Implementación de bases de Gröbner no conmutativas en C++ con un poquito de paralelismo

Iván Renison

Facultad de Matemática, Astronomía, Física y Computación
Universidad Nacional de Córdoba

2025-03-06

Polinomios

Polinomios

Algunos polinomios:

- $5 + 3y - 2xy + x^3y^5$
- $y^2 + x - x^2y^2$

Fijemos:

- Un alfabeto X
- Un cuerpo K

El conjunto de polinomios se denota $K[X]$.
El producto es conmutativo.

Por ejemplo en $\mathbb{Q}[x, y]$:

- $xy = yx$
- $y^2 + x - x^2y^2 = yy + x - yxxy$

Polinomios no conmutativos

Polinomios no conmutativos

Por ejemplo en $\mathbb{Q}\langle x, y, z\rangle$:

- $f_0 = x$
- $f_1 = xy + yz$
- $f_2 = 3xyy - 2xzxy + \frac{4}{3}yzzx$
- $f_3 = yx + zx$
- $f_4 = xyx - yyx$
- $f_5 = xyz + zyx$
- $f_6 = -xyx + zyx - 2yxyx + 2zzyx$

Polinomios no conmutativos

Por ejemplo en $\mathbb{Q}\langle x, y, z\rangle$:

- $f_0 = x$
- $f_1 = xy + yz$
- $f_2 = 3xyy - 2xzxy + \frac{4}{3}yzzx$
- $f_3 = yx + zx$
- $f_4 = xyx - yyx$
- $f_5 = xyz + zyx$
- $f_6 = -xyx + zyx - 2yxyx + 2zzyx$

Se denotan:

- $K\langle X\rangle$ el conjunto de polinomios no conmutativos
- $\langle X\rangle$ el conjunto de monomios

Ordenes monomiales

$\leq_{deglex}$ ordena a $\langle X \rangle$ primero por largo y después por orden lexicográfico.

Ordenes monomiales

$\leq_{deglex}$ ordena a $\langle X \rangle$ primero por largo y después por orden lexicográfico.

En los ejemplos de antes:

- $f_0 = x$
- $f_1 = xy + yz$
- $f_2 = 3xyy - 2xzxy + \frac{4}{3}yzzx$
- $f_3 = yx + zx$
- $f_4 = xyx - yyx$
- $f_5 = xyz + zyx$
- $f_6 = -xyx + zyx - 2yxyx + 2zzyx$

Problema de pertenencia al ideal

Dados $G \subseteq K\langle X\rangle$ y $f \in K\langle X\rangle$, determinar si vale que

$$\exists g_1, ..., g_n \in G, f_1, ..., f_n, f'_1, ..., f'_n \in K\langle X\rangle : f = \sum_{i=1}^{n} f_i g_i f'_i.$$

Problema de pertenencia al ideal

Dados $G \subseteq K\langle X\rangle$ y $f \in K\langle X\rangle$, determinar si vale que

$$\exists g_1, ..., g_n \in G, f_1, ..., f_n, f'_1, ..., f'_n \in K\langle X\rangle : f = \sum_{i=1}^{n} f_i g_i f'_i.$$

Por ejemplo, si:

- $g_0 = f_1 = xy + yz$
- $g_1 = f_3 = yx + zx$
- $G = \{g_0, g_1\}$

Entonces:

- Para $f_4 = xyx - yyx$ si vale

Problema de pertenencia al ideal

Dados $G \subseteq K\langle X\rangle$ y $f \in K\langle X\rangle$, determinar si vale que

$$\exists g_1, ..., g_n \in G, f_1, ..., f_n, f_1', ..., f_n' \in K\langle X\rangle : f = \sum_{i=1}^{n} f_i g_i f_i'.$$

Por ejemplo, si:

- $g_0 = f_1 = xy + yz$
- $g_1 = f_3 = yx + zx$
- $G = \{g_0, g_1\}$

Entonces:

- Para $f_4 = xyx - yyx$ si vale, porque es igual a $g_0x - yg_1$.

Problema de pertenencia al ideal

Dados $G \subseteq K\langle X\rangle$ y $f \in K\langle X\rangle$, determinar si vale que

$$\exists g_1, ..., g_n \in G, f_1, ..., f_n, f'_1, ..., f'_n \in K\langle X\rangle : f = \sum_{i=1}^{n} f_i g_i f'_i.$$

Por ejemplo, si:

- $g_0 = f_1 = xy + yz$
- $g_1 = f_3 = yx + zx$
- $G = \{g_0, g_1\}$

Entonces:

- Para $f_4 = xyx - yyx$ si vale, porque es igual a $g_0x - yg_1$.
- Para $f_5 = xyz + zyx$ no vale.

Ideales

Ideales

Sea $G \subseteq K\langle X\rangle$, se define

$$(G) = \{\sum_{i=1}^{n} c_i g_i c_i' : g_1, ..., g_n \in G, c_1, ..., c_n, c_1', ..., c_n' \in K\langle X\rangle\}$$

A (G) se lo llama el ideal generado por G.

Teorema

Teorema

Sean $G \subseteq K\langle X\rangle$, $f \in K\langle X\rangle$, $g \in G$, $a, b \in \langle X\rangle$ y $c \in \langle X\rangle$. Entonces

$$f \in (G) \Leftrightarrow f + cagb \in (G)$$

Reducciones

Ejemplo

- $g_0 = f_1 = xy + yz$
- $g_1 = f_3 = yx + zx$
- $G = \{g_0, g_1\}$
- $f_4 = xyx - yyx$
- $f_5 = xyz + zyx$
- $f_6 = -xyx + zyx - 2yxyx + 2zzyx$

Algoritmo de Buchberger

Algoritmo de Buchberger

La idea de Buchberger es ir agregando los S-polinomios reducidos

Bases de Gröbner

$$G \text{ es una base de Gröbner} \Leftrightarrow \forall f \in K\langle X\rangle : f \in (G) \Leftrightarrow f \text{ se puede reducir}$$

Bases de Gröbner

G es una base de Gröbner $\Leftrightarrow \forall f \in K\langle X\rangle : f \in (G) \Leftrightarrow f$ se puede reducir

Si agregamos S-polinomios hasta que no queda ninguno entonces obtenemos una base de Gröbner

Algoritmo F4

Algoritmo F4

F4 hace lo mismo que Buchberger pero de a muchos polinomios al mismo tiempo usando álgebra lineal

Implementaciones previas

Implementaciones previas

De Buchberger:

- Paquete GBNP de GAP
- Implementación dentro de Singular
- Implementación dentro de NCAlgebra
- Paquete `OperatorGB` de Mathematica

Implementaciones previas

De Buchberger:

- Paquete GBNP de GAP
- Implementación dentro de Singular
- Implementación dentro de NCAlgebra
- Paquete `OperatorGB` de Mathematica

De F4:

- Implementación dentro de Magma
- Paquete `operator_gb` de Python y SageMath

Mi librería

Librería `ncgb` de C++

Mi librería

Librería `ncgb` de C++

- Implementa estructuras para monomios y polinomios con sus operaciones básicas

Mi librería

Librería `ncgb` de C++

- Implementa estructuras para monomios y polinomios con sus operaciones básicas
- Implementa Buchberger y F4

Mi librería

Librería `ncgb` de C++

- Implementa estructuras para monomios y polinomios con sus operaciones básicas
- Implementa Buchberger y F4
- Funciona para un cuerpo arbitrario

Mi librería

Librería `ncgb` de C++

- Implementa estructuras para monomios y polinomios con sus operaciones básicas
- Implementa Buchberger y F4
- Funciona para un cuerpo arbitrario
- Para los racionales usa la libraría FLINT para las matrices

Mi librería

Librería `ncgb` de C++

- Implementa estructuras para monomios y polinomios con sus operaciones básicas
- Implementa Buchberger y F4
- Funciona para un cuerpo arbitrario
- Para los racionales usa la libraría FLINT para las matrices
- Incluye representación de cofactores ("*reconstrucción de la respuesta*")

Mi librería

Librería `ncgb` de C++

- Implementa estructuras para monomios y polinomios con sus operaciones básicas
- Implementa Buchberger y F4
- Funciona para un cuerpo arbitrario
- Para los racionales usa la libraría FLINT para las matrices
- Incluye representación de cofactores ("*reconstrucción de la respuesta*")
- Tiene un poquito de paralelismo

Mi librería

Librería `ncgb` de C++

- Implementa estructuras para monomios y polinomios con sus operaciones básicas
- Implementa Buchberger y F4
- Funciona para un cuerpo arbitrario
- Para los racionales usa la libraría FLINT para las matrices
- Incluye representación de cofactores ("*reconstrucción de la respuesta*")
- Tiene un poquito de paralelismo
- Falta implementar la optimización de Faugère-Lachartre en F4

Ejemplo de mi librería

Teníamos:

- $g_0 = f_1 = ab + bc$
- $g_1 = f_3 = ba + ca$
- $G = \{g_0, g_1\}$

Dijimos que:

- Para $f_4 = aba - bba$ si vale.
- Para $f_5 = abc + cba$ no vale.

Comparación

Para los racionales:

```
Average times:
Buch: 0.0041429424s
F4: 0.0105502963s
GB: 0.8995876765s
Max times:
Buch: 0.1320753098s
F4: 0.0925185680s
GB: 4.6477575302s
```

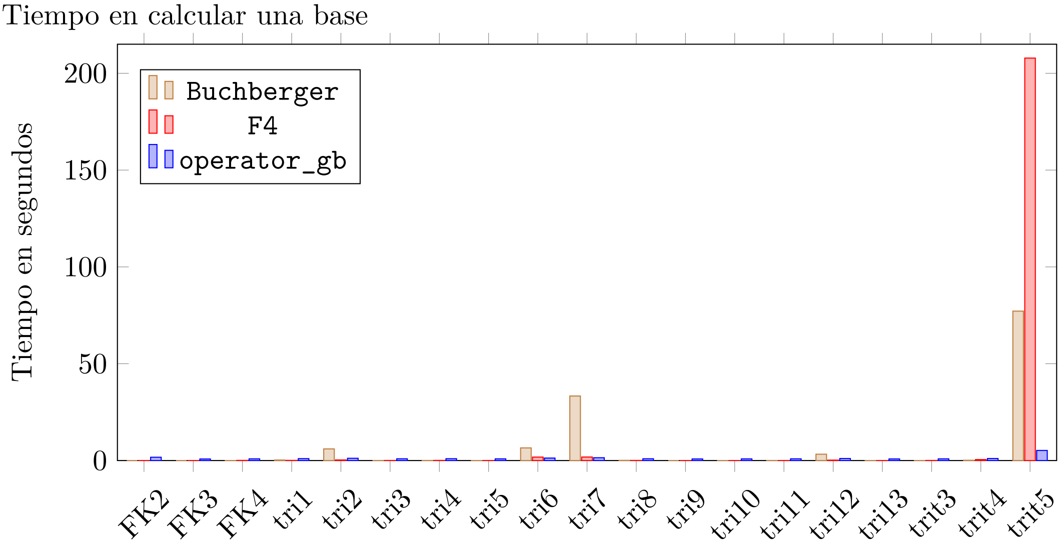
Para la aritmética modular:

```
Average times:
Buch_Zp: 0.0028817248s
F4_Zp: 0.0072190428s
Max times:
Buch_Zp: 0.0779249668s
F4_Zp: 0.0799708366s
```

Comparación

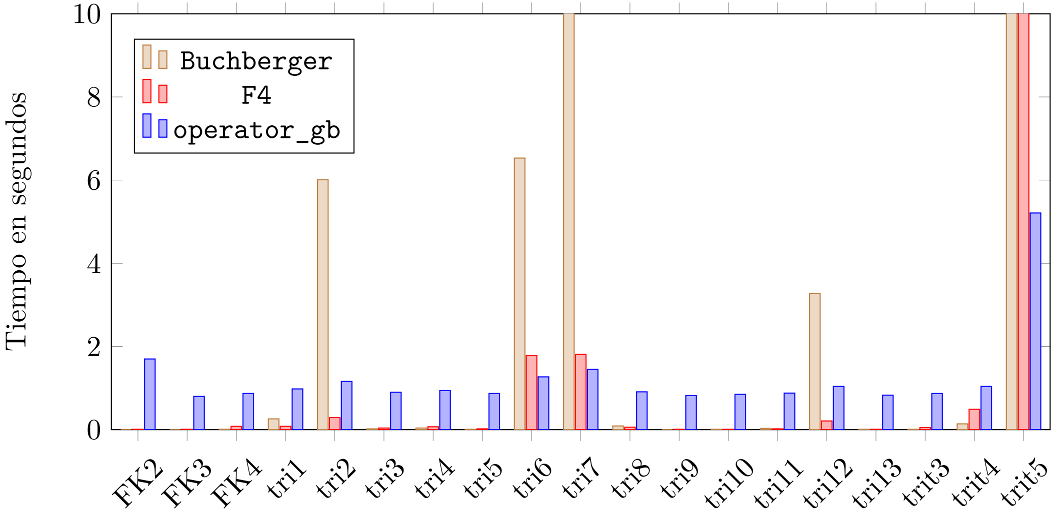
$$
\begin{aligned}
\mathrm{FK2} =& \{a^2\}\\
\mathrm{FK3} =& \{a^2,\ b^2,\ c^2,\ ac+ba+cb,\ ab+bc+ca\}\\
\mathrm{FK4} =& \{a^2,\ b^2,\ c^2,\ d^2,\ e^2,\ f^2,\ ac+ba+cb,\ ae+da+ed,\ bf+db+fd,\ cf+ec+fe,\ ab+bc+ca,\ ad+de+ea,\\
& bd+df+fb,\ ce+ef+fc,\ cd+dc,\ be+eb,\ af+fa\}\\
\mathrm{tri1} =& \{a^2-1,\ b^3-1,\ (ababab^2ab^2)^2-1\}\\
\mathrm{tri2} =& \{a^2-1,\ b^3-1,\ (ababab^2)^3-1\}\\
\mathrm{tri3} =& \{a^3-1,\ b^3-1,\ (abab^2)^2-1\}\\
\mathrm{tri4} =& \{a^3-1,\ b^3-1,\ (abaab^2)^2-1\}\\
\mathrm{tri5} =& \{a^2-1,\ b^5-1,\ (abab^2)^2-1\}\\
\mathrm{tri6} =& \{a^2-1,\ b^5-1,\ (ababab^4)^2-1\}\\
\mathrm{tri7} =& \{a^2-1,\ b^5-1,\ (abab^2ab^4)^2-1\}\\
\mathrm{tri8} =& \{a^2-1,\ b^2-1,\ (ababab^3)^2-1\}\\
\mathrm{tri9} =& \{a^2-1,\ b^3-1,\ (abab^2)^2-1\}\\
\mathrm{tri10} =& \{a^2-1,\ b^3-1,\ (ababab^2)^2-1\}\\
\mathrm{tri11} =& \{a^2-1,\ b^3-1,\ (abababab^2)^2-1\}\\
\mathrm{tri12} =& \{a^2-1,\ b^3-1,\ (ababab^2abab^2)^2-1\}\\
\mathrm{tri13} =& \{a^2-1,\ b^3-1,\ (bababab^2ab^2)^2-1\}\\
\mathrm{trit3} =& \{a^3-1,\ b^3-1,\ c^3-1,\ (ab)^2-1,\ (ac)^2-1,\ (bc)^2-1\}\\
\mathrm{trit4} =& \{a^3-1,\ b^3-1,\ c^4-1,\ (ab)^2-1,\ (ac)^2-1,\ (bc)^2-1\}\\
\mathrm{trit5} =& \{a^3-1,\ b^3-1,\ c^5-1,\ (ab)^2-1,\ (ac)^2-1,\ (bc)^2-1\}
\end{aligned}
$$

Comparación



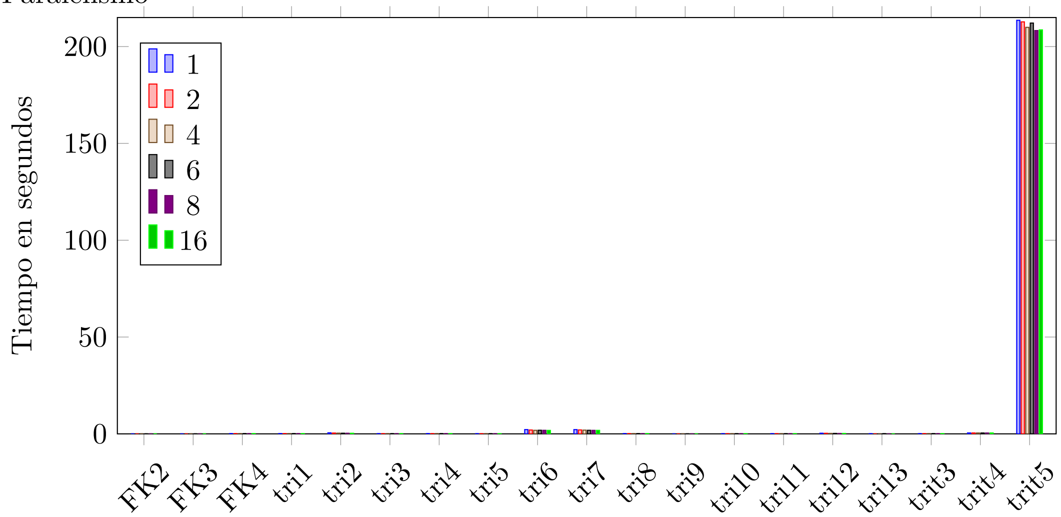
Comparación

Tiempo en calcular una base viendo la parte de más abajo



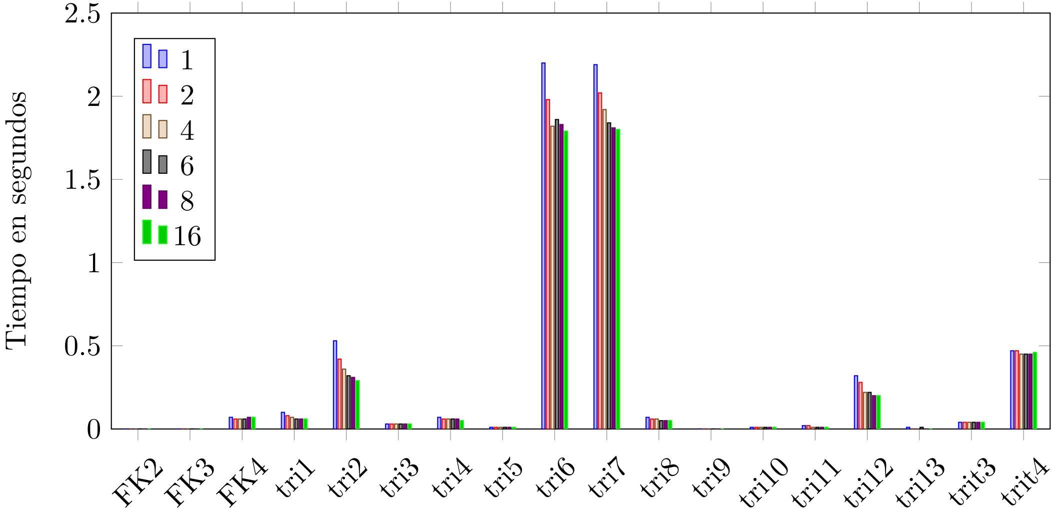
Comparación

Paralelismo



Comparación

Paralelismo sin trit5



Trabajos futuros

- Implementar la optimización de Faugère-Lachartre en F4
- Y si es posible paralelizarla