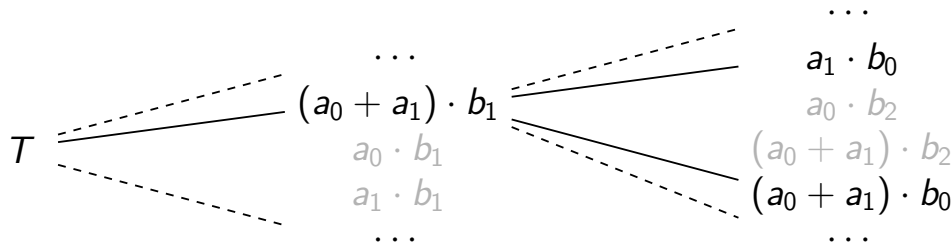
Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ So we add generator $(a_0 + a_1) \cdot b_1$, fixed by σ_0
- ▶ Now, consider adding to $T \oplus \text{Span}((a_0 + a_1) \cdot b_1)$ a generator from orbit $\overline{g_3}$:

$$\begin{array}{ccccc}
 a_0 \cdot b_2 & \xleftrightarrow{\sigma_0} & a_1 \cdot b_0 & \xleftrightarrow{\sigma_1} & a_1 \cdot (b_0 + b_1 + b_2) \\
 \sigma_1 \uparrow \downarrow & & & & \uparrow \downarrow \sigma_0 \\
 (a_0 + a_1) \cdot b_2 & \xleftrightarrow{\sigma_0} & (a_0 + a_1) \cdot b_0 & \xleftrightarrow{\sigma_1} & a_0 \cdot (b_0 + b_1 + b_2)
 \end{array}$$

- ▶ Since $T \oplus \text{Span}((a_0 + a_1) \cdot b_1)$ is fixed by σ_0 , we can still partition $\overline{g_3}$ according to its σ_0 -orbits



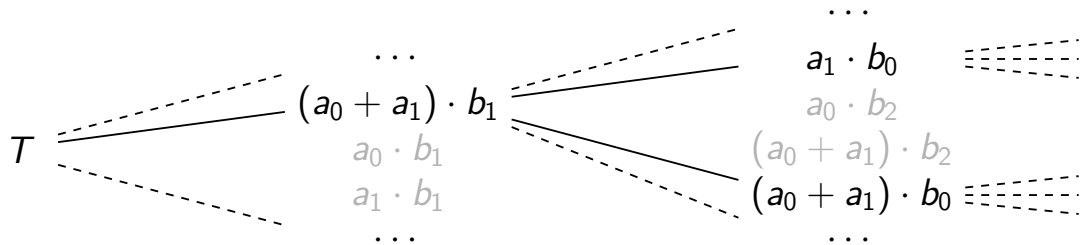
Orbits of a group action: Example

2×3 -term polynomial product in $\mathbb{F}_2[X]$

- ▶ So we add generator $(a_0 + a_1) \cdot b_1$, fixed by σ_0
- ▶ Now, consider adding to $T \oplus \text{Span}((a_0 + a_1) \cdot b_1)$ a generator from orbit $\overline{g_3}$:

$$\begin{array}{ccccc}
 a_0 \cdot b_2 & \xleftrightarrow{\sigma_0} & a_1 \cdot b_0 & \xleftrightarrow{\sigma_1} & a_1 \cdot (b_0 + b_1 + b_2) \\
 \sigma_1 \uparrow \downarrow & & & & \uparrow \downarrow \sigma_0 \\
 (a_0 + a_1) \cdot b_2 & \xleftrightarrow{\sigma_0} & (a_0 + a_1) \cdot b_0 & \xleftrightarrow{\sigma_1} & a_0 \cdot (b_0 + b_1 + b_2)
 \end{array}$$

- ▶ Since $T \oplus \text{Span}((a_0 + a_1) \cdot b_1)$ is fixed by σ_0 , we can still partition $\overline{g_3}$ according to its σ_0 -orbits



Outline of the talk

- ▶ Formulae for polynomial multiplication
- ▶ Enumerating formulae
- ▶ Further improvements and heuristics
- ▶ **Some results**

$n \times m$ -term polynomial multiplication

Ring	$n \times m$	$\#\mathcal{G}$	k	# of tests	# of solutions	# of formulae	Calculation time [s]
$F_2[X]$	2×2	9	3	1	1	1	0.00
	3×3	49	6	9	3	9	0.00
	4×4	225	9	$6.60 \cdot 10^3$	4	4	0.03
	5×5	961	13	$9.65 \cdot 10^9$	27	27	$2.28 \cdot 10^5$
	6×6	3969	14	$4.37 \cdot 10^9$	—	—	$6.03 \cdot 10^5$
		(Sym.) 63	17	$8.08 \cdot 10^6$	6	54	17.7
	7×7	(Sym.) 127	22	$3.38 \cdot 10^{12}$	2 618	19 550	$1.59 \cdot 10^7$
$F_3[X]$	2×2	16	3	1	1	4	0.00
	3×3	169	6	24	22	1 493	0.00
	4×4	1 600	9	$4.11 \cdot 10^5$	726	50 640	14.9
	5×5	14 641	11	$4.89 \cdot 10^7$	—	—	$4.02 \cdot 10^4$
		(Sym.) 121	12	$3.93 \cdot 10^4$	31	6 460	0.14
	6×6	(Sym.) 364	15	$2.37 \cdot 10^8$	4	1 024	$3.79 \cdot 10^3$
	7×7	(Sym.) 1 093	17	$2.69 \cdot 10^{10}$	—	—	$1.50 \cdot 10^6$

$n \times m$ -term polynomial multiplication

Ring	$n \times m$	$\#\mathcal{G}$	k	# of tests	# of solutions	# of formulae	Calculation time [s]
$F_2[X]$	2×2	9	3	1	1	1	0.00
	3×3	49	6	9	3	9	0.00
	4×4	225	9	$6.60 \cdot 10^3$	4	4	0.03
	5×5	961	13	$9.65 \cdot 10^9$	27	27	$2.28 \cdot 10^5$
	6×6	3969	14	$4.37 \cdot 10^9$	—	—	$6.03 \cdot 10^5$
		(Sym.) 63	17	$8.08 \cdot 10^6$	6	54	17.7
	7×7	(Sym.) 127	22	$3.38 \cdot 10^{12}$	2 618	19 550	$1.59 \cdot 10^7$
$F_3[X]$	2×2	16	3	1	1	4	0.00
	3×3	169	6	24	22	1 493	0.00
	4×4	1 600	9	$4.11 \cdot 10^5$	726	50 640	14.9
	5×5	14 641	11	$4.89 \cdot 10^7$	—	—	$4.02 \cdot 10^4$
		(Sym.) 121	12	$3.93 \cdot 10^4$	31	6 460	0.14
	6×6	(Sym.) 364	15	$2.37 \cdot 10^8$	4	1 024	$3.79 \cdot 10^3$
	7×7	(Sym.) 1 093	17	$2.69 \cdot 10^{10}$	—	—	$1.50 \cdot 10^6$

$n \times m$ -term polynomial multiplication

Ring	$n \times m$	$\#\mathcal{G}$	k	# of tests	# of solutions	# of formulae	Calculation time [s]
$F_2[X]$	2×2	9	3	1	1	1	0.00
	3×3	49	6	9	3	9	0.00
	4×4	225	9	$6.60 \cdot 10^3$	4	4	0.03
	5×5	961	13	$9.65 \cdot 10^9$	27	27	$2.28 \cdot 10^5$
	6×6	3969	14	$4.37 \cdot 10^9$	—	—	$6.03 \cdot 10^5$
		(Sym.) 63	17	$8.08 \cdot 10^6$	6	54	17.7
	7×7	(Sym.) 127	22	$3.38 \cdot 10^{12}$	2 618	19 550	$1.59 \cdot 10^7$
$F_3[X]$	2×2	16	3	1	1	4	0.00
	3×3	169	6	24	22	1 493	0.00
	4×4	1 600	9	$4.11 \cdot 10^5$	726	50 640	14.9
	5×5	14 641	11	$4.89 \cdot 10^7$	—	—	$4.02 \cdot 10^4$
		(Sym.) 121	12	$3.93 \cdot 10^4$	31	6 460	0.14
	6×6	(Sym.) 364	15	$2.37 \cdot 10^8$	4	1 024	$3.79 \cdot 10^3$
	7×7	(Sym.) 1 093	17	$2.69 \cdot 10^{10}$	—	—	$1.50 \cdot 10^6$

n × m-term polynomial multiplication

Ring	n × m	# \mathcal{G}	k	# of tests	# of solutions	# of formulae	Calculation time [s]
F₂[X]	2 × 2	9	3	1	1	1	0.00
	3 × 3	49	6	9	3	9	0.00
	4 × 4	225	9	$6.60 \cdot 10^3$	4	4	0.03
	5 × 5	961	13	$9.65 \cdot 10^9$	27	27	$2.28 \cdot 10^5$
	6 × 6	3969	14	$4.37 \cdot 10^9$	—	—	$6.03 \cdot 10^5$
		(Sym.) 63	17	$8.08 \cdot 10^6$	6	54	17.7
	7 × 7	(Sym.) 127	22	$3.38 \cdot 10^{12}$	2 618	19 550	$1.59 \cdot 10^7$
F₃[X]	2 × 2	16	3	1	1	4	0.00
	3 × 3	169	6	24	22	1 493	0.00
	4 × 4	1 600	9	$4.11 \cdot 10^5$	726	50 640	14.9
	5 × 5	14 641	11	$4.89 \cdot 10^7$	—	—	$4.02 \cdot 10^4$
		(Sym.) 121	12	$3.93 \cdot 10^4$	31	6 460	0.14
	6 × 6	(Sym.) 364	15	$2.37 \cdot 10^8$	4	1 024	$3.79 \cdot 10^3$
	7 × 7	(Sym.) 1 093	17	$2.69 \cdot 10^{10}$	—	—	$1.50 \cdot 10^6$

$n \times m$ -term polynomial multiplication

Ring	$n \times m$	$\#\mathcal{G}$	k	# of tests	# of solutions	# of formulae	Calculation time [s]
$F_2[X]$	2×2	9	3	1	1	1	0.00
	3×3	49	6	9	3	9	0.00
	4×4	225	9	$6.60 \cdot 10^3$	4	4	0.03
	5×5	961	13	$9.65 \cdot 10^9$	27	27	$2.28 \cdot 10^5$
	6×6	3969	14	$4.37 \cdot 10^9$	—	—	$6.03 \cdot 10^5$
		(Sym.) 63	17	$8.08 \cdot 10^6$	6	54	17.7
	7×7	(Sym.) 127	22	$3.38 \cdot 10^{12}$	2 618	19 550	$1.59 \cdot 10^7$
$F_3[X]$	2×2	16	3	1	1	4	0.00
	3×3	169	6	24	22	1 493	0.00
	4×4	1 600	9	$4.11 \cdot 10^5$	726	50 640	14.9
	5×5	14 641	11	$4.89 \cdot 10^7$	—	—	$4.02 \cdot 10^4$
		(Sym.) 121	12	$3.93 \cdot 10^4$	31	6 460	0.14
	6×6	(Sym.) 364	15	$2.37 \cdot 10^8$	4	1 024	$3.79 \cdot 10^3$
	7×7	(Sym.) 1 093	17	$2.69 \cdot 10^{10}$	—	—	$1.50 \cdot 10^6$

$n \times m$ -term polynomial multiplication

Ring	$n \times m$	$\#\mathcal{G}$	k	# of tests	# of solutions	# of formulae	Calculation time [s]
$F_2[X]$	2×2	9	3	1	1	1	0.00
	3×3	49	6	9	3	9	0.00
	4×4	225	9	$6.60 \cdot 10^3$	4	4	0.03
	5×5	961	13	$9.65 \cdot 10^9$	27	27	$2.28 \cdot 10^5$
	6×6	3969	14	$4.37 \cdot 10^9$	—	—	$6.03 \cdot 10^5$
		(Sym.) 63	17	$8.08 \cdot 10^6$	6	54	17.7
	7×7	(Sym.) 127	22	$3.38 \cdot 10^{12}$	2 618	19 550	$1.59 \cdot 10^7$
$F_3[X]$	2×2	16	3	1	1	4	0.00
	3×3	169	6	24	22	1 493	0.00
	4×4	1 600	9	$4.11 \cdot 10^5$	726	50 640	14.9
	5×5	14 641	11	$4.89 \cdot 10^7$	—	—	$4.02 \cdot 10^4$
		(Sym.) 121	12	$3.93 \cdot 10^4$	31	6 460	0.14
	6×6	(Sym.) 364	15	$2.37 \cdot 10^8$	4	1 024	$3.79 \cdot 10^3$
	7×7	(Sym.) 1 093	17	$2.69 \cdot 10^{10}$	—	—	$1.50 \cdot 10^6$

Multiplication in small finite-field extensions

Finite field	$\#\mathcal{G}$	k	# of tests	# of solutions	# of formulae	Calculation time [s]
\mathbf{F}_{2^2}	9	3	3	3	3	0.00
\mathbf{F}_{2^3}	49	6	$7.03 \cdot 10^3$	105	147	0.01
\mathbf{F}_{2^4}	225	9	$2.57 \cdot 10^9$	2025	2025	$1.13 \cdot 10^4$
\mathbf{F}_{2^5}	961	9	$3.10 \cdot 10^{10}$	—	—	$8.11 \cdot 10^5$
	(Sym.) 31	13	$3.49 \cdot 10^6$	2015	2015	6.24
\mathbf{F}_{2^6}	(Sym.) 63	15	$2.21 \cdot 10^{10}$	21	21	$6.63 \cdot 10^4$
\mathbf{F}_{2^7}	(Sym.) 127	15	$1.34 \cdot 10^{12}$	—	—	$6.17 \cdot 10^6$
\mathbf{F}_{3^2}	16	3	3	4	16	0.00
\mathbf{F}_{3^3}	169	6	$2.42 \cdot 10^5$	11 843	105 963	1.08
\mathbf{F}_{3^4}	1 600	8	$2.27 \cdot 10^{11}$	—	—	$1.08 \cdot 10^7$
	(Sym.) 40	9	$1.10 \cdot 10^5$	234	615 240	0.45
\mathbf{F}_{3^5}	(Sym.) 121	11	$2.66 \cdot 10^9$	121	121	$1.45 \cdot 10^4$
\mathbf{F}_{3^6}	(Sym.) 364	12	$3.01 \cdot 10^{12}$	—	—	$4.50 \cdot 10^7$

Multiplication in small finite-field extensions

Finite field	$\#\mathcal{G}$	k	# of tests	# of solutions	# of formulae	Calculation time [s]
\mathbf{F}_{2^2}	9	3	3	3	3	0.00
\mathbf{F}_{2^3}	49	6	$7.03 \cdot 10^3$	105	147	0.01
\mathbf{F}_{2^4}	225	9	$2.57 \cdot 10^9$	2025	2025	$1.13 \cdot 10^4$
\mathbf{F}_{2^5}	961	9	$3.10 \cdot 10^{10}$	—	—	$8.11 \cdot 10^5$
	(Sym.) 31	13	$3.49 \cdot 10^6$	2015	2015	6.24
\mathbf{F}_{2^6}	(Sym.) 63	15	$2.21 \cdot 10^{10}$	21	21	$6.63 \cdot 10^4$
\mathbf{F}_{2^7}	(Sym.) 127	15	$1.34 \cdot 10^{12}$	—	—	$6.17 \cdot 10^6$
\mathbf{F}_{3^2}	16	3	3	4	16	0.00
\mathbf{F}_{3^3}	169	6	$2.42 \cdot 10^5$	11 843	105 963	1.08
\mathbf{F}_{3^4}	1 600	8	$2.27 \cdot 10^{11}$	—	—	$1.08 \cdot 10^7$
	(Sym.) 40	9	$1.10 \cdot 10^5$	234	615 240	0.45
\mathbf{F}_{3^5}	(Sym.) 121	11	$2.66 \cdot 10^9$	121	121	$1.45 \cdot 10^4$
\mathbf{F}_{3^6}	(Sym.) 364	12	$3.01 \cdot 10^{12}$	—	—	$4.50 \cdot 10^7$

Conclusion

- ▶ General algorithm
- ▶ Method that proves lower bounds on the number of subproducts

Conclusion

- ▶ General algorithm
- ▶ Method that **proves lower bounds** on the number of subproducts
- ▶ Gives **all formulae**
 - Provides **new formulae** that cannot be found with previous methods
 - We can cherry-pick the one with **minimum number of additions and scalar multiplications**

Conclusion

- ▶ General algorithm
- ▶ Method that **proves lower bounds** on the number of subproducts
- ▶ Gives **all formulae**
 - Provides **new formulae** that cannot be found with previous methods
 - We can cherry-pick the one with **minimum number of additions and scalar multiplications**
- ▶ Work in progress and perspectives
 - **Lifting formulae** for higher-characteristic or characteristic-0 fields
 - Find formulae for **your** bilinear application!

Thank you for your attention

Questions?

Representing elements

- ▶ Finite field K :
 - elements represented using $\log_2 \#K$ bits
 - small fields \Rightarrow fits within a word, within a `uint8_t` (1 byte)

Representing elements

- ▶ Finite field K :
 - elements represented using $\log_2 \#K$ bits
 - small fields \Rightarrow fits within a word, within a `uint8_t` (1 byte)
- ▶ V : N -dimensional K -vector space
 - elements represented using $\log_2 \#K$ vectors of N bits (bitsliced representation)

Representing elements

- ▶ Finite field K :
 - elements represented using $\log_2 \#K$ bits
 - small fields \Rightarrow fits within a word, within a `uint8_t` (1 byte)
- ▶ V : N -dimensional K -vector space
 - elements represented using $\log_2 \#K$ vectors of N bits (bitsliced representation)
 - for instance, for the 2×3 -term polynomial multiplication over $\mathbb{F}_3[X]$:

$$v = a_0b_1 - a_1b_0 + a_1b_2$$

Representing elements

▶ Finite field K :

- elements represented using $\log_2 \#K$ bits
- small fields \Rightarrow fits within a word, within a `uint8_t` (1 byte)

▶ V : N -dimensional K -vector space

- elements represented using $\log_2 \#K$ vectors of N bits (bitsliced representation)
- for instance, for the 2×3 -term polynomial multiplication over $\mathbb{F}_3[X]$:

$$\begin{aligned} v &= a_0b_1 - a_1b_0 + a_1b_2 \\ &= \begin{matrix} & a_0b_0 & a_0b_1 & a_0b_2 & a_1b_0 & a_1b_1 & a_1b_2 \end{matrix} \\ &= (0, 1, 0, -1, 0, 1) \end{aligned}$$

Representing elements

▶ Finite field K :

- elements represented using $\log_2 \#K$ bits
- small fields \Rightarrow fits within a word, within a `uint8_t` (1 byte)

▶ V : N -dimensional K -vector space

- elements represented using $\log_2 \#K$ vectors of N bits (bitsliced representation)
- for instance, for the 2×3 -term polynomial multiplication over $\mathbb{F}_3[X]$:

$$\begin{aligned} v &= a_0b_1 - a_1b_0 + a_1b_2 \\ &= \begin{matrix} & a_0b_0 & a_0b_1 & a_0b_2 & a_1b_0 & a_1b_1 & a_1b_2 \end{matrix} \\ &= (0, 1, 0, 2, 0, 1) \end{aligned}$$

Representing elements

▶ Finite field K :

- elements represented using $\log_2 \#K$ bits
- small fields \Rightarrow fits within a word, within a `uint8_t` (1 byte)

▶ V : N -dimensional K -vector space

- elements represented using $\log_2 \#K$ vectors of N bits (bitsliced representation)
- for instance, for the 2×3 -term polynomial multiplication over $\mathbb{F}_3[X]$:

$$\begin{aligned}v &= a_0b_1 - a_1b_0 + a_1b_2 \\ & \quad \begin{matrix} a_0b_0 & a_0b_1 & a_0b_2 & a_1b_0 & a_1b_1 & a_1b_2 \end{matrix} \\ &= (0, 1, 0, 2, 0, 1) \\ &= (0, 1, 0, 0, 0, 1) + \\ & \quad 2 \cdot (0, 0, 0, 1, 0, 0)\end{aligned}$$

Representing elements

▶ Finite field K :

- elements represented using $\log_2 \#K$ bits
- small fields \Rightarrow fits within a word, within a `uint8_t` (1 byte)

▶ V : N -dimensional K -vector space

- elements represented using $\log_2 \#K$ vectors of N bits (bitsliced representation)
- for instance, for the 2×3 -term polynomial multiplication over $\mathbb{F}_3[X]$:

$$\begin{aligned} v &= a_0b_1 - a_1b_0 + a_1b_2 \\ &= \begin{matrix} & a_0b_0 & a_0b_1 & a_0b_2 & a_1b_0 & a_1b_1 & a_1b_2 \end{matrix} \\ &= (0, 1, 0, 2, 0, 1) \\ &= (0, 1, 0, 0, 0, 1) + \\ &\quad 2 \cdot (0, 0, 0, 1, 0, 0) \end{aligned}$$

$$\text{uint}N_t \ v[2] = \{ 0x22u, 0x08u \};$$

Representing elements

▶ Finite field K :

- elements represented using $\log_2 \#K$ bits
- small fields \Rightarrow fits within a word, within a `uint8_t` (1 byte)

▶ V : N -dimensional K -vector space

- elements represented using $\log_2 \#K$ vectors of N bits (bitsliced representation)
- for instance, for the 2×3 -term polynomial multiplication over $\mathbb{F}_3[X]$:

$$\begin{aligned} v &= a_0b_1 - a_1b_0 + a_1b_2 \\ &= \begin{matrix} & a_0b_0 & a_0b_1 & a_0b_2 & a_1b_0 & a_1b_1 & a_1b_2 \end{matrix} \\ &= (0, 1, 0, 2, 0, 1) \\ &= (0, 1, 0, 0, 0, 1) + \\ &\quad 2 \cdot (0, 0, 0, 1, 0, 0) \end{aligned}$$

$$\text{uint}N_t \ v[2] = \{ 0x22u, 0x08u \};$$

- coefficient-wise add, sub, mul, etc. using bitwise operations on N -bit words

Representing elements

▶ Finite field K :

- elements represented using $\log_2 \#K$ bits
- small fields \Rightarrow fits within a word, within a `uint8_t` (1 byte)

▶ V : N -dimensional K -vector space

- elements represented using $\log_2 \#K$ vectors of N bits (bitsliced representation)
- for instance, for the 2×3 -term polynomial multiplication over $\mathbb{F}_3[X]$:

$$\begin{aligned} v &= a_0b_1 - a_1b_0 + a_1b_2 \\ &= \begin{matrix} & a_0b_0 & a_0b_1 & a_0b_2 & a_1b_0 & a_1b_1 & a_1b_2 \end{matrix} \\ &= (0, 1, 0, 2, 0, 1) \\ &= (0, 1, 0, 0, 0, 1) + \\ &\quad 2 \cdot (0, 0, 0, 1, 0, 0) \end{aligned}$$

$$\text{uint}N_t \ v[2] = \{ 0x22u, 0x08u \};$$

- coefficient-wise add, sub, mul, etc. using bitwise operations on N -bit words
e.g., addition $w \leftarrow u + v$ with $K = \mathbb{F}_3$:

$$w[0] = (u[0] | v[0]) \wedge ((u[0] | u[1]) \& (v[0] | v[1]));$$

$$w[1] = (u[1] | v[1]) \wedge ((u[0] | u[1]) \& (v[0] | v[1]));$$

Representing elements

► Finite field K :

- elements represented using $\log_2 \#K$ bits
- small fields \Rightarrow fits within a word, within a `uint8_t` (1 byte)

► V : N -dimensional K -vector space

- elements represented using $\log_2 \#K$ vectors of N bits (bitsliced representation)
- for instance, for the 2×3 -term polynomial multiplication over $\mathbb{F}_3[X]$:

$$\begin{aligned} v &= a_0b_1 - a_1b_0 + a_1b_2 \\ &= \begin{matrix} & a_0b_0 & a_0b_1 & a_0b_2 & a_1b_0 & a_1b_1 & a_1b_2 \end{matrix} \\ &= (0, 1, 0, 2, 0, 1) \\ &= (0, 1, 0, 0, 0, 1) + \\ &\quad 2 \cdot (0, 0, 0, 1, 0, 0) \end{aligned}$$

$$\text{uint}N_t \ v[2] = \{ 0x22u, 0x08u \};$$

- coefficient-wise add, sub, mul, etc. using bitwise operations on N -bit words
e.g., addition $w \leftarrow u + v$ with $K = \mathbb{F}_3$:

$$w[0] = (u[0] | v[0]) \wedge ((u[0] | u[1]) \& (v[0] | v[1]));$$

$$w[1] = (u[1] | v[1]) \wedge ((u[0] | u[1]) \& (v[0] | v[1]));$$

- machine-specific bound on N : ≤ 64 , ≤ 128 (SSE2), or ≤ 256 (AVX)

Representing subspaces

- ▶ Subspaces $W \subset V$ represented using matrices in row echelon form

$$W = \text{RowSpace} \left(\begin{array}{cccccccccc} & \overbrace{\hspace{10em}}^{\dim V = N \text{ columns}} & & & & & & & & & \\ 0 & w_{1,j_1} & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \\ 0 & 0 & 0 & w_{2,j_2} & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \\ \vdots & \vdots & \vdots & 0 & 0 & \ddots & & & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & w_{r,j_r} & \cdots & \end{array} \right) \left. \vphantom{\begin{array}{c} \\ \\ \\ \\ \end{array}} \right\} \dim W = r \text{ rows}$$

where each pivot $w_{i,j_i} \neq 0$, and with $j_1 < j_2 < \cdots < j_r$

Representing subspaces

- ▶ Subspaces $W \subset V$ represented using matrices in row echelon form

$$W = \text{RowSpace} \left(\begin{array}{cccccccccc} & & & & \overbrace{\hspace{10em}}^{\dim V = N \text{ columns}} & & & & & & \\ 0 & w_{1,j_1} & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \\ 0 & 0 & 0 & w_{2,j_2} & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \\ \vdots & \vdots & \vdots & 0 & 0 & \ddots & & & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & w_{r,j_r} & \cdots & \end{array} \right) \left. \vphantom{\begin{array}{c} \\ \\ \\ \\ \end{array}} \right\} \dim W = r \text{ rows}$$

where each pivot $w_{i,j_i} \neq 0$, and with $j_1 < j_2 < \cdots < j_r$

- ▶ Actual representation: since $\dim W \leq \dim V$, force the matrix W to N rows
 - rows corresponding to pivot w_{i,j_i} placed at row j_i
 - other rows set to zero

Representing subspaces

- ▶ Subspaces $W \subset V$ represented using matrices in row echelon form

$$W = \text{RowSpace} \left(\begin{array}{cccccccccc}
 & & & & \overbrace{\hspace{10em}}^{\dim V = N \text{ columns}} & & & & & \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & w_{1,j_1} & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & w_{2,j_2} & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & & & & \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & w_{r,j_r} & \cdots \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
 \end{array} \right) \left. \vphantom{\begin{array}{c} \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \end{array}} \right\} \dim V = N \text{ rows}$$

where each pivot $w_{i,j_i} \neq 0$, and with $j_1 < j_2 < \cdots < j_r$

- ▶ Actual representation: since $\dim W \leq \dim V$, force the matrix W to N rows
 - rows corresponding to pivot w_{i,j_i} placed at row j_i
 - other rows set to zero

Representing subspaces

- Subspaces $W \subset V$ represented using matrices in row echelon form

$$W = \text{RowSpace} \left(\begin{array}{cccccccccc}
 & & & & \overbrace{\hspace{10em}}^{\dim V = N \text{ columns}} & & & & & \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & w_{j_1, j_1} & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & w_{j_2, j_2} & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & & & & \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & w_{j_r, j_r} & \cdots \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
 \end{array} \right) \left. \vphantom{\begin{array}{c} \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \end{array}} \right\} \dim V = N \text{ rows}$$

where each pivot $w_{i, j_i} \neq 0$, and with $j_1 < j_2 < \cdots < j_r$

- Actual representation: since $\dim W \leq \dim V$, force the matrix W to N rows
- rows corresponding to pivot w_{i, j_i} placed at row j_i
 - other rows set to zero
 - all pivots now lie on the diagonal

Representing subspaces

- ▶ Subspaces $W \subset V$ represented using matrices in row echelon form

$$\begin{array}{c}
 \text{dim } V = N \text{ columns} \\
 \left(\begin{array}{cccccccccc}
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & w_{j_1, j_1} & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & w_{j_2, j_2} & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & & & & \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & w_{j_r, j_r} & \cdots \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
 \end{array} \right) \left. \begin{array}{l} \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \end{array} \right\} \text{dim } V = N \text{ rows}
 \end{array}$$

$W = \text{RowSpace}$

where each pivot $w_{i, j_i} \neq 0$, and with $j_1 < j_2 < \cdots < j_r$

- ▶ Actual representation: since $\text{dim } W \leq \text{dim } V$, force the matrix W to N rows
 - rows corresponding to pivot w_{i, j_i} placed at row j_i
 - other rows set to zero
 - all pivots now lie on the diagonal
 - ☹ waste of memory
 - 😊 easier to insert extra rows when expanding W

Reduction by a subspace

- ▶ Given a vector $v \in V$ and a subspace W :
 - sequentially cancel the contribution of pivot rows of W to v
 - stop when no pivot corresponds to the leading coefficient of v

Reduction by a subspace

- Given a vector $v \in V$ and a subspace W :
- sequentially cancel the contribution of pivot rows of W to v
 - stop when no pivot corresponds to the leading coefficient of v
- 1: **for** $i \leftarrow 1$ **to** N **do**
 - 2: **if** $v = 0$ **then return** 0
 - 3: **if** $v_i \neq 0$ **then**
 - 4: **if** $w_{i,i} \neq 0$ **then** $v \leftarrow v - (v_i/w_{i,i})w_i$
 - 5: **else return** v
 - 6: **end for**

Reduction by a subspace

- ▶ Given a vector $v \in V$ and a subspace W :
 - sequentially cancel the contribution of pivot rows of W to v
 - stop when no pivot corresponds to the leading coefficient of v
- 1: **for** $i \leftarrow 1$ **to** N **do**
 - 2: **if** $v = 0$ **then return** 0
 - 3: **if** $v_i \neq 0$ **then**
 - 4: **if** $w_{i,i} \neq 0$ **then** $v \leftarrow v - (v_i/w_{i,i})w_i$
 - 5: **else return** v
 - 6: **end for**
- ▶ Test if $v \in W$:
 - $v \in W$ if and only if $\text{reduce}(v, W) = 0$

Reduction by a subspace

- ▶ Given a vector $v \in V$ and a subspace W :
 - sequentially cancel the contribution of pivot rows of W to v
 - stop when no pivot corresponds to the leading coefficient of v
- 1: **for** $i \leftarrow 1$ **to** N **do**
 - 2: **if** $v = 0$ **then return** 0
 - 3: **if** $v_i \neq 0$ **then**
 - 4: **if** $w_{i,i} \neq 0$ **then** $v \leftarrow v - (v_i/w_{i,i})w_i$
 - 5: **else return** v
 - 6: **end for**
- ▶ Test if $v \in W$:
 - $v \in W$ if and only if $\text{reduce}(v, W) = 0$
 - ▶ Construct $W \oplus \text{Span}(g)$, for $g \in \mathcal{G} \setminus W$:
 - compute $\tilde{g} \leftarrow \text{reduce}(g, W)$
 - let i be the iteration at which the algorithm returns \tilde{g} ($\neq 0$)
 - then $w_i \leftarrow \tilde{g}$

Reduction by a subspace

- ▶ Given a vector $v \in V$ and a subspace W :
 - sequentially cancel the contribution of pivot rows of W to v
 - stop when no pivot corresponds to the leading coefficient of v
- 1: **for** $i \leftarrow 1$ **to** N **do**
- 2: **if** $v = 0$ **then return** 0
- 3: **if** $v_i \neq 0$ **then**
- 4: **if** $w_{i,i} \neq 0$ **then** $v \leftarrow v - (v_i/w_{i,i})w_i$
- 5: **else return** v
- 6: **end for**
- ▶ Test if $v \in W$:
 - $v \in W$ if and only if $\text{reduce}(v, W) = 0$
- ▶ Construct $W \oplus \text{Span}(g)$, for $g \in \mathcal{G} \setminus W$:
 - compute $\tilde{g} \leftarrow \text{reduce}(g, W)$
 - let i be the iteration at which the algorithm returns \tilde{g} ($\neq 0$)
 - then $w_i \leftarrow \tilde{g}$
- ▶ Improvement: since W is constructed incrementally
 - keep the vectors of \mathcal{G} reduced by W at all times
 - when expanding W by $\text{Span}(g)$, reduce \mathcal{G} by g only

Computing $W \cap \mathcal{G}$

- ▶ Find all $g \in \mathcal{G}$ such that $g \in W$

Computing $W \cap \mathcal{G}$

- ▶ Find all $g \in \mathcal{G}$ such that $g \in W$
- ▶ For each $g \in \mathcal{G}$, test if $\text{reduce}(g, W) = 0$?
 - slow (only one vector at a time, lots of conditional branches, etc.)
 - we don't need to reduce g by W

Computing $W \cap \mathcal{G}$

- ▶ Find all $g \in \mathcal{G}$ such that $g \in W$
- ▶ For each $g \in \mathcal{G}$, test if $\text{reduce}(g, W) = 0$?
 - slow (only one vector at a time, lots of conditional branches, etc.)
 - we don't need to reduce g by W
- ▶ Idea (courtesy of E. Thomé): put matrix W in reduced row echelon form

$$W = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & w_{j_1, j_1} & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & w_{j_2, j_2} & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & w_{j_r, j_r} & \cdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Computing $W \cap \mathcal{G}$

- ▶ Find all $g \in \mathcal{G}$ such that $g \in W$
- ▶ For each $g \in \mathcal{G}$, test if $\text{reduce}(g, W) = 0$?
 - slow (only one vector at a time, lots of conditional branches, etc.)
 - we don't need to reduce g by W
- ▶ Idea (courtesy of E. Thomé): put matrix W in reduced row echelon form

$$W' = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Computing $W \cap \mathcal{G}$

- ▶ Find all $g \in \mathcal{G}$ such that $g \in W$
- ▶ For each $g \in \mathcal{G}$, test if $\text{reduce}(g, W) = 0$?
 - slow (only one vector at a time, lots of conditional branches, etc.)
 - we don't need to reduce g by W
- ▶ Idea (courtesy of E. Thomé): put matrix W in reduced row echelon form

$$W' = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \dots & \vdots & \dots & \dots & \dots & \vdots & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \vdots & 0 \\ 0 & 0 & 0 & 1 & \dots & \dots & \dots & \vdots & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \vdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & & \vdots & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Computing $W \cap \mathcal{G}$

- ▶ Find all $g \in \mathcal{G}$ such that $g \in W$
- ▶ For each $g \in \mathcal{G}$, test if $\text{reduce}(g, W) = 0$?
 - slow (only one vector at a time, lots of conditional branches, etc.)
 - we don't need to reduce g by W
- ▶ Idea (courtesy of E. Thomé): put matrix W in reduced row echelon form

$$W' = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \dots & \vdots & \dots & \dots & \dots & \vdots & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \vdots & 0 \\ 0 & 0 & 0 & 1 & \dots & \dots & \dots & \vdots & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \vdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & & \vdots & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

- all pivot columns j_i to 0, except for $w_{j_i, j_i} = 1$

Computing $W \cap \mathcal{G}$

- ▶ Find all $g \in \mathcal{G}$ such that $g \in W$
- ▶ For each $g \in \mathcal{G}$, test if $\text{reduce}(g, W) = 0$?
 - slow (only one vector at a time, lots of conditional branches, etc.)
 - we don't need to reduce g by W
- ▶ Idea (courtesy of E. Thomé): put matrix W in reduced row echelon form

$$W' = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \dots & \vdots & \dots & \dots & \dots & \vdots & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \vdots & 0 \\ 0 & 0 & 0 & 1 & \dots & \dots & \dots & \vdots & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \vdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & & \vdots & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

- all pivot columns j_i to 0, except for $w_{j_i, j_i} = 1$
- $g \in W$ if and only if $g^T \cdot W' = g$

Computing $W \cap \mathcal{G}$

- ▶ Find all $g \in \mathcal{G}$ such that $g \in W$
- ▶ For each $g \in \mathcal{G}$, test if $\text{reduce}(g, W) = 0$?
 - slow (only one vector at a time, lots of conditional branches, etc.)
 - we don't need to reduce g by W
- ▶ Idea (courtesy of E. Thomé): put matrix W in reduced row echelon form

$$W' = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \dots & \vdots & \dots & \dots & \dots & \vdots & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \vdots & 0 \\ 0 & 0 & 0 & 1 & \dots & \dots & \dots & \vdots & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \vdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & & \vdots & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

- all pivot columns j_i to 0, except for $w_{j_i, j_i} = 1$
- $g \in W$ if and only if $g^T \cdot W' = g^T$
- batch 64, 128, or 256 products $g^T \cdot (W' - \text{Id})$