# Prefácio

A teoria de corpos finitos desenvolveu-se extensivamente no século XIX, porém a sua origem data dos séculos XVII e XVIII. Os primeiros pesquisadores a considerar corpos finitos foram Pierre de Fermat, Leonhard Euler, Joseph-Louis Lagrange, Adrien-Marie Legendre e Carl Gauss. Na época, os únicos corpos finitos conhecidos eram os corpos contendo um número primo de elementos.

A aparição do artigo "*Sur la théorie des nombres*" de Évariste Galois [44], em 1830, foi fundamental para o surgimento de várias questões quanto à estrutura de corpos finitos em geral. Em 1857, Richard Dedekind caracterizou corpos finitos com $p^n$ elementos, onde $p$ é primo, em termos de anéis quocientes de polinômios. Anos mais tarde, em 1893, Eliakim Moore mostrou que qualquer corpo finito contém $p^n$ elementos.

O livro de Leonard Dickson [35], publicado em 1901, já tinha os resultados mais importantes sobre a estrutura de corpos finitos. A seguir, apresentamos uma pequena lista com alguns desses resultados:

1. O número de elementos num corpo finito é uma potência da característica prima do corpo.

2. Se $p$ é primo e $n$ é um inteiro positivo, então existe um único corpo finito com $p^n$ elementos, a menos de isomorfismos.

3. O grupo multiplicativo dos elementos não nulos de um corpo finito é cíclico.

4. Seja $F$ um corpo com $p^n$ elementos. O número de elementos num subcorpo de $F$ é da forma $p^d$, onde $d$ é um divisor de $n$.

Reciprocamente, se $d$ divide $n$, então existe um subcorpo de $F$ com $p^d$ elementos.

5. Todo elemento $a$ num corpo finito com $q$ elementos satisfaz $a^q = a$.

No século XX, o uso de corpos finitos foi extremamente difundido, em parte devido à aparição dos computadores. Algumas áreas de aplicação de corpos finitos são: criptografia, teoria de códigos, processamento digital de sinais, seqüências pseudo-aleatórias, transformada discreta de Fourier e wavelets. Corpos finitos também aparecem relacionados às áreas da matemática como combinatória, geometria algébrica, geometria aritmética, geometria finita e teoria de números.

Este livro é uma introdução à teoria de corpos finitos e às suas aplicações em criptografia e teoria de códigos. Tentamos elaborar um texto que fosse auto-contido e que tivesse um mínimo de pré-requisitos. No entanto, um curso de álgebra e de álgebra linear, assim como uma certa maturidade algorítmica, seria adequado para uma melhor compreensão do conteúdo. Pretendemos, por meio destas notas, divulgar uma área que tem cada vez mais atraído pesquisadores na matemática, ciência da computação e engenharia. Sendo assim, ao longo dos capítulos, apresentamos vários problemas em aberto, que podem servir como temas para projetos de iniciação científica, mestrado ou mesmo doutorado. O leitor perceberá que muitos problemas nesta área são bastante simples de se entender, exigindo um conhecimento mínimo da teoria. Em algumas ocasiões, estaremos sendo breves com a descrição do problema, omitindo maiores detalhes. Gostaríamos de deixar claro que a nossa proposta, neste sentido, é apenas a de mostrar várias possíveis pontas de pesquisa que a teoria de corpos finitos proporciona, esperando assim motivá-lo para uma investigação mais profunda no assunto. Referências acerca de cada problema são fornecidas.

No Capítulo 1, apresentamos alguns conceitos e resultados na teoria de anéis e de corpos em geral para então discutirmos a estrutura de corpos finitos. Em particular, provamos os resultados da lista acima. Polinômios são elementos bastante presentes neste livro. Assim, também mencionamos alguns resultados básicos sobre os anéis de polinômios. A aritmética em corpos finitos é de grande

importância nas aplicações que envolvem implementações no computador. No Capítulo 2, discutimos alguns aspectos computacionais das operações aritméticas, dependendo da forma escolhida para representar os elementos num corpo finito, que pode ser via polinômios, elementos primitivos e elementos normais. O Capítulo 3 é dedicado aos polinômios irredutíveis sobre um corpo finito. Abordamos os problemas de testar se um polinômio é irredutível e de fatorar um polinômio como um produto de irredutíveis. No Capítulo 4, apresentamos alguns criptossistemas e o problema do logaritmo discreto. As aplicações na teoria de códigos aparecem no Capítulo 5. Nos Apêndices A, B e C, cobrimos alguns tópicos da teoria de números necessários neste livro.

Ao leitor que deseja se aprofundar em alguns dos tópicos cobertos neste livro, recomendamos os livros clássicos de corpos finitos de Lidl e Niederreiter [90] (para a teoria de corpos finitos), de von zur Gathen e Gerhard [56] (para algoritmos e operações aritméticas em corpos finitos), de Berlekamp [8] (para a teoria algébrica de códigos) e de Golomb [61] (para seqüências em corpos finitos e suas aplicações). As revistas científicas mais importantes, que têm contribuído para divulgar boa parte da pesquisa nesta área, são: *Finite Fields and Their Applications*, *IEEE Transactions on Information Theory* e *Designs, Codes and Cryptography*.

Finalmente, gostaríamos de agradecer às pessoas que contribuíram para a existência deste livro. Agradecemos a Yoshiharu Kohayakawa que sugeriu aos autores que propusessem um curso sobre corpos finitos para o 26° Colóquio Brasileiro de Matemática. Este livro contém as notas de aula deste curso. Agradecemos também aos organizadores do Colóquio pela excelente oportunidade.

Parte do material deste livro é proveniente das notas de aula dos cursos de "Corpos Finitos e Teoria de Códigos" e "Álgebra Computacional Moderna", ministrados nos últimos seis anos, na School of Mathematics and Statistics da Carleton University. Tais notas foram editadas pelos alunos: Gerald Boamah, Joanne Charlebois, Colette Haley, Zhaohuan Liu, Thomas Maloley, Zhiyuan Qiao, Sebastian Raaphorst, Stephen Savard, David Steeves e Benjamin Young.

Vários pesquisadores leram partes de versões preliminares deste livro. Agradecimentos especiais vão para Ricardo Dahab, Julio Cesar López Hernández, Lucia Moura, Virgínia Rodrigues, Rogério Siquei-

ra, Benjamin Steinberg e Vilmar Trevisan, por todos os comentários e sugestões úteis que ajudaram este livro a ter a forma presente. É claro que os autores se responsabilizam por qualquer erro que possa existir no livro. Uma lista de errata estará disponível em

`http://www.math.carleton.ca/~daniel/research/livroerrata.html`

Também agradecemos a Rodrigo Hausen pela ajuda técnica.

Ariane Masuda gostaria de agradecer à School of Mathematics and Statistics da Carleton University, onde parte deste livro foi produzida, ao Department of Mathematics and Statistics da University of Ottawa, onde a outra parte foi elaborada, e ao NSERC (Natural Sciences and Engineering Research Council of Canada) pelo apoio financeiro.

Daniel Panario gostaria de agradecer ao NSERC pelo contínuo apoio financeiro à sua pesquisa, ao Department de Llenguatges i Sistemes Informàtics da Universitat Politècnica de Catalunya, onde parte deste livro foi escrita durante o seu ano sabático, e ao Ministerio de Educación y Ciencia do Governo da España pelo apoio financeiro durante este ano.

Ariane Masuda e Daniel Panario
Ottawa e Barcelona, maio de 2007.

# Conteúdo

# Capítulo 1

# Fundamentos

O objetivo deste capítulo é introduzir o leitor à teoria de corpos finitos. Começaremos com uma breve apresentação da teoria de anéis e de corpos, nas Seções 1.1 e 1.2. Polinômios são uma ferramenta bastante presente nas áreas de criptografia e teoria de códigos; por isso, serão discutidos na Seção 1.3 com ênfase especial. Em seguida, na Seção 1.4, concentraremos nossa atenção em corpos finitos.

Uma referência para este capítulo é o livro de Hefez e Villela [69].

## 1.1 Anéis

**Definição 1.1.1.** Um *anel* $(R, +, \cdot)$ é um conjunto $R$ munido de duas operações binárias, adição e multiplicação, tal que

(a) $(R, +)$ é um grupo abeliano;

(b) $a \cdot (b \cdot c) = (a \cdot b) \cdot c$ para quaisquer $a, b, c \in R$; e

(c) $a \cdot (b + c) = a \cdot b + a \cdot c$, $(b + c) \cdot a = b \cdot a + c \cdot a$ para quaisquer $a, b, c \in R$.

Denotaremos por $0_R$, ou simplesmente por $0$, o elemento neutro da adição e por $-a$ o inverso aditivo de $a$. Segue dos axiomas acima

que $a \cdot 0_R = 0_R \cdot a = 0_R$ para qualquer $a$ num anel $R$. É comum denotarmos o produto $a \cdot b$ apenas por $ab$.

**Exemplo 1.1.2.**

1. $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$, $\mathbb{C}$ e $\mathbb{Z}_n$ são anéis com as operações usuais de adição e multiplicação.

2. Seja $R$ um anel. Dados dois polinômios $f = \sum_{i=0}^{n} a_i x^i$ e $g = \sum_{j=0}^{m} b_j x^j$ tais que cada $a_i, b_j \in R$ e $m \leq n$, definimos a soma $f + g$ e o produto $fg$ da seguinte maneira:

$$f+g = \sum_{k=0}^{n} (a_k + b_k) x^k \text{ e } fg = \sum_{k=0}^{m+n} (a_0 b_k + a_1 b_{k-1} + \cdots + a_k b_0) x^k,$$

onde $a_i = b_j = 0$ para quaisquer $i$, $j$ com $i > n$, $j > m$. O conjunto de todos os polinômios sobre $R$ com essas duas operações é um anel conhecido como o *anel dos polinômios sobre* $R$ e denotado por $R[x]$.

Anéis com propriedades adicionais têm nomes especiais.

**Definição 1.1.3.** Um anel $R$ é chamado

(a) um *anel com identidade*, se existe um elemento em $R$, denotado por $1_R$, tal que $a \cdot 1_R = 1_R \cdot a = a$ para todo $a \in R$ (neste caso, $1_R$ é chamado o *elemento neutro* da multiplicação, também simplesmente denotado por 1);

(b) um *anel comutativo*, se a multiplicação é comutativa, ou seja, $a \cdot b = b \cdot a$ para quaisquer $a, b \in R$;

(c) um *domínio de integridade*, se $R$ é um anel comutativo com identidade e se, para quaisquer $a, b \in R$ tais que $a \cdot b = 0$, tem-se que $a = 0$ ou $b = 0$;

(d) um *anel com divisão*, se os elementos não nulos de $R$ formam um grupo sob a multiplicação;

(e) um *corpo*, se é um anel comutativo com divisão.

Neste livro, todo anel será considerado um anel comutativo com identidade e assim, por simplicidade, diremos apenas anel.

**Exemplo 1.1.4.**

1. $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$, $\mathbb{C}$ e $\mathbb{Z}_p$ com $p$ primo são domínios de integridade. De fato, em $\mathbb{C}$, se $ab = 0$ então $a = 0$ ou $b = 0$. Em $\mathbb{Z}_p$, se $ab = 0$ então $p$ divide $ab$. Como $p$ é primo, segue que $p$ divide $a$ ou $b$.

2. $\mathbb{Z}_n$ com $n$ composto não é um domínio de integridade. De fato, escrevendo $n = ab$ com $0 < a, b < n$, vemos que $a$ e $b$ são ambos diferentes de zero módulo $n$.

3. Seja $R$ um anel. Se $f(x) = a_n x^n + \cdots + a_1 x + a_0$ é um polinômio sobre $R$ com $a_n \neq 0$, dizemos que cada $a_i$ é um *coeficiente* de $f$, $a_n$ é o *coeficiente do termo dominante* de $f$ e o *grau* de $f$ é $n$. O *polinômio nulo* tem todos os coeficientes iguais a zero e neste caso, por convenção, o grau é $-\infty$. Se $R$ é um domínio de integridade, então, em $R[x]$, temos que $\text{grau}(fg) = \text{grau}(f) + \text{grau}(g)$. Em particular, se $f$ e $g$ são polinômios não nulos em $R[x]$, então $fg$ também é não nulo. Isto mostra que, quando $R$ é um domínio de integridade, $R[x]$ também é um domínio de integridade.

4. $\mathbb{Z}$ não é um corpo já que, para qualquer inteiro $a \neq \pm 1$, não existe um inteiro $b$ tal que $ab = 1$. Por outro lado, temos que $\mathbb{Q}$, $\mathbb{R}$ e $\mathbb{C}$ são corpos.

5. $\mathbb{Z}_p$ com $p$ primo é um corpo. O seguinte resultado é uma prova deste fato.

**Teorema 1.1.5.** *Todo domínio de integridade finito é um corpo.*

*Demonstração.* Sejam $R$ um domínio de integridade finito e $a$ um elemento não nulo em $R$. Consideramos a aplicação $\phi\colon R \to R$ definida por $\phi(x) = ax$. Como $a(x - y) = 0$ implica que $x - y = 0$, a aplicação $\phi$ é injetora. Por outro lado, $R$ é finito e portanto $\phi$ também é sobrejetora; em particular, existe $\bar{a} \in R$ tal que $a\bar{a} = 1$. □

A recíproca deste resultado é mais geral: qualquer corpo é um domínio de integridade. Para isso, seja $F$ um corpo e suponhamos que $ab = 0$ onde $a, b \in F$ e $a \neq 0$. Seja $\bar{a} \in F$ tal que $\bar{a}a = 1$. Então

$$b = 1 \cdot b = (\bar{a}a)b = \bar{a}(ab) = \bar{a} \cdot 0 = 0.$$

Em vista do Exemplo 1.1.4 (2), estabelecemos a seguinte definição.

**Definição 1.1.6.** Um elemento não nulo $a$ num anel $R$ é um *divisor de zero*, se existe um elemento não nulo $b$ em $R$ tal que $ab = 0$.

**Exemplo 1.1.7.**

1. Um domínio de integridade não contém divisores de zero.

2. Os divisores de zero em $\mathbb{Z}_{12}$ são $2, 3, 4, 6, 8, 9$ e $10$.

As seguintes terminologias serão utilizadas freqüentemente.

**Definição 1.1.8.** Seja $R$ um anel.

(a) Um elemento $a \in R$ é um *divisor* de $b \in R$, se existe $c \in R$ tal que $b = ac$. Neste caso, escrevemos $a \mid b$.

(b) Um elemento não nulo $r$ é um *elemento invertível* em $R$, se existe $s \in R$ tal que $rs = 1$. Neste caso, escrevemos $r^{-1} = s$.

(c) Os elementos $a$ e $b$ em $R$ são *associados*, se existe um elemento invertível $r \in R$ tal que $a = rb$.

(d) Seja $p$ um elemento não nulo que não seja invertível em $R$. Dizemos que $p$ é *irredutível* em $R$, se $p = ab$ com $a, b \in R$ implica que $a$ ou $b$ seja invertível em $R$. Caso contrário, $p$ é *redutível* em $R$.

**Exemplo 1.1.9.**

1. Em $\mathbb{Z}$, os únicos elementos invertíveis são $1$ e $-1$. Para qualquer inteiro $a$, temos que $a$ e $-a$ são associados. Se $p$ é um número primo, então $p$ e $-p$ são ambos irredutíveis.

2. Num corpo, qualquer elemento não nulo é um elemento invertível.

3. Em $\mathbb{Z}_n$, um elemento não nulo $a$ é um elemento invertível se e somente se $a$ e $n$ são relativamente primos. Com efeito, a última condição é equivalente a ter inteiros $r$ e $s$ com $1 = ar + ns$ e assim $a^{-1} \equiv r \pmod{n}$. Em particular, se $n$ é primo, todos os elementos não nulos em $\mathbb{Z}_n$ são invertíveis. Isto dá uma outra justificativa ao fato de que $\mathbb{Z}_n$ é um corpo, sempre que $n$ for primo (veja Exemplo 1.1.4 (5)).

4. Seja $R$ um domínio de integridade. Usando a propriedade do grau de um produto de polinômios mencionada no Exemplo 1.1.4 (3), segue que os elementos invertíveis de $R$ e $R[x]$ coincidem.

5. Seja $F$ um corpo. Qualquer polinômio linear $ax + b$ com $a \neq 0$ é irredutível em $F[x]$.

6. Ao mencionarmos polinômios irredutíveis em $R[x]$, é comum também dizermos *polinômios irredutíveis sobre* $R$. Por exemplo, o polinômio $x^2 - 2 = (x - \sqrt{2})(x + \sqrt{2})$ é irredutível sobre $\mathbb{Q}$, mas é redutível sobre $\mathbb{R}$.

7. Vamos generalizar o exemplo anterior. Dizemos que $\alpha$ é uma *raiz* de $f$, se $f(\alpha) = 0$. Sejam $F$ um corpo, $f \in F[x]$ e $\alpha \in F$. Neste caso, o polinômio $x - \alpha$ divide $f$ em $F[x]$. Além disso, se $\text{grau}(f) > 1$, então $f$ é redutível sobre $F$. Isto dá um critério de irredutibilidade: se $\text{grau}(f) = 2$ ou 3 então $f$ é irredutível sobre $F$ se e somente se $f$ não tem raízes em $F$. A restrição no grau do polinômio se deve ao fato de que polinômios redutíveis de grau maior que 3 não necessariamente possuem fatores lineares. Por exemplo, em $\mathbb{R}[x]$, o polinômio $x^4 + 4x^2 + 3 = (x^2 + 1)(x^2 + 3)$ é redutível, mas não tem raízes em $\mathbb{R}$.

8. Em $\mathbb{C}[x]$, qualquer polinômio de grau maior que 1 é redutível sobre $\mathbb{C}$. Esta é um conseqüência imediata do *Teorema Fundamental da Álgebra*, que garante que tais polinômios sempre têm uma raiz em $\mathbb{C}$.

Sejam $R$ um anel, $a \in R$ e $n$ um inteiro. Definimos o produto $na$

por

$$na = \begin{cases} 0 & \text{se } n = 0 \\ \underbrace{a + \cdots + a}_{n \text{ vezes}} & \text{se } n > 0 \\ -((-n)a) & \text{se } n < 0. \end{cases}$$

Esta multiplicação tem a propriedade de que $(na)(mb) = (nm)(ab)$ para todos os inteiros $n$, $m$ e para todos os elementos $a, b$ em $R$. Usando esta operação, definimos agora a *característica* de um anel.

**Definição 1.1.10.** Seja $R$ um anel para o qual existe um inteiro positivo $n$ tal que

$$n \cdot 1 = \underbrace{1 + \cdots + 1}_{n \text{ vezes}} = 0.$$

O menor tal inteiro positivo $n$ é chamado a *característica de* $R$ e denotado por $\text{car}(R)$. Neste caso, também dizemos que $R$ tem *característica positiva*; além disso, se $n$ é primo, então dizemos que $R$ tem *característica prima*. Se $R$ não tem característica positiva, dizemos que $R$ tem *característica zero*.

**Exemplo 1.1.11.** $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$ e $\mathbb{C}$ têm característica zero, enquanto $\mathbb{Z}_n$ tem característica $n$ quando $n \geq 2$.

**Teorema 1.1.12.** *Seja $R$ um anel. Se $R$ tem característica positiva e se $R$ não contém divisores de zero, então a característica de $R$ é prima.*

*Demonstração.* Seja $n \geq 2$ a característica de $R$. Escrevendo $n = ab$, onde $a, b \in \mathbb{N}$ e $1 < a, b < n$, obtemos que $0 = n \cdot 1 = (ab) \cdot 1 = (a \cdot 1)(b \cdot 1)$. Como $R$ não contém divisores de zero, temos que $a \cdot 1 = 0$ ou $b \cdot 1 = 0$. Em qualquer caso, temos uma contradição, pois $n$ é a característica. □

Sejam $R$ um anel e $S$ um subconjunto de $R$. Dizemos que $S$ é um *subanel* de $R$, se $S$ também é um anel sob as mesmas operações de $R$ com o mesmo elemento neutro da multiplicação de $R$. O seguinte critério fornece um teste muito útil para verificar que um subconjunto de um anel é de fato um subanel.

**Teorema 1.1.13.** *Um subconjunto $S$ de um anel $R$ é um subanel se e somente se $1_R \in S$ e, para quaisquer $s$, $t$ em $S$, temos que $s-t$ e $st$ pertencem a $S$.*

Não é difícil verificar que as condições acima são suficientes, já que $S$ herda várias propriedades por ser um subconjunto de um anel. Usando o teste acima, podemos comprovar rapidamente os seguintes exemplos.

**Exemplo 1.1.14.**

1. $\mathbb{Z}[i] = \{m + ni \colon m, n \in \mathbb{Z}\}$ é um subanel de $\mathbb{C}$, também conhecido como o *anel dos inteiros de Gauss*.
2. $\mathbb{R}[x^2] = \{f(x^2) \colon f \in \mathbb{R}[x]\}$ é um subanel de $\mathbb{R}[x]$.
3. $\mathbb{Z}\left[\frac{1}{2}\right] = \left\{\frac{m}{2^n} \colon m, n \in \mathbb{Z},\ n \geq 0\right\}$ é um subanel de $\mathbb{Q}$.

Alguns anéis têm a mesma estrutura algébrica. Esta semelhança pode ser expressa por meio de uma aplicação que preserva os elementos neutros da multiplicação, a adição e a multiplicação.

**Definição 1.1.15.** Sejam $R$ e $S$ anéis cujos elementos neutros da multiplicação são $1_R$ e $1_S$, respectivamente. Um *homomorfismo de anéis* é uma aplicação $\phi \colon R \to S$ satisfazendo $\phi(1_R) = 1_S$, $\phi(a+b) = \phi(a) + \phi(b)$, $\phi(ab) = \phi(a)\phi(b)$ para quaisquer $a, b \in R$. Além disso, se $\phi$ é bijetora, dizemos que $\phi$ é um *isomorfismo* e escrevemos $R \cong S$ para dizer que $R$ e $S$ são *isomorfos*.

Como conseqüência desta definição, qualquer homomorfismo também preserva zeros, negativos, multiplicações por inteiros e potências. Isto significa que se $0_R$ e $0_S$ são os zeros de $R$ e $S$, respectivamente, então $\phi(0_r) = 0_S$, $\phi(-a) = -\phi(a)$, $\phi(na) = n\phi(a)$ e $\phi(a^k) = \phi(a)^k$ para quaisquer $a \in R$, $n \in \mathbb{Z}$ e $k \in \mathbb{N}$.

**Exemplo 1.1.16.**

1. A aplicação $\phi \colon \mathbb{Z} \to \mathbb{Z}_2$ definida por $\phi(a) = \begin{cases} 0, & \text{se } a \text{ é par} \\ 1, & \text{se } a \text{ é ímpar} \end{cases}$ é um homomorfismo.

2. A aplicação $\phi\colon \mathbb{Z}[i] \to \mathbb{Z}[i]$ definida por $\phi(a+bi) = a - bi$ é um isomorfismo.

3. O anel $R$ formado pelos elementos $0, 8, 16$ de $\mathbb{Z}_{24}$ é isomorfo a $\mathbb{Z}_3$, já que a aplicação $\phi\colon R \to \mathbb{Z}_3$ dada por $\phi(0) = 0$, $\phi(8) = 2$ e $\phi(16) = 1$ é um isomorfismo.

Se $\phi\colon R \to S$ é um homomorfismo de anéis, a *imagem* de $\phi$, definida por $\phi(R) = \{s \in S\colon s = \phi(r) \text{ para algum } r \in R\}$, é um subanel de $S$. Para verificarmos isso, observamos inicialmente que $1_S = \phi(1_R)$, ou seja, $1_S \in \phi(R)$. Suponhamos que $s_1$ e $s_2$ estejam na imagem de $\phi$, digamos, $\phi(r_1) = s_1$ e $\phi(r_2) = s_2$ com $r_1, r_2 \in R$. Então $\phi(r_1 - r_2) = s_1 - s_2$ e $\phi(r_1 r_2) = s_1 s_2$. Logo, os elementos $s_1 - s_2$ e $s_1 s_2$ pertencem a $\phi(R)$. No final desta seção, analisaremos uma estrutura algébrica que é isomorfa a $\phi(R)$ (Teorema 1.1.24).

Isomorfismos serão importantes para entendermos a estrutura de um corpo finito. Na Seção 1.4, determinaremos exatamente os valores de $q$ para os quais um corpo contendo $q$ elementos existe. Além disso, quando um corpo de $q$ elementos existe, veremos que é único a menos de isomorfismos. Um outro conceito necessário neste resultado é o de *anel quociente*, mas primeiro precisamos ver o que é um ideal.

**Definição 1.1.17.** Um subgrupo aditivo $I$ de um anel $R$ é um *ideal* de $R$, se $ar \in I$ para quaisquer $a \in I$ e $r \in R$.

**Exemplo 1.1.18.**

1. Em qualquer anel $R$, os ideais triviais são $\{0\}$ e $R$. O ideal $0 = \{0\}$ é conhecido como o *ideal zero*. Qualquer ideal $I$ de $R$ é um *ideal próprio*, se $I \neq R$.

2. Para qualquer $n \geq 0$, o conjunto $n\mathbb{Z}$ de todos os múltiplos de $n$ é um ideal de $\mathbb{Z}$.

3. O exemplo anterior pode ser generalizado para qualquer anel $R$. Dado $a \in R$, o conjunto $\{ra : r \in R\}$ é um ideal de $R$ chamado o *ideal gerado por* $a$ e denotado por $(a)$.

4. Sejam $R$ e $S$ anéis. Dado um homomorfismo de anéis $\phi\colon R \to S$, o *núcleo* de $\phi$, definido por $\ker\phi = \{r \in R\colon \phi(r) = 0\}$, é um

ideal de $R$. De fato, suponhamos que $a$ e $b$ estejam no núcleo de $\phi$. Segue que $\phi(a-b)=0$ e $\phi(ar)=0$ para qualquer $r \in R$.

O núcleo de um homomorfismo pode ser usado para verificar que o homomorfismo é injetor. Com efeito, se $\phi$ é um homomorfismo, então $\phi$ é injetor se e somente se $\ker\phi = \{0\}$. A demonstração é bastante simples. Se $\phi$ é injetor e $x \in \ker\phi$ então $x = 0$, pois $\phi(0) = 0$. Reciprocamente, se $\ker\phi = \{0\}$ e $\phi(r_1) = \phi(r_2)$, temos que $\phi(r_1 - r_2) = 0$, ou seja, $r_1 - r_2 \in \ker\phi$; logo $r_1 = r_2$.

A partição de um anel em conjuntos disjuntos conhecidos como *classes laterais* é bastante útil no estudo de anéis, pois dá uma maneira de construir novos anéis a partir de dados anéis.

**Definição 1.1.19.** Sejam $R$ um anel e $I$ um ideal de $R$. O conjunto $r + I = \{r + a\colon a \in I\}$ com $r \in R$ forma uma *classe lateral* de $I$ em $R$. O conjunto de todas as classes laterais de $I$ em $R$ é conhecido como o *anel quociente* de $R$ por $I$ e é denotado por $R/I$.

A *relação*[1] em $R$ induzida por esta definição é dada pela igualdade $r+I = s+I$, sempre que $r-s \in I$. Esta é uma *relação de equivalência*[2] e portanto as classes laterais de $I$ em $R$ formam uma partição[3] de $R$. Definindo as operações de adição e de multiplicação em $R/I$ de maneira natural, não é difícil ver que $R/I$ é um anel.

**Teorema 1.1.20.** *Seja $I$ um ideal de um anel $R$. Então $R/I$ é um anel com a adição e a multiplicação definidas por*

$$(r + I) + (s + I) = (r + s) + I \qquad e \qquad (r + I)(s + I) = rs + I,$$

*respectivamente.*

De fato, em $R/I$, o elemento neutro da adição é $0 + I$, o inverso aditivo de $r+I$ é $-r+I$, o elemento neutro da multiplicação é $1_R+I$ e todas as outras condições para ser um anel são satisfeitas já que $R$ é um anel.

[1] Qualquer subconjunto $S \subseteq R\times R$ é uma *relação* em $R$. Denotamos tal relação por $\sim$, onde $a \sim b$, se $(a, b) \in S$.

[2] A relação $\sim$ é *de equivalência*, se $\sim$ é reflexiva, simétrica e transitiva.

[3] Se $\sim$ é uma relação de equivalência, dizemos que a *classe de equivalência* de $a$ é formada por todos os elementos $b \in R$ tais que $a \sim b$. Neste caso, o conjunto de todas as classes de equivalência forma uma *partição* de $R$.

**Exemplo 1.1.21.**

1. $\mathbb{Z}/n\mathbb{Z} \cong \mathbb{Z}_n$

2. Seja $R = \mathbb{Z}[i]$ o anel dos inteiros de Gauss e seja $I$ o ideal gerado por $2+3i$. Mostraremos que $R/I$ é um corpo descrevendo suas classes laterais. Seja $\alpha = (m+ni)+I \in R/I$, onde $m, n \in \mathbb{Z}$. Como $(1-i)(2+3i) = 5+i$, temos que $i+I = -5+I$ e assim $\alpha = (m-5n)+I$. Em outras palavras, $\alpha = a+I$ onde $a \in \mathbb{Z}$. Uma outra redução é obtida observando que $13 = (2+3i)(2-3i)$, o que implica que $13+I = 0+I$. Escrevendo $a = 13\ell + r$ com $0 \leq r \leq 12$ dá que $\alpha = r+I$. Além disso, quaisquer duas tais classes laterais são distintas. Para mostrarmos isso, suponhamos que $r+I = r'+I$ com $0 \leq r, r' \leq 12$. Então $r-r' = (2+3i)(m+ni)$ para certos $m, n \in \mathbb{Z}$. Tomando os quadrados dos valores absolutos de ambos os lados produz $(r-r')^2 = 13(m^2+n^2)$, o que significa que 13 divide $r-r'$, i.e. $r = r'$. Logo $R/I = \{r+I : 0 \leq r \leq 12\}$ possui 13 elementos. Além disso, a aplicação $\phi : R/I \to \mathbb{Z}_{13}$, definida por $\phi(r+I) = r$, é um isomorfismo. Como $\mathbb{Z}_{13}$ é um corpo, concluímos que $R/I$ também é um corpo.

Na verdade, não é uma coincidência que no último exemplo obtivemos um anel quociente $R/I$ que é um corpo. De fato, caracterizaremos os ideais que precisamente produzem anéis quocientes que são domínios de integridade e corpos. Isto será realizado em termos de ideais especiais.

**Definição 1.1.22.** Seja $R$ um anel.

(a) Um ideal $I$ de $R$ é *principal*, se existe $a \in R$ tal que $I = (a)$. Neste caso, $I$ também é chamado o *ideal principal gerado por* $a$.

(b) Um ideal $P$ de $R$ é *primo*, se $P \neq R$ e se $ab \in P$ implica que $a \in P$ ou $b \in P$.

(c) Um ideal $M$ de $R$ é *maximal*, se $M \neq R$ e se os únicos ideais $I$ de $R$ tais que $M \subseteq I \subseteq R$ são $I = M$ e $I = R$.

(d) O anel $R$ é um *domínio de ideais principais*, se $R$ é um domínio de integridade e se cada ideal $I$ de $R$ é principal.

Agora podemos mostrar os resultados anunciados anteriormente.

**Teorema 1.1.23.** *Seja $R$ um anel.*

1. *Um ideal $M$ de $R$ é maximal se e somente se $R/M$ é um corpo.*
2. *Um ideal $P$ de $R$ é primo se e somente se $R/P$ é um domínio de integridade.*
3. *Todo ideal maximal é primo.*
4. *Se $R$ é um domínio de ideais principais, então $R/(p)$ é um corpo se e somente se $p$ é irredutível em $R$.*
5. *Se $R$ é um domínio de ideais principais e $p \neq 0$, então $(p)$ é um ideal primo se e somente se $(p)$ é um ideal maximal.*

*Demonstração.* Começaremos com a prova de (1). Seja $M$ um ideal maximal de $R$. Dado $\alpha \in R \setminus M$, é suficiente mostrar que $\alpha + M$ é invertível em $R/M$. Para isso, vamos mostrar que $A = \{\alpha r + m \colon r \in R, m \in M\}$ é um ideal contendo $M$. Claramente, $A$ é um subgrupo aditivo que contém $M$. Além disso, para todo $r' \in R$, temos que $(\alpha r + m)r' = \alpha rr' + mr'$ pertence a $A$ e portanto $A$ é um ideal. Já que $\alpha \notin M$ e $\alpha = \alpha \cdot 1 + 0 \in A$, segue que $M \neq A$. Como $M$ é maximal, obtemos que $A = R$. Em particular, $1 = \alpha r + m$ com $r \in R$ e $m \in M$. Portanto, $(\alpha + M)(r + M) = 1 + M$.

Reciprocamente, seja $I$ um ideal de $R$ tal que $I \neq M$ e $M \subseteq I$. Seja $a \in I \setminus M$. Existe $r \in R$ satisfazendo $(a + M)(r + M) = 1 + M$ e assim $ar + m = 1$ para algum $m \in M$. Como $ar + m \in I$, segue que $1 \in I$ e assim $I = R$. Logo $M$ é maximal.

Para demonstrar (2), observamos que o anel quociente $R/P$ é um domínio de integridade se e somente se $(a+P)(b+P) = 0+P$ implica que $a + P = 0 + P$ ou $b + P = 0 + P$, ou equivalentemente se $ab \in P$ implica que $a \in P$ ou $b \in P$.

Se $M$ é um ideal maximal de $R$ então $R/M$ é um corpo por (1), logo também é um domínio de integridade. Por (2), concluímos que o ideal $M$ é primo. Isto prova (3).

Para a prova de (4), suponhamos que $(p)$ seja maximal e que $p = ab$. Então $(p) \subseteq (a)$, o que implica que $(a) = (p)$ ou $(a) = R$. No primeiro caso, temos que $a = pr$ para algum $r \in R$, isto é, $p = prb$, ou equivalentemente, $p(1 - rb) = 0$. Como $R$ é um domínio de integridade, segue que $rb = 1$, ou seja, $b$ é invertível. No outro caso, quando $(a) = R$, temos que $a$ é invertível, pois $1 \in R$ e portanto $ra = 1$ para algum $r \in R$. Em qualquer caso, temos que $p$ é irredutível.

Reciprocamente, suponhamos que $p$ seja irredutível. Então $p$ não é invertível e portanto $(p) \neq R$. Suponhamos que $(p) \subseteq (a)$. Segue que $p = ar$ onde $r \in R$. Como $p$ é irredutível, obtemos que $a$ ou $r$ é invertível, o que implica que $(a) = R$ ou $(a) = (p)$. Logo $(p)$ é maximal.

Finalmente, provamos (5). Sabemos que os ideais maximais são primos por (3). Então, basta provarmos que $(p)$ primo implica que $(p)$ seja maximal quando $p \neq 0$. De acordo com (4), é suficiente demonstar que $p$ é irredutível. Suponhamos que $p = ab$. Então, $ab = p \in (p)$ e portanto $a \in (p)$ ou $b \in (p)$. Suponhamos sem perda de generalidade que $a \in (p)$, ou seja, $a = pc$ com $c \in R$. Portanto, $p = pcb$, isto é, $p(1 - cb) = 0$. Como $p \neq 0$, temos que $cb = 1$, logo $b$ é invertível. Concluímos que $p$ é irredutível e assim $(p)$ é maximal por (4). □

Para terminar esta seção, apresentamos o teorema de isomorfismo de anéis.

**Teorema 1.1.24.** *Se $\phi\colon R \to S$ é um homomorfismo de anéis, então $R/\ker\phi$ é isomorfo a $\phi(R)$.*

*Demonstração.* Vamos mostrar que a aplicação $\Phi\colon R/\ker\phi \to \phi(R)$ com $\Phi(r + \ker\phi) = \phi(r)$ é um isomorfismo de anéis. Primeiramente, veremos que $\Phi$ está bem definida e é injetora. Sejam $r_1, r_2 \in R$. Temos que $r_1 + \ker\phi = r_2 + \ker\phi$ se e somente se $\phi(r_1 - r_2) = 0$, o que é equivalente a $\phi(r_1) = \phi(r_2)$. Como $\phi$ é um homomorfismo, segue imediatamente que $\Phi$ também é um homomorfismo. Além disso, $\Phi$ é claramente sobrejetora. □

## 1.2 Corpos

Sejam $F$ um corpo e $K$ um subconjunto de $F$. Se $K$ também é um corpo sob as operações de $F$, dizemos que $K$ é um *subcorpo* de $F$ e que $F$ é uma *extensão* de $K$.

**Exemplo 1.2.1.** $\mathbb{Q}$ é um subcorpo de $\mathbb{R}$ e $\mathbb{C}$; $\mathbb{R}$ é um subcorpo de $\mathbb{C}$.

Similarmente ao teste de subanel (Teorema 1.1.13), o seguinte critério determina se um subconjunto de um corpo é um subcorpo.

**Teorema 1.2.2.** *Um subconjunto $K$ de um corpo $F$ é um subcorpo se e somente se as seguintes condições são satisfeitas:*

1. *$K$ contém pelo menos um elemento não nulo;*
2. *se $a, b \in K$ então $a - b \in K$;*
3. *se $a, b \in K$ e $b \neq 0$ então $ab^{-1} \in K$.*

O subcorpo $K$ é *próprio*, se $K \neq F$. Um corpo que não contém nenhum subcorpo próprio é chamado um *corpo primo*.

**Exemplo 1.2.3.**

1. $\mathbb{Z}/p\mathbb{Z}$ é um corpo primo. Com efeito, se $K$ é um subcorpo de $\mathbb{Z}/p\mathbb{Z}$, então 1 deve estar em $K$. Como $K$ é fechado para a adição, todo elemento em $\mathbb{Z}/p\mathbb{Z}$ está em $K$ e assim $K = \mathbb{Z}/p\mathbb{Z}$.
2. $\mathbb{Q}$ é um corpo primo, porque qualquer subcorpo de $\mathbb{Q}$ contém 1, mas então contém $\mathbb{Q}$, já que é fechado para a adição e por inversos.

O próximo resultado mostra que os dois exemplos acima na verdade representam todos os corpos primos a menos de isomorfismos, já que um corpo primo pode ser visto como um subcorpo gerado pelo elemento neutro 1.

**Teorema 1.2.4.** *O subcorpo primo de um corpo $F$ é isomorfo a $\mathbb{Z}/p\mathbb{Z}$ ou a $\mathbb{Q}$ de acordo com a característica de $F$ ser prima ou zero.*

*Demonstração.* Seja $\phi\colon \mathbb{Z} \to F$ o homomorfismo de anéis definido por $\phi(n) = n \cdot 1_F$. O núcleo de $\phi$ é $(\mathrm{car}F)\mathbb{Z}$. Se $\mathrm{car}F = p$ para algum primo $p$, então, pelo Teorema 1.1.24, $\phi(\mathbb{Z}) \cong \mathbb{Z}/p\mathbb{Z}$ que é um corpo primo. Se $\mathrm{car}F = 0$, então $\phi$ é injetora. Neste caso, $\phi(\mathbb{Z})$ é um anel isomorfo a $\mathbb{Z}$. Definimos agora $\phi'\colon \mathbb{Q} \to F$ por $\phi'(m/n) = (m \cdot 1_F)(n \cdot 1_F)^{-1}$, se $n \neq 0$. Temos que $\phi'$ é um homomorfismo injetor. Com efeito,

$$\begin{aligned}\phi'\left(\frac{m_1}{n_1} + \frac{m_2}{n_2}\right) &= ((m_1n_2 + m_2n_1) \cdot 1_F)((n_1n_2) \cdot 1_F)^{-1} \\ &= (m_1 \cdot 1_F)(n_1 \cdot 1_F)^{-1} + (m_2 \cdot 1_F)(n_2 \cdot 1_F)^{-1} \\ &= \phi'\left(\frac{m_1}{n_1}\right) + \phi'\left(\frac{m_2}{n_2}\right)\end{aligned}$$

e

$$\begin{aligned}\phi'\left(\frac{m_1}{n_1} \cdot \frac{m_2}{n_2}\right) &= ((m_1m_2) \cdot 1_F)((n_1n_2) \cdot 1_F)^{-1} \\ &= (m_1 \cdot 1_F)(n_1 \cdot 1_F)^{-1}(m_2 \cdot 1_F)(n_2 \cdot 1_F)^{-1} \\ &= \phi'\left(\frac{m_1}{n_1}\right) \cdot \phi'\left(\frac{m_2}{n_2}\right).\end{aligned}$$

Além disso, se $\phi'(m/n) = 0$ então $m1_F = 0$ e portanto $m = 0$, já que $\mathrm{car}F = 0$. Concluímos que $\phi'(\mathbb{Q})$, que é o menor subcorpo de $F$ contendo $1_F$, é isomorfo ao corpo primo $\mathbb{Q}$. □

A intersecção de subcorpos de um corpo é um subcorpo. Logo, corpos primos também podem ser obtidos considerando a intersecção da coleção de todos os subcorpos de um dado corpo. Na verdade, a idéia de tomar intersecções pode ser generalizada para obter outros corpos.

**Definição 1.2.5.** Sejam $K$ um subcorpo do corpo $F$ e $M$ um subconjunto de $F$. O corpo $K(M)$ é definido como a intersecção de todos os subcorpos de $F$ contendo ambos $K$ e $M$ e é chamado o *corpo de extensão de* $K$ obtido pela *adjunção* dos elementos de $M$. Quando $M = \{\theta_1, \theta_2, \ldots, \theta_n\}$, escrevemos $K(M) = K(\theta_1, \theta_2, \ldots, \theta_n)$.

Quando $M = \{\theta\}$, dizemos que $L = K(\theta)$ é uma *extensão simples* de $K$.

Se $L$ é um corpo de extensão de $K$, então $L$ pode ser visto como um espaço vetorial sobre $K$. Primeiramente, isto deve-se ao fato de que os elementos de $L$ formam um grupo abeliano sob a adição. Além disso, a multiplicação de um elemento $\alpha \in L$ por um escalar $r \in K$ dá $r\alpha \in L$ satisfazendo

$$\begin{aligned} r(\alpha+\beta) &= r\alpha + r\beta \\ (r+s)\alpha &= r\alpha + s\alpha \\ (rs)\alpha &= r(s\alpha) \\ 1_K \cdot \alpha &= \alpha \end{aligned}$$

para quaisquer $r, s \in K$ e $\alpha, \beta \in L$. Quando $L$ é um espaço vetorial de dimensão finita sobre $K$, a dimensão é o *grau* de $L$ sobre $K$ e é denotada por $[L : K]$. Neste caso, dizemos que $L$ é uma *extensão finita* de $K$.

**Teorema 1.2.6.** *Se $M$ é uma extensão finita de $L$ e $L$ é uma extensão finita de $K$, então $M$ é uma extensão finita de $K$ com $[M : K] = [M : L][L : K]$.*

*Demonstração.* Suponhamos que $\{\alpha_1, \alpha_2, \ldots, \alpha_m\}$ e $\{\beta_1, \beta_2, \ldots, \beta_n\}$ sejam bases de $M$ sobre $L$ e de $L$ sobre $K$, respectivamente. Então $[M : L] = m$, $[L : K] = n$ e queremos provar que $[M : K] = mn$. Qualquer elemento $\alpha \in M$ pode ser escrito como

$$\alpha = \gamma_1\alpha_1 + \gamma_2\alpha_2 + \cdots + \gamma_m\alpha_m$$

onde $\gamma_1, \ldots, \gamma_m \in L$. Agora, para cada $i$ tal que $1 \leq i \leq m$, temos

$$\gamma_i = r_{i1}\beta_1 + r_{i2}\beta_2 + \cdots + r_{in}\beta_n$$

onde $r_{i1}, \ldots, r_{in} \in K$. Logo, combinando as duas expressões acima dá que

$$\alpha = \sum_{i=1}^{m} \gamma_i\alpha_i = \sum_{i=1}^{m}\sum_{j=1}^{n} r_{ij}\beta_j\alpha_i$$

onde cada $r_{ij}$ pertence a $K$. A seguir, mostraremos que o conjunto $\{\beta_j\alpha_i \colon 1 \leq j \leq n, 1 \leq i \leq m\}$ é linearmente independente e assim

é uma base de $M$ sobre $K$. Seja $\sum_{i=1}^{m}\sum_{j=1}^{n} s_{ij}\beta_j\alpha_i = 0$ onde cada $s_{ij} \in K$. Como $\{\alpha_1, \ldots, \alpha_m\}$ forma uma base de $M$ sobre $L$, obtemos $\sum_{j=1}^{n} s_{ij}\beta_j = 0$ para qualquer $i$. Finalmente, como $\{\beta_1, \ldots, \beta_n\}$ é uma base de $L$ sobre $K$, concluímos que cada $s_{ij}$ é zero. □

## 1.3 Polinômios

Começamos esta seção com um resultado importante sobre polinômios.

**Teorema 1.3.1.** [Algoritmo da divisão] *Sejam $F$ um corpo e $f, g \in F[x]$ com $g \neq 0$. Existem únicos polinômios $h$, $r \in F[x]$ tais que $f = gh + r$ e* $\text{grau}(r) < \text{grau}(g)$.

*Demonstração.* Seja $S = \{f - gh : h \in F[x]\}$. Digamos que $r = f - gh$ é um polinômio em $S$ de menor grau e que $\text{grau}(r) = m$, $\text{grau}(g) = n$. Mostraremos que $m < n$.

Suponhamos por contradição que $m \geq n$. Seja $a \in F$ tal que o produto de $a$ e o coeficiente do termo dominante de $g$ dá o coeficiente do termo dominante de $r$. Subtraindo $ax^{m-n}g$ de ambos os lados de $r = f - gh$ dá que

$$r - ax^{m-n}g = f - gh - ax^{m-n}g,$$

onde o polinômio do lado esquerdo tem grau menor que $m$. O lado direito é igual a $f - g(h + ax^{m-n}) \in S$. Isto contradiz o fato de que $r$ é o polinômio de menor grau em $S$. Assim, existem polinômios $h, r \in F[x]$ tais que $f = gh + r$ e $m < n$.

Suponhamos que existam polinômios $h_1$, $h_2$, $r_1$, $r_2$ em $F[x]$ tais que $f = gh_1 + r_1$, $f = gh_2 + r_2$, $\text{grau}(r_1) < \text{grau}(g)$ e $\text{grau}(r_2) < \text{grau}(g)$. Assim, $gh_1 + r_1 = gh_2 + r_2$ implica que $g(h_1 - h_2) = r_2 - r_1$. Como $\text{grau}(r_2 - r_1) < \text{grau}(g)$, temos que $h_1 - h_2 = 0$ e assim $r_2 - r_1 = 0$. Logo $h_1 = h_2$ e $r_1 = r_2$. □

Seja $R$ um anel. Para que o algoritmo da divisão seja válido em $R[x]$, é necessário que o coeficiente do termo dominante de $g$ seja invertível em $R$.

**Definição 1.3.2.** Seja $F$ um corpo. Dados $f$ e $g \in F[x]$, existe um único polinômio mônico $d \in F[x]$ tal que

(a) $d$ divide $f$ e $g$,

(b) qualquer polinômio $h \in F[x]$ dividindo ambos $f$ e $g$ divide também $d$.

Este polinômio $d$ é o *máximo divisor comum de $f$ e $g$*, denotado por $\text{mdc}(f, g)$.

Observamos que o $\text{mdc}(f, g)$ é o polinômio mônico de maior grau dentre os polinômios que dividem ambos $f$ e $g$ em $F[x]$.

A existência do $\text{mdc}(f, g)$ vem do fato de que $F[x]$ é um domínio de ideais principais. Isto será provado mais adiante (Teorema 1.3.6). De fato, veremos que $f$ e $g$ têm um divisor comum $d$ tal que $d = af + bg$ para certos $a, b \in F[x]$. Desta expressão, resulta que qualquer divisor comum de $f$ e $g$ divide $d$.

A fim de encontrar o $\text{mdc}(f, g)$, apresentaremos agora o *algoritmo euclideano*, que é baseado em várias aplicações do algoritmo da divisão. Começamos com a divisão de $f$ por $g$, supondo que o resto desta divisão seja $r_1$. A próxima divisão é a de $g$ por $r_1$ com resto $r_2$, digamos. Dividindo $r_1$ por $r_2$ e continuando com este processo, obtemos:

$$\begin{aligned} f &= q_1 g + r_1 \\ g &= q_2 r_1 + r_2 \\ r_1 &= q_3 r_2 + r_3 \\ &\vdots \\ r_{s-3} &= q_{s-1} r_{s-2} + r_{s-1} \\ r_{s-2} &= q_s r_{s-1} + r_s \\ r_{s-1} &= q_{s+1} r_s + 0, \end{aligned} \tag{1.1}$$

onde $q_1, \ldots, q_{s+1}, r_1, \ldots, r_s$ são polinômios sobre $F$ e

$$\text{grau}(g) > \text{grau}(r_1) > \text{grau}(r_2) > \cdots > \text{grau}(r_{s-1}) > \text{grau}(r_s) \geq 0.$$

Como o $\text{grau}(g)$ é finito, este processo de divisões sempre termina com uma divisão exata. Quando isso acontece, temos que $\text{mdc}(f, g) = a^{-1} r_s$, onde $a$ é o coeficiente do termo dominante do último resto não nulo $r_s$. Esta multiplicação de $r_s$ por $a^{-1}$ é realizada para que tenhamos um polinômio mônico.

**Exemplo 1.3.3.**

1. Sejam $f(x) = x^3 + 3$ e $g(x) = 2x^2 + 2$ onde $f, g \in \mathbb{Z}_5[x]$. Temos que

$$\begin{aligned} x^3 + 3 &= 3x(2x^2 + 2) + (4x + 3) \\ 2x^2 + 2 &= (3x + 4)(4x + 3) + 0. \end{aligned}$$

   Então $\text{mdc}(x^3+3, 2x^2+2) = (4^{-1})(4x+3) = 4(4x+3) = x+2$.

2. Sejam $f(x) = 2x^6 + x^3 + x^2 + 2$ e $g(x) = x^4 + x^2 + 2x$ onde $f, g \in \mathbb{Z}_3[x]$. Temos que

$$\begin{aligned} 2x^6 + x^3 + x^2 + 2 &= (2x^2 + 1)(x^4 + x^2 + 2x) + (x + 2) \\ x^4 + x^2 + 2x &= (x^3 + x^2 + 2x + 1)(x + 2) + 1 \\ x + 2 &= (x + 2)(1) + 0. \end{aligned}$$

   Então $\text{mdc}(f, g) = 1$.

Quando $\text{mdc}(f, g) = 1$, dizemos que $f$ e $g$ são polinômios *coprimos* ou *relativamente primos*.

Como dissemos antes, é possível escrever o $\text{mdc}(f, g)$ como uma combinação linear de $f$ e $g$, no sentido de que $\text{mdc}(f, g) = af + bg$ para certos polinômios $a$ e $b$ em $F[x]$. Esta representação é muito útil como veremos mais tarde. Vamos ver agora como encontrar tais polinômios $a$ e $b$ usando o *algoritmo euclideano estendido*. Como o próprio nome sugere, estaremos usando dados das divisões efetuadas no algoritmo euclideano. Escrevemos o resto de cada divisão, em termos dos outros polinômios envolvidos em cada expressão de (1.1). Começando de baixo para cima, pela penúltima expressão, obtemos que:

$$\begin{aligned} r_s &= r_{s-2} - q_s r_{s-1} \\ r_{s-1} &= r_{s-3} - q_{s-1} r_{s-2} \\ &\vdots \\ r_3 &= r_1 - q_3 r_2 \\ r_2 &= g - q_2 r_1 \\ r_1 &= f - q_1 g. \end{aligned}$$

Por simplicidade, suponhamos que $s = 3$. Combinamos as expressões acima da seguinte maneira:

$$\begin{aligned} r_3 &= r_1 - q_3 r_2 = r_1 - q_3(g - q_2 r_1) \\ &= r_1(1 + q_2 q_3) - q_3 g = (f - q_1 g)(1 + q_2 q_3) - q_3 g \\ &= f(1 + q_2 q_3) + g(-q_1 - q_1 q_2 q_3 - q_3). \end{aligned}$$

Portanto, $a = 1 + q_2 q_3$ e $b = -q_1 - q_1 q_2 q_3 - q_3$. No caso geral, procedemos de foram análoga com esta série de substituições para encontrar os polinômios $a$ e $b$.

**Exemplo 1.3.4.** Sejam $f(x) = 2x^{10} + x^7 + x^2 + 1$ e $g(x) = x^7 + 1$ em $\mathbb{Z}_3[x]$. Vamos escrever o $\mathrm{mdc}(f, g)$ como combinação linear de $f$ e $g$. Pelo algoritmo euclideano, temos as seguintes expressões:

$$\begin{aligned} 2x^{10} + x^7 + x^2 + 1 &= (2x^3 + 1)(x^7 + 1) + (x^3 + x^2) \\ x^7 + 1 &= (x^4 + 2x^3 + x^2 + 2x + 1)(x^3 + x^2) + (2x^2 + 1) \\ x^3 + x^2 &= (2x + 2)(2x^2 + 1) + (x + 1) \\ 2x^2 + 1 &= (2x + 1)(x + 1) + 0. \end{aligned}$$

Logo $\mathrm{mdc}(f, g) = x + 1$. Além disso, temos

$$\begin{aligned} & x + 1 \\ = \; & (x^3 + x^2) - (2x + 2)(2x^2 + 1) \\ = \; & (x^3 + x^2) - (2x + 2)((x^7 + 1) - (x^4 + 2x^3 + x^2 + 2x + 1)(x^3 + x^2)) \\ = \; & (2x^5)(x^3 + x^2) + (x + 1)(x^7 + 1) \\ = \; & (2x^5)((2x^{10} + x^7 + x^2 + 1) - (2x^3 + 1)(x^7 + 1)) + (x + 1)(x^7 + 1) \\ = \; & 2x^5(2x^{10} + x^7 + x^2 + 1) + (2x^8 + x^5 + x + 1)(x^7 + 1). \end{aligned}$$

Assim, obtemos que $\mathrm{mdc}(f, g) = (2x^5)f + (2x^8 + x^5 + x + 1)g$.

Consideramos umas das propriedades mais importantes de $F[x]$: a *fatoração única de polinômios.*

**Teorema 1.3.5.** [Fatoração única de polinômios] *Seja $F$ um corpo. Qualquer polinômio $f \in F[x]$ de grau positivo pode ser escrito como*

$$f = a f_1^{e_1} f_2^{e_2} \dots f_k^{e_k},$$

*onde $a \in F$, $e_1, e_2, \dots, e_k \in \mathbb{N}$ e $f_1, f_2, \dots, f_k$ são polinômios mônicos irredutíveis sobre $F$. Além disso, esta fatoração é única a menos da ordem dos fatores.*

A prova deste teorema, que será omitida neste livro, não é construtiva, uma vez que não fornece um algoritmo para fatorar polinômios. Veremos alguns métodos para fatorar polinômios sobre um corpo finito no Capítulo 3.

O próximo resultado mostra que qualquer ideal em $F[x]$ é principal. A fatoração única de polinômios e a existência do máximo divisor comum são conseqüências dele.

**Teorema 1.3.6.** *Seja $F$ um corpo. Então $F[x]$ é um domínio de ideais principais. Mais precisamente, para qualquer ideal não nulo $I$ de $F[x]$, existe um único polinômio mônico $f \in F[x]$ tal que $I = (f)$.*

*Demonstração.* Pelo Exemplo 1.1.4 (3), temos que $F[x]$ é um domínio de integridade. Seja $I$ um ideal não nulo de $F[x]$. Seja $f$ um polinômio mônico não constante de menor grau em $I$. Claramente $(f) \subseteq I$. Para a outra inclusão, seja $g \in I$. Pelo algoritmo da divisão (Teorema 1.3.1), escrevemos $g = fh + r$ onde $h, r \in F[x]$ e $\text{grau}(r) < \text{grau}(f)$. Como $r = g - fh$ está em $I$, a minimalidade do grau de $f$ implica que $r = 0$. Portanto $g = fh$ e assim $I = (f)$.

Para a unicidade, suponhamos que $I = (\hat{f})$, onde $\hat{f}$ é um polinômio mônico sobre $F$. Então $f \mid \hat{f}$ e $\hat{f} \mid f$, mas como $f$ e $\hat{f}$ são ambos mônicos, obtemos $f = \hat{f}$. □

A seguir, discutiremos os quocientes $F[x]/(f)$ do anel dos polinômios $F[x]$ pelo ideal gerado pelo polinômio $f$. Primeiro, combinando Teoremas 1.1.23(4) e 1.3.6, obtemos o seguinte resultado.

**Teorema 1.3.7.** *Sejam $F$ um corpo e $f$ um polinômio mônico de grau positivo sobre $F$. Então $F[x]/(f)$ é um corpo se e somente se $f$ é irredutível sobre $F$.*

A fim de descrevermos os elementos em $F[x]/(f)$, vamos explorar o seguinte exemplo.

**Exemplo 1.3.8.** Mostraremos que

$$F = \mathbb{R}[x]/(x^2+1) = \{r_0 + r_1\alpha \colon r_0, r_1 \in \mathbb{R} \text{ e } \alpha^2 + 1 = 0\}.$$

Primeiro observamos que $x^2 + 1$ é irredutível sobre $\mathbb{R}$, logo $F$ é um corpo. Um elemento típico em $F$ é $\bar{f} = f + (x^2+1)$ onde $f \in \mathbb{R}[x]$. Se

grau$(f) > 2$, pelo algoritmo da divisão, temos que $f = (x^2+1)h + r$ onde $r = r_0 + r_1 x \in \mathbb{R}[x]$. Assim $\bar{f} = r + (x^2+1)$. Como $x^2+1 \in (x^2+1)$, segue que $\overline{x^2+1} = \bar{0}$ em $F$. Logo $\bar{f} = r_0 + r_1\alpha$, onde $\alpha = \bar{x}$ e portanto $\alpha^2 + 1 = 0$. As operações em $F$ satisfazem

$$\overline{f+g} = \bar{f} + \bar{g} \qquad \text{e} \qquad \overline{fg} = \bar{f}\bar{g}.$$

Mais explicitamente,

$$(a+b\alpha) + (c+d\alpha) = (a+c) + (b+d)\alpha \qquad \text{e}$$
$$(a+b\alpha)(c+d\alpha) = (ac-bd) + (ad+bc)\alpha.$$

Como $F$ é um corpo, a pergunta natural é: como podemos calcular o inverso de $\bar{f}$ se $r_0$ e $r_1$ não são ambos nulos? Acontece que $\text{mdc}(\bar{f}, \alpha^2+1) = 1$. Pelo algoritmo euclideano estendido, escrevemos $\bar{f}\bar{g} + (\alpha^2+1)\bar{h} = 1$ e assim $\bar{f}^{-1} = \bar{g}$. Também, vê-se facilmente que $(a+b\alpha)^{-1} = (a-b\alpha)/(a^2+b^2)$

No exemplo acima, substituindo $\alpha$ por $i$, obtemos que $F = \mathbb{R}(i) = \mathbb{C}$. Em geral, temos o seguinte resultado.

**Teorema 1.3.9.** *Sejam $F$ um corpo e $f$ um polinômio mônico de grau positivo $n$ sobre $F$. Então o anel quociente $F[x]/(f)$ pode ser descrito como*

$$\{a_0 + a_1\alpha + \cdots + a_{n-1}\alpha^{n-1} : a_0, a_1, \ldots, a_{n-1} \in F \text{ e } f(\alpha) = 0\}.$$

Veremos um tipo especial de extensão de corpos.

**Definição 1.3.10.** Seja $K$ uma extensão do corpo $F$. Um elemento $\theta \in K$ é *algébrico sobre* $F$, se existe um polinômio $f \in F[x]$ tal que $f(\theta) = 0$. Se todos os elementos de $K$ são algébricos sobre $F$, dizemos que a extensão $K$ de $F$ é *algébrica*.

Suponhamos que $\theta$ seja algébrico sobre $F$. Seja $M$ um polinômio mônico de menor grau dentre todos os polinômios $f$ em $F[x]$ tais que $f(\theta) = 0$. Afirmamos que $M$ divide qualquer polinômio $f$ com tal propriedade. De fato, escrevendo $f = qM + r$ onde grau$(r) <$ grau$(M)$, obtemos que $r(\theta) = 0$. Pela minimalidade do grau de $M$, concluímos que $r = 0$. Já veremos que $M$ é na verdade único. Esta observação nos motiva à seguinte definição.

**Definição 1.3.11.** Seja $\theta \in K$ um elemento algébrico sobre $F$. O único polinômio mônico $M \in F[x]$ que gera o ideal

$$I = \{f \in F[x] \colon f(\theta) = 0\}$$

é chamado o *polinômio minimal* de $\theta$ sobre $F$.

Reunimos agora o comentário acima e suas conseqüências.

**Teorema 1.3.12.** *Sejam $K$ uma extensão do corpo $F$ e $\theta \in K$ um elemento algébrico sobre $F$. O polinômio minimal $M$ de $\theta$ sobre $F$ possui as seguintes propriedades.*

1. *$M$ é irredutível sobre $F$.*
2. *Seja $f$ um polinômio em $F[x]$. Então $f(\theta) = 0$ se e somente se $M$ divide $f$.*
3. *$M$ é o único polinômio mônico sobre $F$ de menor grau tal que $M(\theta) = 0$.*
4. *$M$ é o único polinômio mônico irredutível sobre $F$ satisfazendo $M(\theta) = 0$.*
5. *Temos que* $\operatorname{grau}(M)$ *divide* $[K : F]$. *Em particular,* $\operatorname{grau}(M) \leq [K : F]$. *Além disso,* $\operatorname{grau}(M) = [K : F]$ *se e somente se* $K = F(\theta)$.

*Demonstração.* Vamos provar (1). Definimos um homomorfismo de anéis $\psi \colon F[x] \to K$ por $\psi(f) = f(\theta)$. Temos que $\ker \psi = (M)$. Como $F[x]/(M) \cong \psi(F[x])$ é um domínio de integridade, o ideal $(M)$ é primo e assim é maximal, pelo Teorema 1.1.23. Uma outra aplicação do mesmo teorema dá que $M$ é irredutível sobre $F$.

Segue diretamente da definição de polinômio minimal e de (1) que (2)–(4) se verificam.

Para provar (5), temos que $F[x]/(M)$ é um corpo e é isomorfo a $\psi(F[x])$, que então deve ser $F(\theta)$. Teorema 1.3.9 dá que $[F(\theta) : F] = \operatorname{grau}(M)$. Portanto, temos que $\operatorname{grau}(M)$ divide $[K : F]$, pela fórmula dada pelo Teorema 1.2.6:

$$[K : F] = [K : F(\theta)][F(\theta) : F].$$

Além disso, grau$(M) = [K : F]$ se e somente se $[K : F(\theta)] = 1$. Em outras palavras, $K = F(\theta)$. □

Polinômios minimais desempenham um papel fundamental na teoria de códigos. Na Seção 5.2, veremos a relação entre este tipo de polinômios e os códigos BCH. A Seção 3.5 é dedicada a polinômios minimais sobre corpos finitos.

*Corpos de decomposição* são usados para descrever corpos finitos, como veremos na próxima seção.

**Definição 1.3.13.** Sejam $F$ e $K$ corpos, onde $K$ é uma extensão de $F$. O corpo $K$ é um *corpo de decomposição* do polinômio $f \in F[x]$, se $f$ é um produto de fatores lineares em $K[x]$ e se $f$ não é um produto de fatores lineares sobre qualquer subcorpo próprio de $K$ contendo $F$.

Em outras palavras, o corpo de decomposição de um polinômio $f$ sobre $F$ é o menor corpo que contém $F$ e as raízes de $f$. Assim, se $\theta_1, \ldots, \theta_n$ são as raízes de $f$ numa extensão de $F$, então $F(\theta_1, \ldots, \theta_n)$ é um corpo de decomposição de $f$. Na verdade, existe apenas um único corpo de decomposição de um polinômio, a menos de isomorfismos.

**Teorema 1.3.14.** *Sejam $F$ um corpo e $f$ um polinômio sobre $F$. Existe uma extensão $K$ de $F$ que é um corpo de decomposição de $f$. Além disso, quaisquer dois corpos de decomposição de $f$ sobre $F$ são isomorfos.*

**Exemplo 1.3.15.**

1. O corpo de decomposição de $x^2 - 2$ sobre $\mathbb{Q}$ é $\mathbb{Q}(\sqrt{2})$.
2. O corpo de decomposição de $x^4 - 5x^2 + 6$ sobre $\mathbb{Q}$ é $\mathbb{Q}(\sqrt{2}, \sqrt{3})$.

## 1.4 Corpos finitos

Seja $F$ um corpo com $q$ elementos, onde $q < \infty$. Neste caso, dizemos que $F$ é um *corpo finito* e $q$ é a *ordem* de $F$. Um exemplo importante de um corpo finito é o seguinte.

**Exemplo 1.4.1.** Sejam $p$ um primo, $\mathbb{F}_p$ o conjunto $\{0, 1, \ldots, p-1\}$ e $\phi : \mathbb{Z}/p\mathbb{Z} \to \mathbb{F}_p$ a aplicação bijetora definida por $\phi(a + p\mathbb{Z}) = a$. Então $\mathbb{F}_p$ com a estrutura de corpo induzida por $\mathbb{Z}/p\mathbb{Z}$ é um corpo finito de ordem $p$, que também é um corpo primo pelo Exemplo 1.2.3.

O próximo resultado mostra que corpos finitos têm característica positiva.

**Corolário 1.4.2.** *A característica de qualquer corpo finito é prima.*

*Demonstração.* Seja $F$ um corpo finito com identidade $1_F$. Existem inteiros $m$ e $n$ com $1 \leq m < n$ e $n \cdot 1_F = m \cdot 1_F$, já que $F$ é finito. Portanto $(n-m) \cdot 1_F = 0$ e logo $F$ tem característica positiva. Pelo Teorema 1.1.12, a característica de $F$ é prima. □

Agora vamos mostrar que a ordem de um corpo finito é uma potência da característica prima.

**Teorema 1.4.3.** *Sejam $K$ um corpo finito de ordem $q$ e $F$ uma extensão finita de $K$ de grau $n$. Então, a ordem de $F$ é $q^n$. Em particular, se $F$ é um corpo finito de ordem $q$ e característica $p$, então $q = p^n$ onde $n$ é o grau da extensão de $F$ sobre $\mathbb{F}_p$.*

*Demonstração.* Seja $\{\beta_1, \beta_2, \ldots, \beta_n\}$ uma base para o espaço vetorial $F$ sobre $K$. Qualquer elemento em $F$ tem uma expressão única da forma

$$a_1\beta_1 + a_2\beta_2 + \cdots + a_n\beta_n$$

com $a_1, a_2, \ldots, a_n \in K$. Há $q$ valores possíveis para cada $a_i$, logo o número total de elementos em $F$ é $q^n$. Pelo Teorema 1.2.4, o corpo $F$ contém o corpo primo $\mathbb{F}_p$ e assim concluímos esta prova. □

Há duas perguntas naturais que surgem deste resultado. Primeiro, existe um corpo de ordem $p^n$ para cada primo $p$ e cada inteiro positivo $n$? Se sim, ele é único? A reposta a ambas as perguntas é sim, sendo que para a segunda pergunta consideramos a unicidade a menos de isomorfismos. Um método para construir um corpo com $p^n$ elementos é dado pelo Teorema 1.3.7, mas o problema é que não temos nenhuma garantia de que exista um polinômio irredutível de grau $n$ sobre $\mathbb{F}_p$, para qualquer $n$. A seguir, apresentaremos uma série de lemas preliminares, alguns dos quais serão usados na prova da existência e da unicidade de corpos finitos.

**Lema 1.4.4.** *Num corpo finito $F$ de ordem $q$, qualquer $a \in F$ satisfaz $a^q = a$.*

*Demonstração.* Isto é trivial se $a = 0$. Caso contrário, como $F^* = F \setminus \{0\}$ é um grupo multiplicativo de ordem $q-1$, segue que $a^{q-1} = 1$ para todo $a \neq 0$. □

Usando este lema, temos o seguinte resultado.

**Lema 1.4.5.** *Seja $F$ um corpo finito de ordem $q$. Se $r$ e $s$ são inteiros tais que $r \equiv s \pmod{q-1}$, então $a^r = a^s$ para todo $a \in F$.*

*Demonstração.* Isto é trivial se $a = 0$. Suponhamos que $a \neq 0$ e $r - s = \ell(q-1)$ para algum inteiro $\ell$. Pelo Lema 1.4.4, temos que $a^r = a^{s+\ell(q-1)} = a^s(a^{q-1})^\ell = a^s \cdot 1^\ell = a^s$. □

Outra conseqüência do Lema 1.4.4 é que o polinômio $x^q - x$ é um produto de fatores lineares num corpo de ordem $q$.

**Lema 1.4.6.** *Seja $F$ um corpo de ordem $q$ e característica $p$. Então o polinômio $x^q - x$ fatora-se em $F[x]$ como*

$$x^q - x = \prod_{a \in F} (x - a).$$

*Além disso, $F$ é um corpo de decomposição de $x^q - x$ sobre $\mathbb{F}_p$.*

*Demonstração.* O polinômio $x^q - x$ tem no máximo $q$ raízes em $F$. Pelo Lema 1.4.4, qualquer $a \in F$ é uma raiz de $x^q - x$ e assim $x^q - x$ fatora-se sobre $F$, mas isto não acontece sobre qualquer outro subcorpo de $F$. □

O próximo lema é muito útil na aritmética de corpos finitos. Ele implica que, num corpo finito de característica $p$, a aplicação $a \mapsto a^p$ seja um automorfismo.

**Lema 1.4.7.** *Seja $F$ um corpo de característica $p$. Então, para qualquer inteiro $n \geq 0$, temos que*

$$(a + b)^{p^n} = a^{p^n} + b^{p^n}.$$

*Demonstração.* Usamos indução em $n$ para mostrar que $(a+b)^{p^n} = a^{p^n} + b^{p^n}$. Para $n = 1$, observamos que todo coeficiente binomial $\binom{p}{i}$ com $0 < i < p$ na expansão de $(a+b)^p$ é zero, já que

$$\binom{p}{i} = \frac{p(p-1)\dots(p-i+1)}{i!} \equiv 0 \pmod{p}.$$

Segue da hipótese de indução que

$$(a+b)^{p^{n+1}} = ((a+b)^{p^n})^p = (a^{p^n} + b^{p^n})^p = a^{p^{n+1}} + b^{p^{n+1}}.$$

Portanto $(a+b)^{p^n} = a^{p^n} + b^{p^n}$ para todo inteiro $n \geq 0$. □

O seguinte corolário será usado mais tarde.

**Corolário 1.4.8.** *Seja $F$ um corpo de característica $p$. A aplicação $\theta : F \to F$ dada por $\theta(a) = a^p$ é um isomorfismo de anéis tal que $\theta(a) = a$ para todo $a \in \mathbb{F}_p$.*

*Demonstração.* Lema 1.4.7 mostra que $\theta(a+b) = \theta(a) + \theta(b)$. Obviamente, $\theta(ab) = (ab)^p = a^p b^p = \theta(a)\theta(b)$ e $\theta(1) = 1^p = 1$. Logo $\theta$ é um homomorfismo. Se $a \in \ker\theta$, então $0 = \theta(a) = a^p$ e portanto $a = 0$. Sendo assim, concluímos que $\theta$ é injetora e logo é bijetora. Como $F$ tem característica $p$, temos que $\mathbb{F}_p \subseteq F$. Pelo Lema 1.4.4, temos que $\theta(a) = a^p = a$ para qualquer $a \in \mathbb{F}_p$. □

Agora estamos preparados para demonstrar a existência e a unicidade de corpos finitos.

**Teorema 1.4.9.** [Existência e unicidade de corpos finitos] *Para qualquer primo $p$ e qualquer inteiro positivo $n$, existe um corpo finito com $p^n$ elementos. Além disso, qualquer corpo finito com $p^n$ elementos é isomorfo ao corpo de decomposição de $x^{p^n} - x$ sobre $\mathbb{F}_p$.*

*Demonstração.* Suponhamos $q = p^n$. Seja $F$ o corpo de decomposição de $x^q - x$ sobre $\mathbb{F}_p$. Todas as raízes de $x^q - x$ em $F$ são distintas. De fato, se um polinômio $f$ tem fatores repetidos, então $\mathrm{mdc}(f, f') \neq 1$ (veremos isso mais detalhadamente na Seção 3.6). No nosso caso,

$$\mathrm{mdc}(x^q - x, (x^q - x)') = \mathrm{mdc}(x^q - x, qx^{q-1} - 1) = \mathrm{mdc}(x^q - x, -1) = 1.$$

Assim $S = \{a \in F : a^q = a\}$ tem $q$ elementos. A seguir, mostraremos que $S$ é um corpo. Vamos verificar as condições do Teorema 1.2.2.

Claramente $S$ é um subconjunto de $F$ que contém 0 e 1. Além disso, usando o Lema 1.4.7, se $a, b \in S$ então

$$(a-b)^q = a^q + (-b)^q = a^q - b^q = a - b$$

implica que $a - b \in S$. Para $b \neq 0$, temos

$$(ab^{-1})^q = a^q (b^q)^{-1} = ab^{-1},$$

isto é, $ab^{-1} \in S$. Logo $S$ é um corpo e contém todas as raízes de $x^q - x$, ou seja, $x^q - x$ fatora-se completamente em $S$. Portanto, $S = F$ e $F$ é um corpo finito com $q$ elementos.

Para mostrar a unicidade, seja $F$ um corpo finito com $q = p^n$ elementos. Então $F$ tem característica $p$ e contém $\mathbb{F}_p$ como um subcorpo primo. Lema 1.4.6 implica que $F$ seja um corpo de decomposição de $x^q - x$ sobre $\mathbb{F}_p$. O Teorema 1.3.14 completa a prova. □

Se $p$ é primo e $q = p^n$, então o corpo de ordem $q$ é denotado por $\mathbb{F}_q$. A seguir, apresentamos uma conseqüência do Teorema 1.3.7.

**Teorema 1.4.10.** *Seja $q = p^n$. Se $f$ é um polinômio irredutível sobre $\mathbb{F}_p$ de grau $n$ então $\mathbb{F}_q \cong \mathbb{F}_p[x]/(f)$.*

Este isomorfismo fornece uma maneira de representar os elementos num corpo finito como veremos na Seção 2.1. O seguinte resultado também será usado mais tarde.

**Teorema 1.4.11.** *Seja $f \in \mathbb{F}_q[x]$ irredutível sobre $\mathbb{F}_q$. Então existe uma extensão simples de $\mathbb{F}_q$ sendo definida por uma raiz de $f$. Além disso, se $\theta$ é uma raiz de $f$, então $\mathbb{F}_q(\theta) \cong \mathbb{F}_q[x]/(f)$.*

Os subcorpos de um corpo finito podem ser precisamente descritos. Para isso, precisamos de um lema elementar.

**Lema 1.4.12.** *Sejam $R$ um anel e $r \in R$. Suponhamos que $m$ e $n$ sejam inteiros positivos tais que $m \mid n$. Então $(r^m - 1) \mid (r^n - 1)$.*

*Demonstração.* Se $n = \ell m$, então temos que

$$r^n - 1 = (r^m)^\ell - 1 = (r^m - 1)\left((r^m)^{\ell-1} + (r^m)^{\ell-2} + \cdots + 1\right),$$

estabelecendo assim o lema. □

**Teorema 1.4.13.** *Seja $\mathbb{F}_q$ um corpo finito com $q = p^n$ elementos. Então todo subcorpo de $\mathbb{F}_q$ tem ordem $p^m$, onde $m$ é um divisor positivo de $n$. Reciprocamente, se $m$ é um divisor positivo de $n$, então existe exatamente um subcorpo de $\mathbb{F}_q$ com $p^m$ elementos.*

*Demonstração.* Se $q = p^n$, então qualquer subcorpo $F$ de $\mathbb{F}_q$ tem ordem $p^m$ com $0 < m \leq n$. Se $[\mathbb{F}_q : F] = \ell$, então $p^n = (p^m)^\ell = p^{m\ell}$, pelo Teorema 1.4.3, e assim $m \mid n$.

Reciprocamente, se $m$ é um divisor positivo de $n$, pelo Lema 1.4.12, obtemos que $(p^m - 1) \mid (p^n - 1)$ e assim

$$(x^{p^m-1} - 1) \mid (x^{p^n-1} - 1),$$

por uma outra aplicação do Lema 1.4.12, desta vez no anel $\mathbb{F}_p[x]$. Deduzimos que toda raiz de $x^{p^m} - x$ é uma raiz de $x^{p^n} - x = x^q - x$ e portanto toda raiz de $x^{p^m} - x$ pertence a $\mathbb{F}_q$. Segue que $\mathbb{F}_q$ deve conter um corpo de decomposição de $x^{p^m} - x$ sobre $\mathbb{F}_q$. Teorema 1.4.9 implica que tal corpo de decomposição contenha $p^m$ elementos.

Portanto, existe exatamente um subcorpo com $p^m$ elementos, caracterizado pelas raízes do polinômio $x^{p^m} - x$ em $\mathbb{F}_q$. □

**Exemplo 1.4.14.**

1. Como os divisores positivos de 10 são 1, 2, 5 e 10, os subcorpos de $\mathbb{F}_{2^{10}}$ são $\mathbb{F}_2$, $\mathbb{F}_{2^2}$, $\mathbb{F}_{2^5}$ e $\mathbb{F}_{2^{10}}$.

2. $\mathbb{F}_{3^{18}}$ tem cinco subcorpos próprios: $\mathbb{F}_3$, $\mathbb{F}_{3^2}$, $\mathbb{F}_{3^3}$, $\mathbb{F}_{3^6}$ e $\mathbb{F}_{3^9}$.

Os exemplos acima estão ilustrados na Figura 1.1.

O nosso próximo passo é definir os *conjugados* de um elemento num corpo finito.

**Definição 1.4.15.** Sejam $\mathbb{F}_{q^m}$ uma extensão de $\mathbb{F}_q$ e $\alpha$ um elemento em $\mathbb{F}_{q^m}$. Os elementos $\alpha, \alpha^q, \alpha^{q^2}, \ldots, \alpha^{q^{m-1}}$ são chamados os *conjugados* de $\alpha$ com respeito a $\mathbb{F}_q$.

A soma dos conjugados de $\alpha$ produz uma função especial muito usada na prática. Veremos algumas dessas aplicações mais adiante.

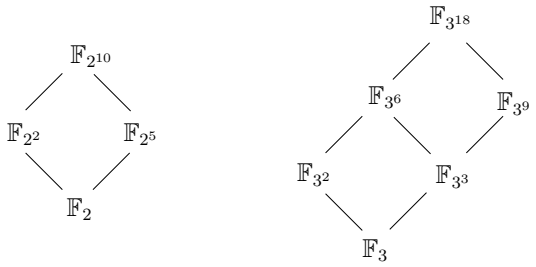



Figura 1.1: Subcorpos de $\mathbb{F}_{2^{10}}$ e $\mathbb{F}_{3^{18}}$.

**Definição 1.4.16.** Para cada $\alpha \in \mathbb{F}_{q^m}$, o *traço* de $\alpha$ sobre $\mathbb{F}_q$ é definido por

$$\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(\alpha) = \alpha + \alpha^q + \alpha^{q^2} + \cdots + \alpha^{q^{m-1}}.$$

A próxima proposição contém algumas propriedades do traço como conseqüências dos Lemas 1.4.4 e 1.4.7.

**Proposição 1.4.17.** *Sejam $\mathbb{F}_{q^m}$ uma extensão de $\mathbb{F}_q$, $\alpha, \beta \in \mathbb{F}_{q^m}$ e $a, b \in \mathbb{F}_q$. Então:*

1. $\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(\alpha) \in \mathbb{F}_q$;
2. $\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(a\alpha + b\beta) = a\,\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(\alpha) + b\,\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(\beta)$;
3. $\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}$ *é uma transformação linear de* $\mathbb{F}_{q^m}$ *em* $\mathbb{F}_q$, *onde* $\mathbb{F}_{q^m}$ *e* $\mathbb{F}_q$ *são vistos como espaços vetoriais sobre* $\mathbb{F}_q$;
4. $\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(a) = ma$;
5. $\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(\alpha^q) = \mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(\alpha)$.

# Capítulo 2

# Aritmética em corpos finitos

Neste capítulo, discutiremos brevemente as complexidades computacionais das seguintes operações aritméticas em corpos finitos: adição, multiplicação, exponenciação e cálculo de inversos. As idéias de alguns métodos serão apresentadas.

Corpos finitos oferecem a flexibilidade de que seus elementos podem ser representados de maneiras diferentes. Dependendo da operação aritmética, uma representação pode ser mais conveniente do que outra em termos de custos computacionais. Veremos três formas de representar um elemento num corpo finito. As operações aritméticas serão discutidas de acordo com cada tipo de representação: através de um polinômio, um elemento primitivo e um elemento normal, nas Seções 2.1, 2.2 e 2.3, respectivamente.

Independentemente da representação em questão, a aritmética em $\mathbb{F}_q$ dependerá da aritmética em $\mathbb{F}_p$, onde $p$ é a característica de $\mathbb{F}_q$. Lembramos que $\mathbb{F}_p \cong \mathbb{Z}_p$, onde a aritmética é feita módulo $p$. A procura por implementações eficientes de operações módulo $p$ é intensa e de grande importância. No entanto, não as discutiremos neste livro, pois elas nos desviariam do nosso objetivo, que é mostrar a aritmética ao nível das extensões de corpos finitos, em geral. A complexidade de uma operação em $\mathbb{F}_q$ pode ser dada em termos do número de

operações em $\mathbb{F}_p$. Este enfoque é natural na busca por métodos eficientes de operações em $\mathbb{F}_q$, já que a comparação destes métodos fica assim independente dos métodos usados para a aritmética do corpo primo.

Na prática, o primo $p$ pode ser bastante grande, chegando a exceder o tamanho de uma palavra do computador. Uma solução utilizada para este problema envolve operações com inteiros longos. Para maiores detalhes, veja o livro de Knuth [83] e o pacote NTL[1] de Shoup [125], por exemplo.

Neste livro, estaremos considerando o caso mais geral de contar as operações em $\mathbb{F}_{q^n}$ em função das operações em $\mathbb{F}_q$. Em particular, $q$ pode ser primo, que é o caso considerado acima.

Para a análise assintótica do custo das operações aritméticas, usaremos a seguinte definição. Sejam $f$ e $g$ funções de $\mathbb{N}$ (ou de $\mathbb{R}^+$) em $\mathbb{R}$. Dizemos que $f$ é $O(g)$ se existem constantes $C \in \mathbb{R}^+$ e $n_0 \in \mathbb{N}$ tais que $|f(x)| \leq C|g(x)|$ para todo $x \geq n_0$. Dizemos que $f$ é $o(g)$, se $f$ é $O(g)$ e $g$ não é $O(f)$.

## 2.1 Representação polinomial

Pelo Teorema 1.4.10, temos que $\mathbb{F}_{q^n} \cong \mathbb{F}_q[x]/(f)$, onde $f$ é um polinômio irredutível de grau $n$ sobre $\mathbb{F}_q$. Assim, pelo Teorema 1.3.9, temos que qualquer elemento em $\mathbb{F}_{q^n}$ pode ser representado por um polinômio em $\mathbb{F}_q[x]$ de grau menor que $n$ e que o próprio $f$ é o zero do corpo.

Ao operarmos com dois polinômios, se o grau do polinômio resultante ultrapassar $n$, devemos fazer uma redução para que tenhamos uma representação em termos de um polinômio de grau menor que $n$. Para isso, seja $r$ o resto da divisão de $g$ por $f$. Então $r$ tem grau menor que $n$ e os polinômios $r$ e $g$ representam o mesmo elemento em $\mathbb{F}_{q^n}$. Neste caso, escrevemos $g \equiv r \pmod{f}$.

**Exemplo 2.1.1.** Observamos que $f(x) = x^2 + x + 1$ tem grau 2 e não possui raízes em $\mathbb{F}_2$. Logo, pelo Exemplo 1.1.9 (7), o polinômio $f$ é irredutível sobre $\mathbb{F}_2$. Temos assim que $\mathbb{F}_4$ e $\mathbb{F}_2[x]/(f)$ são isomorfos. Os elementos de $\mathbb{F}_4$, representados em termos de polinômios, são

[1] Number Theory Library.

0, 1, $x$ e $x+1$. A multiplicação de $x$ por $x+1$, por exemplo, é $x(x+1) = x^2 + x \equiv 1 \pmod{f}$. As tabelas completas de adição e de multiplicação para $\mathbb{F}_4$ são:

| $+$ | $0$ | $1$ | $x$ | $x+1$ |
|---|---|---|---|---|
| $0$ | $0$ | $1$ | $x$ | $x+1$ |
| $1$ | $1$ | $0$ | $x+1$ | $x$ |
| $x$ | $x$ | $x+1$ | $0$ | $1$ |
| $x+1$ | $x+1$ | $x$ | $1$ | $0$ |

| $\cdot$ | $0$ | $1$ | $x$ | $x+1$ |
|---|---|---|---|---|
| $0$ | $0$ | $0$ | $0$ | $0$ |
| $1$ | $0$ | $1$ | $x$ | $x+1$ |
| $x$ | $0$ | $x$ | $x+1$ | $1$ |
| $x+1$ | $0$ | $x+1$ | $1$ | $x$ |

**Exemplo 2.1.2.** A Tabela 2.1 mostra a adição e a multiplicação em $\mathbb{F}_8$ visto como $\mathbb{F}_2[x]/(x^3+x+1)$. O polinômio $x^3+x+1$ é irredutível sobre $\mathbb{F}_2$, pois tem grau 3 e não possui raízes em $\mathbb{F}_2$.

Em seguida, discutiremos as operações aritméticas em $\mathbb{F}_{q^n}$ através da representação polinomial sobre $\mathbb{F}_q$. Inicialmente, discutiremos a complexidade computacional destas operações usando métodos diretos. Em seguida, discutiremos as idéias de dois métodos específicos para a multiplicação num corpo finito: o algoritmo de Karatsuba e o algoritmo baseado na Transformada Discreta de Fourier (FFT, em inglês). No fim desta seção, apresentaremos o método da repetição de quadrados usado para a exponenciação.

| $+$ | $0$ | $1$ | $x$ | $x+1$ | $x^2$ | $x^2+1$ | $x^2+x$ | $x^2+x+1$ |
|---|---|---|---|---|---|---|---|---|
| $0$ | $0$ | $1$ | $x$ | $x+1$ | $x^2$ | $x^2+1$ | $x^2+x$ | $x^2+x+1$ |
| $1$ | $1$ | $0$ | $x+1$ | $x$ | $x^2+1$ | $x^2$ | $x^2+x+1$ | $x^2+x$ |
| $x$ | $x$ | $x+1$ | $0$ | $1$ | $x^2+x$ | $x^2+x+1$ | $x^2$ | $x^2+1$ |
| $x+1$ | $x+1$ | $x$ | $1$ | $0$ | $x^2+x+1$ | $x^2+x$ | $x^2+1$ | $x^2$ |
| $x^2$ | $x^2$ | $x^2+1$ | $x^2+x$ | $x^2+x+1$ | $0$ | $1$ | $x$ | $x+1$ |
| $x^2+1$ | $x^2+1$ | $x^2$ | $x^2+x+1$ | $x^2+x$ | $1$ | $0$ | $x+1$ | $x$ |
| $x^2+x$ | $x^2+x$ | $x^2+x+1$ | $x^2$ | $x^2+1$ | $x$ | $x+1$ | $0$ | $1$ |
| $x^2+x+1$ | $x^2+x+1$ | $x^2+x$ | $x^2+1$ | $x^2$ | $x+1$ | $x$ | $1$ | $0$ |

| $\cdot$ | $0$ | $1$ | $x$ | $x+1$ | $x^2$ | $x^2+1$ | $x^2+x$ | $x^2+x+1$ |
|---|---|---|---|---|---|---|---|---|
| $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ |
| $1$ | $0$ | $1$ | $x$ | $x+1$ | $x^2$ | $x^2+1$ | $x^2+x$ | $x^2+x+1$ |
| $x$ | $0$ | $x$ | $x^2$ | $x^2+x$ | $x+1$ | $1$ | $x^2+x+1$ | $x^2+1$ |
| $x+1$ | $0$ | $x+1$ | $x^2+x$ | $x^2+1$ | $x^2+x+1$ | $x^2$ | $1$ | $x$ |
| $x^2$ | $0$ | $x^2$ | $x+1$ | $x^2+x+1$ | $x^2+x$ | $x$ | $x^2+1$ | $1$ |
| $x^2+1$ | $0$ | $x^2+1$ | $1$ | $x^2$ | $x$ | $x^2+x+1$ | $x+1$ | $x^2+x$ |
| $x^2+x$ | $0$ | $x^2+x$ | $x^2+x+1$ | $1$ | $x^2+1$ | $x+1$ | $x$ | $x^2$ |
| $x^2+x+1$ | $0$ | $x^2+x+1$ | $x^2+1$ | $x$ | $1$ | $x^2+x$ | $x^2$ | $x+1$ |

Tabela 2.1: Tabelas de adição e de multiplicação de $\mathbb{F}_8$ visto como $\mathbb{F}_2[x]/(x^3+x+1)$.

### 2.1.1 Métodos diretos

Sejam $a$ e $b$ elementos de $\mathbb{F}_{q^n}$ representados pelos polinômios $a(x) = \sum_{i=0}^{\ell} a_i x^i$ e $b(x) = \sum_{j=0}^{m} b_j x^j$ em $\mathbb{F}_q[x]$, respectivamente, onde $\ell \geq m$. Suponhamos sem perda de generalidade que $\ell, m < n$. A operação mais básica entre polinômios é a adição:

$$a(x) + b(x) = \sum_{k=0}^{\ell} (a_k + b_k) x^k,$$

onde $b_{m+1} = \cdots = b_\ell = 0$. O custo de adicionar dois polinômios é $O(\ell)$ operações em $\mathbb{F}_q$.

Para a multiplicação, temos que $a(x)b(x) = \sum_{k=0}^{\ell+m} c_k x^k$, onde

$$c_k = \sum_{\substack{0 \leq i \leq \ell \\ 0 \leq j \leq m \\ k=i+j}} a_i b_j.$$

A questão aqui é como calcular cada coeficiente $c_k$. Se usarmos o método tradicional (também conhecido como *multiplicação clássica*), o custo de multiplicar dois polinômios de grau $\ell$ e $m$ com $\ell \geq m$ é $O(\ell^2)$ operações em $\mathbb{F}_q$. Veremos mais adiante que há métodos melhores que este.

A exponenciação é um caso particular da multiplicação. Os métodos clássicos podem tornar a exponenciação bastante cara, se a potência for muito grande. Na Seção 2.1.4, veremos uma maneira mais rápida baseada na repetição de quadrados.

A maneira mais conhecida de calcular inversos num corpo finito é através do algoritmo euclideano estendido. Suponhamos que $a$ é não nulo. Como $f$ é irredutível sobre $\mathbb{F}_q$ e $\text{grau}(a) < \text{grau}(f)$, temos que $\text{mdc}(a, f) = 1$. Pelo algoritmo euclideano estendido, existem polinômios $r, s$ em $\mathbb{F}_q[x]$ tais que $ar + fs = 1$, isto é, o inverso de $a$ em $\mathbb{F}_{q^n}$ é representado pelo polinômio $r$. O mdc de dois polinômios de grau no máximo $\ell$ em $\mathbb{F}_{q^n}[x]$ pode ser calculado com $O(\ell^2)$ operações em $\mathbb{F}_q$ usando aritmética clássica e o algoritmo euclideano. Portanto, o custo de calcular inversos também é $O(\ell^2)$.

Para finalizar esta seção, observamos que as complexidades apresentadas acima para as operações de adição e multiplicação no corpo

finito $\mathbb{F}_{q^n}$ equivalem às complexidades das respectivas operações no anel de polinômios $\mathbb{F}_q[x]$. Uma operação que não foi mencionada acima explicitamente é a divisão. Num corpo finito, a divisão é obtida pelo cálculo de um certo inverso seguido pela multiplicação, isto é, $a/b = ab^{-1}$. Para a divisão de polinômios, sejam $a, b \in \mathbb{F}_q[x]$, onde $b$ é um polinômio mônico não nulo. Se usarmos a divisão clássica com multiplicação clássica, quando uma multiplicação polinomial é necessária, o custo da divisão é $O(m(\ell - m))$ ou $O(\ell^2)$ operações em $\mathbb{F}_q$. Esta informação será usada no próximo capítulo.

### 2.1.2 Algoritmo de Karatsuba

Por simplicidade, suponhamos que os polinômios a serem multiplicados tenham grau $\ell - 1$ onde $\ell = 2^k$. Se este não é o caso, podemos sempre preencher o polinômio com zeros até a próxima potência de dois menos um. Há também outras maneiras eficientes de se lidar com o caso em que $\ell$ não é uma potência de dois, mas que não serão discutidas aqui. Para $m = \ell/2$, escrevemos

$$a(x) = A_1(x)x^m + A_0(x) \quad \text{e} \quad b(x) = B_1(x)x^m + B_0(x),$$

onde

$$\begin{aligned} A_1(x) &= a_{\ell-1}x^{m-1} + \cdots + a_m, \\ A_0(x) &= a_{m-1}x^{m-1} + \cdots + a_0, \\ B_1(x) &= b_{\ell-1}x^{m-1} + \cdots + b_m, \\ B_0(x) &= b_{m-1}x^{m-1} + \cdots + b_0. \end{aligned}$$

Assim, temos que

$$\begin{aligned} a(x)b(x) &= A_1(x)B_1(x)x^\ell \\ &\quad +(A_0(x)B_1(x) + A_1(x)B_0(x))x^m \\ &\quad +A_0(x)B_0(x). \end{aligned}$$

Seja $T(\ell)$ o pior custo de multiplicar dois polinômios de grau $\ell - 1$. Este custo é dado pela recorrência

$$\begin{aligned} T(1) &= 1, \\ T(\ell) &= 4T\left(\frac{\ell}{2}\right) + C_1\ell, \quad \ell = 2^k > 1, \end{aligned}$$

onde $C_1\ell$ é o custo de calcular as adições de polinômios. Resolvendo esta recorrência, obtemos que $T(\ell)$ é $O(\ell^2)$ e assim não melhoramos o algoritmo clássico.

Karatsuba [82] propôs calcular o produto de dois polinômios de grau $\ell - 1$, calculando três produtos de polinômios de grau $(\ell/2) - 1$, a saber, $A_0B_0$, $A_1B_1$ e $(A_0 + A_1)(B_0 + B_1)$, pois

$$A_0B_1 + A_1B_0 = (A_0 + A_1)(B_0 + B_1) - A_0B_0 - A_1B_1.$$

A recorrência que dá o custo deste algoritmo é

$$\begin{aligned} T(1) &= 1, \\ T(\ell) &= 3T\left(\frac{\ell}{2}\right) + C_2\ell, \quad \ell = 2^k > 1, \end{aligned}$$

onde $C_2\ell$ é o custo de calcular adições de polinômios. Resolvendo esta recorrência (exercício), obtemos que $T(\ell)$ é $O(\ell^{\log_2 3})$ ou $O(\ell^{1.59})$.

Implementações do método de Karatsuba mostram que não somente ele é rápido assintoticamente, mas também é prático: para quase todo $\ell$, este algoritmo é melhor do que a multiplicação clássica.

Observamos que o conhecido método de Strassen para multiplicação de matrizes é baseado em idéias similares às do método de Karatsuba.

### 2.1.3 Mutiplicação baseada no FFT

Um método de multiplicação mais rápido que o de Karatsuba, pelo menos assintoticamente, será apresentado nesta seção. A idéia chave é baseada na interpolação polinomial, que é um processo que permite obter um polinômio conhecendo apenas seus valores em certos pontos. Em outras palavras, dados $r_0, \ldots, r_\ell$ e $s_0, \ldots, s_\ell$ em $\mathbb{F}_{q^n}$, onde $r_i \neq r_j$ para $i \neq j$, é possível encontrar um polinômio $f$ de grau menor ou igual a $\ell$ em $\mathbb{F}_{q^n}[x]$ tal que $f(r_i) = s_i$ para todo $i$, $0 \leq i \leq \ell$. Além disso, tal polinômio é único.

Sejam $a(x) = \sum_{i=0}^{\ell} a_i x^i$ e $b(x) = \sum_{i=0}^{\ell} b_i x^i$ polinômios de grau $\ell$ sobre $\mathbb{F}_q$. Tomamos $2\ell + 1$ elementos, $u_0, \ldots, u_{2\ell}$, dois a dois distintos, e multiplicamos as imagens desses valores segundo as funções induzidas por $a$ e $b$:

$$a(u_0)b(u_0), \ldots, a(u_{2\ell})b(u_{2\ell}).$$

Depois aplicamos algum método de interpolação nesses valores a fim de obter o produto desejado.

Precisamos de um método rápido para avaliar polinômios em muitos pontos, assim como um método de interpolação rápido. O método mais rápido para avaliar um polinômio $a(x)$ num ponto particular $u$ é o método de Horner, baseado na identidade

$$a(u) = (\cdots((a_\ell u + a_{\ell-1})\, u + a_{\ell-2})\cdots)\, u + a_0,$$

que requer $O(\ell)$ operações em $\mathbb{F}_q$. Se aplicarmos o método de Horner para avaliar polinômios $a$ e $b$ nos pontos $u_0, \ldots, u_{2\ell}$, teríamos um custo total de $O(\ell^2)$ operações em $\mathbb{F}_q$. Além disso, uma implementação direta do algoritmo de interpolação de Lagrange também produziria um custo de $O(\ell^2)$. Assim, não teríamos nenhuma vantagem sobre a multiplicação clássica.

Schönhage e Strassen [118] superaram essas dificuldades usando o FFT. Este método rápido de avaliar polinômios é baseado nas potências de uma raiz $n$-ésima primitiva da unidade, digamos, $\omega$. O passo de interpolação também pode ser executado rapidamente usando o algoritmo FFT com $\omega^{-1}$ ao invés de $\omega$. Para mais detalhes sobre o assunto, veja, por exemplo, o livro de von zur Gathen e Gerhard [56]. Outras referências importantes são [18, 19, 117].

O custo de executar o FFT é $O(\ell \log \ell \log\log \ell)$. Logo, podemos multiplicar dois polinômios de grau $\ell$ usando $O(\ell \log \ell \log\log \ell)$ operações em $\mathbb{F}_q$. Usando métodos rápidos de multiplicação, o tempo assintótico de uma divisão de polinômios passa a ser $O(\ell \log \ell \log\log \ell)$. Ainda não está claro se o algoritmo de multiplicação rápida usando FFT é prático, mas alguns pesquisadores acreditam que sim [123].

Os métodos FFT também podem ser usados para calcular o mdc e o inverso, usando $O(\ell \log^2 \ell \log\log \ell)$ operações em $\mathbb{F}_q$; veja [2].

Uma referência para a multiplicação de polinômios usando FFT em outros anéis é o livro de Moreira e Saldanha [100].

### 2.1.4 Método da repetição de quadrados

Lembramos que estamos considerando $\mathbb{F}_{q^n} \cong \mathbb{F}_q[x]/(f)$, onde $f$ é um polinômio irredutível de grau $n$ sobre $\mathbb{F}_q$. Nesta seção, veremos como calcular $g^k$ módulo $f$, onde $g \in \mathbb{F}_q[x]$ e $\text{grau}(g) < n$. Pelo Lema 1.4.5, podemos supor que $0 \leq k < q^n - 1$.

O primeiro passo é calcular $g, g^2, g^4, \ldots, g^{2^{\lfloor \log k \rfloor}}$, onde a base do logaritmo é 2. Usando a representação binária de $k$, digamos, $k = \sum_{i=0}^{\lfloor \log k \rfloor} b_i 2^i$ onde $b_i \in \{0,1\}$, temos que

$$g^k = g^{\left(\sum_{i=0}^{\lfloor \log k \rfloor} b_i 2^i\right)} = \prod_{i=0}^{\lfloor \log k \rfloor} g^{b_i 2^i}.$$

**Exemplo 2.1.3.** Vamos calcular $g^{23} \pmod f$ usando o método da repetição dos quadrados. Temos que $23 = 1 + 2 + 2^2 + 2^4$. Calculamos $g_1 \equiv g \pmod f, g_2 \equiv g_1^2 \pmod f$, $g_4 \equiv g_2^2 \pmod f$, $g_8 \equiv g_4^2 \pmod f, g_{16} \equiv g_8^2 \pmod f$. Então $g^{23} \equiv g_1 g_2 g_4 g_{16} \pmod f$.

Este método é muito rápido já que custa $\lfloor \log k \rfloor$ cálculos de quadrados (ou multiplicações) e no máximo outras $\lfloor \log k \rfloor$ multiplicações para combinar os quadrados. Logo, requer aproximadamente $2\lfloor \log k \rfloor$ multiplicações, ao invés de $k-1$ multiplicações no método direto. Mais precisamente, o número de multiplicações necessárias é

$$\lfloor \log k \rfloor + \nu(k) - 1,$$

onde $\nu(k)$ é o número de 1's na representação binária de $k$. Em geral, são necessárias $O(\log k)$ multiplicações em $\mathbb{F}_{q^n}$. Como $k < q^n$, o custo é $O(n^2 \log q^n)$ operações em $\mathbb{F}_q$, ou seja, $O(n^3 \log q)$ operações em $\mathbb{F}_q$ usando métodos diretos e $O(n^2 \log q \log n \log\log n)$ operações em $\mathbb{F}_q$ usando métodos FFT.

Um caso de particular importância nos algoritmos de fatoração é quando $k = q$. Veremos isto nas Seções 3.2 e 3.6. Neste caso, o custo é de $O(n^2 \log q)$ operações em $\mathbb{F}_q$ usando métodos diretos e de $O(n \log q \log n \log\log n)$ operações em $\mathbb{F}_q$ usando métodos FFT.

Na prática, em $\mathbb{F}_{2^n}$, calcular $a^2$ pode ser mais rápido do que multiplicar $a \cdot a$. Por exemplo, segundo [27, Algoritmo 10.14], o cálculo do quadrado é aproximadamente 20% mais rápido do que a multiplicação clássica. Esta diferença é bastante significante nas aplicações.

## 2.2 Elementos primitivos

Uma propriedade importante de corpos finitos é que qualquer elemento não nulo é uma potência de um certo elemento fixo.

**Teorema 2.2.1.** *O grupo multiplicativo de qualquer corpo finito é cíclico.*

*Demonstração.* Suponhamos $q > 2$. Pelo teorema de estrutura de grupos abelianos finitos, temos que $\mathbb{F}_q^* \cong \mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2} \times \cdots \times \mathbb{Z}_{n_k}$, onde $n_1 \geq 2$ e $n_1 \mid n_2 \mid \cdots \mid n_k$. Portanto, temos que $a^{n_k} = 1$ para todo $a \in \mathbb{F}_q^*$. Então $\mathbb{F}_q^*$ consiste de raízes de $x^{n_k} - 1$ e assim $|\mathbb{F}_q^*| \leq n_k$. Por outro lado, $|\mathbb{F}_q^*| = n_1 \cdots n_k$. Portanto, $k = 1$ e $\mathbb{F}_q^* \cong \mathbb{Z}_{n_1}$. □

**Definição 2.2.2.** Um gerador de $\mathbb{F}_q^*$ é chamado um *elemento primitivo*.

Teorema 2.2.1 fornece uma maneira bastante conveniente de representar os elementos não nulos de um corpo finito. Suponhamos que $\alpha$ seja um elemento primitivo de $\mathbb{F}_q$. Então $\mathbb{F}_q^* = \{1, \alpha, \alpha^2, \ldots, \alpha^{q-2}\}$. Infelizmente a prova deste resultado não dá nenhuma indicação de como encontrar tal elemento primitivo. O método mais ingênuo é o de tentativa e eliminação de erros, que consiste na busca de um elemento $\alpha \in \mathbb{F}_q$ cuja ordem é $q-1$, isto é, $q-1 = \min_{i \in \mathbb{N}}\{i \colon \alpha^i = 1\}$. Uma vez que encontramos tal $\alpha$, então teremos encontrado todos os elementos primitivos, a saber, qualquer $\alpha^i$, onde $i$ e $q-1$ são relativamente primos. Assim, há $\phi(q-1)$ elementos primitivos, onde $\phi$ é a função de Euler (veja Apêndice C).

**Exemplo 2.2.3.** Em $\mathbb{F}_{13}$, temos que

$$3^1 = 3, \quad 3^2 = 9, \quad 3^3 = 1,$$

logo 3 não é um elemento primitivo de $\mathbb{F}_{13}$. Por outro lado, temos que

$$2^1 = 2, \quad 2^2 = 4, \quad 2^3 = 8, \quad 2^4 = 3, \quad 2^5 = 6, \quad 2^6 = 12,$$

$$2^7 = 11, \quad 2^8 = 9, \quad 2^9 = 5, \quad 2^{10} = 10, \quad 2^{11} = 7, \quad 2^{12} = 1,$$

ou seja, 2 é um elemento primitivo de $\mathbb{F}_{13}$. Como $\phi(12) = 4$, há três outros elementos primitivos de $\mathbb{F}_{13}$, a saber, $2^5 = 6$, $2^7 = 11$, $2^{11} = 7$.

No caso em que $q$ é muito grande, o método descrito acima não é muito conveniente. Na verdade, ainda não se conhece um algoritmo de tempo polinomial em $\log q$ para o problema de encontrar elementos primitivos. Veremos na seção seguinte o algoritmo de Gauss que é eficiente para $q$ pequeno.

Ao representarmos um elemento num corpo finito como uma potência de um elemento primitivo, as operações de multiplicação, exponenciação e cálculo de inversos são facilmente realizadas. Primeiramente, lembramos que, pelo Lema 1.4.5, se $a > q-1$ então podemos tomar $r$ tal que $r \equiv a \pmod{q-1}$ e $0 \leq r < q-1$ para obtermos $\gamma^a = \gamma^r$, para todo $\gamma \in \mathbb{F}_q$. Suponhamos que $\gamma$ é um elemento primitivo de $\mathbb{F}_q$. Para a multiplicação, temos que $\gamma^a\gamma^b = \gamma^{a+b}$. Para a exponenciação, temos que $(\gamma^a)^k = \gamma^{ak}$ para todo $k \in \mathbb{Z}$. Em particular, para o cálculo de inversos, temos que $(\gamma^a)^{-1} = \gamma^{-a}$. Sempre que o expoente for maior que $q-1$, podemos obter a redução módulo $q-1$ discutida acima.

**Exemplo 2.2.4.** O elemento $\gamma$ correspondente ao polinômio $1+x$ é um elemento primitivo em $\mathbb{F}_{16} \cong \mathbb{F}_2[x]/(x^4+x+1)$. De fato, temos a seguinte tabela das potências de $\gamma$ juntamente com suas representações polinomiais.

| potência de $\gamma$ | polinômio |
|---|---|
| 1 | 1 |
| $\gamma$ | $x+1$ |
| $\gamma^2$ | $x^2+1$ |
| $\gamma^3$ | $x^3+x^2+x+1$ |
| $\gamma^4$ | $x$ |
| $\gamma^5$ | $x^2+x$ |
| $\gamma^6$ | $x^3+x$ |
| $\gamma^7$ | $x^3+x^2+1$ |
| $\gamma^8$ | $x^2$ |
| $\gamma^9$ | $x^3+x^2$ |
| $\gamma^{10}$ | $x^2+x+1$ |
| $\gamma^{11}$ | $x^3+1$ |
| $\gamma^{12}$ | $x^3$ |
| $\gamma^{13}$ | $x^3+x+1$ |
| $\gamma^{14}$ | $x^3+x^2+x$ |

Para ilustrarmos as operações aritméticas usando o elemento primitivo $\gamma$, calculamos:

1. $\gamma^7\gamma^{14} = \gamma^{21} = \gamma^6$; isto é muito mais rápido do que calcular $[(x^3+x^2+1)(x^3+x^2+x)] \pmod{x^4+x+1}$;

2. $(\gamma^{13})^{-1} = \gamma^{-13} = \gamma^2$; isto é muito mais rápido do que calcular $(x^3+x+1)^{-1} \pmod{x^4+x+1}$ usando o algoritmo euclideano estendido;

3. $\gamma^i = \gamma^3$ para todo $i$ tal que $i \equiv 3 \pmod{15}$; isto é muito mais rápido do que calcular $(x+1)^{318} \pmod{x^4+x+1}$, por exemplo.

### 2.2.1 Algoritmo de Gauss

Apresentaremos um algoritmo desenvolvido por Gauss para encontrar elementos primitivos, que é eficiente apenas para $q$ pequeno. Este algoritmo fornece uma seqüência de elementos $\alpha_1, \alpha_2, \ldots, \alpha_i \in \mathbb{F}_q$ tais que

$$\mathrm{ord}(\alpha_1) < \mathrm{ord}(\alpha_2) < \cdots < \mathrm{ord}(\alpha_i) = q-1.$$

**Algoritmo 2.2.5.** [Gauss]

(1) Inicialize $i = 1$. Seja $\alpha_1 \in \mathbb{F}_q^*$. Defina $t_1 = \mathrm{ord}(\alpha_1)$.

(2) Se $t_i = q-1$, então pare. O elemento $\alpha_i$ é um elemento primitivo.

(3) Caso contrário, escolha um elemento $\beta \in \mathbb{F}_q^*$, que não seja uma potência de $\alpha_i$.

Seja $s = \mathrm{ord}(\beta)$. Se $s = q-1$, então atribua $\alpha_{i+1} = \beta$, faça $i = i+1$ e pare.

(4) Encontre $d$ e $e$ tais que $d \mid t_i$, $e \mid s$, $de = \mathrm{mmc}(t_i, s)$ e $\mathrm{mdc}(d, e) = 1$. Sejam $\alpha_{i+1} = \alpha_i^{t_i/d}\beta^{s/e}$ e $t_{i+1} = \mathrm{mmc}(t_i, s)$. Faça $i = i+1$ e vá para o passo 2.

Seguem alguns comentários sobre o algoritmo acima.

**Observações:**

(a) No passo 3, temos que $s \nmid t_i$, já que todas as soluções de $x^{t_i} = 1$ devem ser potências de $\alpha_i$, e $\beta$ não é uma potência de $\alpha_i$. Logo, o $\mathrm{mmc}(t_i, s)$ é um mútiplo de $t_i$ que é maior que $t_i$.

(b) No passo 4, usamos o fato de que, dados dois inteiros $t_i$ e $s$, é possível encontrar $d$ e $e$ tais que $d \mid t_i$, $e \mid s$, $\mathrm{mdc}(d, e) = 1$ e $\mathrm{mmc}(t_i, s) = de$ (exercício).

**Exemplo 2.2.6.** Como $\mathrm{mmc}(12, 15) = 60$, uma possível escolha é $d = 4$, $e = 15$, já que $4 \mid 12$, $15 \mid 15$, $\mathrm{mdc}(4, 15) = 1$ e $4 \cdot 15 = \mathrm{mmc}(12, 15) = 60$.

(c) No passo 4, temos que $\mathrm{ord}(\alpha_{i+1}) = \mathrm{mmc}(t_i, s)$. Com efeito, como $\alpha_i$ tem ordem $t_i$, então $\alpha_i^{t_i/d}$ tem ordem $d$. Como $\beta$ tem ordem $s$, $\beta^{s/e}$ tem ordem $e$. Por outro lado, usando o teorema a seguir, o produto de $\alpha_i^{t_i/d}\beta^{s/e}$ tem ordem $de = \mathrm{mmc}(t_i, s)$, quando $d$ e $e$ são relativamente primos.

**Teorema 2.2.7.** *Sejam* $\mathrm{ord}(\alpha) = m$, $\mathrm{ord}(\beta) = n$ *e* $\mathrm{mdc}(m, n) = 1$. *Então* $\mathrm{ord}(\alpha\beta) = mn$.

*Demonstração.* Seja $\mathrm{ord}(\alpha\beta) = \ell$. Então

$$1 = (\alpha\beta)^{\ell} = (\alpha\beta)^{\ell m} = \alpha^{\ell m}\beta^{\ell m} = \beta^{\ell m},$$

já que $\alpha^m = 1$. Assim, $\beta^{\ell m} = 1$ e como $\mathrm{ord}(\beta) = n$, temos que $n \mid \ell m$. Como $\mathrm{mdc}(m, n) = 1$, segue que $n \mid \ell$.

Similarmente, temos que $m \mid \ell$. Combinando os comentários anteriores, temos que $\mathrm{mdc}(n, m) = 1$, $m \mid \ell$, $n \mid \ell$, $mn \mid \ell = \mathrm{ord}(\alpha\beta)$.

Reciprocamente, $(\alpha\beta)^{mn} = \alpha^{mn}\beta^{mn} = 1$ e assim temos que $\mathrm{ord}(\alpha\beta) \mid mn$. Portanto $mn = \mathrm{ord}(\alpha\beta)$. □

O algoritmo de Gauss gera sucessivamente os elementos $\alpha_{i+1}$ com ordem $t_{i+1} = \mathrm{mmc}(t_i, s)$, já que $\alpha_{i+1} = \alpha_i^{t_i/d}\beta^{s/e}$, $\mathrm{ord}(\alpha_i^{t_i/d}) = d$, $\mathrm{ord}(\beta^{s/e}) = e$ e $\mathrm{mdc}(d, e) = 1$. Portanto, o teorema anterior garante que $\mathrm{ord}(\alpha_{i+1}) = de = \mathrm{mmc}(t_i, s)$.

Além disso, como o $\mathrm{mmc}(t_i, s)$ é um multiplo de $t_i$ e é maior que $t_i$, porque $s \nmid t_i$, pela Observação (a), este processo sempre termina com um elemento primitivo.

**Exemplo 2.2.8.**

1. Vamos considerar $\mathbb{F}_7^*$.

   (1) Tomamos $\alpha_1 = 2$. Então $\alpha_1^2 = 4$, $\alpha_1^3 = 1$. Assim, $t_1 = \mathrm{ord}(\alpha_1) = 3$.

   (2) $t_1 \neq q - 1 = 6$

   (3) Escolhemos $\beta$ diferente de uma potência de $\alpha_1$. Digamos que $\beta = 3$: $\beta^2 = 2$, $\beta^3 = 6$, $\beta^4 = 4$, $\beta^5 = 5$, $\beta^6 = 1$. Assim, $\mathrm{ord}(\beta) = 6$. Pare, $\beta = 3$ é um elemento primitivo.

2. Seja $\mathbb{F}_{5^2}^*$ definido como $\mathbb{F}_5[x]/(x^2 + 3)$.

   (1) Tomamos $\alpha_1 = x$. Então $\alpha_1 = x$, $\alpha_1^2 = x^2 = 2$, $\alpha_1^3 = 2x$, $\alpha_1^4 = 2x^2 = 4$, $\alpha_1^5 = 4x$, $\alpha_1^6 = 4x^2 = 3$, $\alpha_1^7 = 3x$, $\alpha_1^8 = 3x^2 = 1$. Assim, $t_1 = \mathrm{ord}(\alpha_1) = 8$.

   (2) $t_1 = 8 \neq q - 1 = 24$.

   (3) Escolhemos $\beta$ diferente de uma potência de $\alpha_1$. Digamos, $\beta = x + 1$: $\beta^2 = x^2 + 2x + 1 = 2x + 3$, $\beta^3 = 2x^2 + 3 = 2$, $\beta^4 = 2x + 2$, $\beta^5 = 2x^2 + 4x + 2 = 4x + 1$, $\beta^6 = 4x^2 + 1 = 4$, $\beta^7 = 4x + 4$, $\beta^8 = 4x^2 + 3x + 4 = 3x + 2$, $\beta^9 = 3x^2 + 2 = 3$, $\beta^{10} = 3x + 3$, $\beta^{11} = 3x^2 + x + 3 = x + 4$, $\beta^{12} = x^2 + 4 = 1$. Então $\beta = x + 1$ tem ordem 12 e $12 \neq q - 1 = 24$.

   (4) $\alpha_1 = x, \mathrm{ord}(\alpha_1) = t_1 = 8$; $\beta = x + 1$, $\mathrm{ord}(\beta) = s = 12$. Precisamos $d, e$ tais que $de = \mathrm{mmc}(t_1, s) = \mathrm{mmc}(8, 12) = 24$, $\mathrm{mdc}(d, e) = 1$, $d \mid t_1$ e $e \mid s$. Temos que

   $$t_1 = 8 = 2 \cdot 2 \cdot 2, \qquad s = 12 = 2 \cdot 2 \cdot 3.$$

   Assim, temos que $\mathrm{mmc}(8, 12) = 2 \cdot 2 \cdot 2 \cdot 3$ e $\mathrm{mdc}(8, 3) = 1$. Tomamos $d = 8$ e $e = 3$. Logo, encontramos $d \mid 8$, $e \mid 12$, $\mathrm{mdc}(d, e) = 1$ e $\mathrm{mmc}(t_1, s) = de = 24$. Então

   $$\alpha_2 = \alpha_1^{t_1/d} \beta^{s/e} = \alpha_1^{8/8} \beta^{12/3} = \alpha_1 \beta^4.$$

Isto é

$$\alpha_2 = x(x+1)^4 = x(2x+2) = 2x^2 + 2x = 2x + 4.$$

Portanto, $\alpha_2 = 2x + 4$ tem ordem $de = \text{mmc}(t_1, s) = \text{mmc}(8, 12) = 24 = q - 1$. Logo, $\alpha_2$ é um elemento primitivo em $\mathbb{F}_5[x]/(x^2+3)$.

Voltaremos ao problema de encontrar elementos primitivos na Seção 3.4.

## 2.3 Elementos normais

Nesta seção, consideramos bases formadas por um elemento $\alpha \in \mathbb{F}_{q^n}$ e seus conjugados para $\mathbb{F}_{q^n}$ sobre $\mathbb{F}_q$.

**Definição 2.3.1.** Seja $\alpha \in \mathbb{F}_{q^n}$. Uma *base normal* de $\mathbb{F}_{q^n}$ sobre $\mathbb{F}_q$ é uma base da forma $N = \{\alpha, \alpha^q, \ldots, \alpha^{q^{n-1}}\}$. Neste caso, dizemos que $\alpha$ é um *elemento normal* de $\mathbb{F}_{q^n}$ sobre $\mathbb{F}_q$ e que $\alpha$ *gera* a base normal $N$. Os elementos em $N$ são chamados os *conjugados* de $\alpha$.

A definição acima indica uma outra maneira de representar os elementos de um corpo finito. Elementos normais são desejados por produzirem implementações mais rápidas, principalmente em aplicações onde a exponenciação é necessária com certa freqüência. Veremos isso na Seção 2.3.1.

Os elementos primitivos e os elementos normais são ambos definidos em termos de certas potências de um único elemento. No entanto, têm naturezas distintas. Os elementos primitivos geram por si só todos os elementos de $\mathbb{F}_q^*$, no sentido de que qualquer elemento em $\mathbb{F}_q^*$ é uma potência de um elemento primitivo. Por exemplo, se $\gamma$ é um elemento primitivo de $\mathbb{F}_q$ e se $a \in \mathbb{F}_q^*$, então $a = \gamma^i$ para certo $i$. Por outro lado, elementos normais são elementos com a propriedade de que suas $q$-potências formam uma base. Assim, se $\alpha$ é um elemento normal de $\mathbb{F}_{q^n}$ sobre $\mathbb{F}_q$, temos uma outra representação para $a$, a saber, $a = a_0\alpha + a_1\alpha^q + \cdots + a_{n-1}\alpha^{q^{n-1}}$, para certos $a_0, a_1, \ldots, a_{n-1}$ em $\mathbb{F}_q$.

Uma outra observação imediata é que se $\alpha$ é normal, então qualquer conjugado de $\alpha$ também é normal.

Bases normais existem em qualquer extensão finita de um corpo finito. Por volta de 1850, Eisenstein e Schönemann apresentaram independentemente provas parciais deste resultado, mas foi apenas em 1888 que Hensel [70] conseguiu uma prova completa. Tal resultado é conhecido como o *Teorema da Base Normal*, cuja prova pode ser encontrada em [75, Teorema 3.1.1] e [90, Teorema 2.35], por exemplo.

**Teorema 2.3.2.** *Para qualquer inteiro positivo $n$, existe uma base normal de $\mathbb{F}_{q^n}$ sobre $\mathbb{F}_q$.*

O número de elemento normais foi estabelecido por Ore em 1934; veja o livro de Jungnickel [75, Teorema 3.1.5], por exemplo.

**Teorema 2.3.3.** *Seja $n$ um inteiro positivo. Escreva $n = p^a m$, onde $m$ não é múltiplo de $p = \mathrm{car}(\mathbb{F}_q)$. O número de bases normais de $\mathbb{F}_{q^n}$ sobre $\mathbb{F}_q$ é*

$$\frac{1}{n}\Phi_q(x^n-1) = \frac{q^n}{n}\prod_{d|m}\left(1-q^{-o_d(q)}\right)^{\phi(d)/o_d(q)},$$

*onde $\Phi_q(f)$ é o número de polinômios de grau menor que o grau de $f$ relativamente primos com $f$, $o_n(b)$ é a ordem de $b$ módulo $n$ e $\phi(d)$ é o número de inteiros positivos menores que $d$ relativamente primos com $d$.*

Apesar da fórmula para o número de bases normais existir, ela não é muito clara, já que depende de outros parâmetros. Por isso, vários autores têm pesquisado cotas inferiores e superiores para este número que possam dar uma estimativa mais clara. Gao e Panario [51, Teorema 3.4] obtiveram cotas, quando $n = (q-1)(q^2-1)\cdots(q^m-1)$ e $m$ é um inteiro maior que 1. O resultado é

$$\frac{q^n}{e^{0.83}n(1+\log_q n)} \leq \frac{1}{n}\Phi_q(x^n-1) \leq \frac{q^n}{e^{\gamma-c_q}n\sqrt{1+\log_q n}}, \qquad (2.1)$$

onde $c_q = q/(q-1)(\sqrt{q}-1)$ e $\gamma = 0.577216\ldots$ é a constante de Euler. Veja também [43] para outras cotas obtidas por Frandsen. Von zur Gathen e Giesbrecht [57] mostraram que a probabilidade de que um elemento arbitrário em $\mathbb{F}_{q^n}$ seja um elemento normal é maior que $1/(16\log_q n)$.

As cotas mencionadas acima mostram que há um grande número de elementos normais num corpo finito. Para encontrar um elemento normal em $\mathbb{F}_{q^n}$, um método simples é testar se um elemento escolhido aleatoriamente uniforme é ou não normal, repetindo o teste enquanto a resposta for negativa [57]. A questão é como testar a normalidade de um elemento. A seguir, apresentaremos um critério simples para verificar a normalidade de um elemento em $\mathbb{F}_{q^n}$; veja [75, Teorema 3.1.8] ou [90, Teorema 2.39].

**Teorema 2.3.4.** *Um elemento $\alpha$ é normal em $\mathbb{F}_{q^n}$ sobre $\mathbb{F}_q$ se e somente se* $\mathrm{mdc}(x^n - 1, \alpha^{q^{n-1}} x^{n-1} + \cdots + \alpha^q x + \alpha) = 1$ *em* $\mathbb{F}_{q^n}[x]$.

### 2.3.1 Aritmética usando bases normais

Nesta seção, discutiremos as operações básicas de aritmética entre elementos num corpo finito usando a representação em termos de bases normais. Para isso, seja $N = \{\alpha_0, \alpha_1, \ldots, \alpha_{n-1}\}$ uma base de $\mathbb{F}_{q^n}$ sobre $\mathbb{F}_q$ com $\alpha_i = \alpha^{q^i}$. Sejam $A$ e $B$ elementos em $\mathbb{F}_{q^n}$ definidos por

$$A = \sum_{i=0}^{n-1} a_i \alpha_i \quad \text{e} \quad B = \sum_{i=0}^{n-1} b_i \alpha_i,$$

onde $a_i, b_i \in \mathbb{F}_q$. Abusando a notação, também representaremos $A$ e $B$ por $A = (a_0, \ldots, a_{n-1})$ e $B = (b_0, \ldots, b_{n-1})$.

A adição de $A$ e $B$ é trivial:

$$A + B = (a_0 + b_0, a_1 + b_1, \ldots, a_{n-1} + b_{n-1}).$$

O cálculo da $q$-ésima potência também é imediata [70]. De fato, temos

$$A^q = \left( \sum_{i=0}^{n-1} a_i \alpha_i \right)^q = \sum_{i=0}^{n-1} a_i^q \alpha_i^q = \sum_{i=0}^{n-1} a_i \alpha_{i+1} = (a_{n-1}, a_0, \ldots, a_{n-2}),$$

ou seja, calcular uma $q$-ésima potência é equivalente a efetuar um deslocamento nas coordenadas, o que é muito barato em termos de implementação.

A complexidade da exponenciação em geral foi analisada inicialmente por Stinson [127] para o caso $q = 2$ e depois por von zur Gathen [54] para qualquer $q$. Eles mostraram que a exponenciação pode ser feita com $O(n/\log_q n)$ multiplicações em $\mathbb{F}_q$, ou seja, $O(n^2 \log\log n)$ operações em $\mathbb{F}_q$; veja também [47, 48].

Seja $C = (c_0, \ldots, c_{n-1}) = \sum_{i=0}^{n-1} c_i \alpha_i$ o produto de $A$ e $B$. Queremos calcular cada $c_k$ em função dos $a_i$'s e $b_j$'s. Como

$$C = \sum_{i,j} a_i b_j \alpha_i \alpha_j,$$

escrevemos cada $\alpha_i \alpha_j$ como combinação linear dos elementos da base $N$; digamos que

$$\alpha_i \alpha_j = \sum_{k=0}^{n-1} t_{i,j}^{(k)} \alpha_k \tag{2.2}$$

para certos $t_{i,j}^{(k)} \in \mathbb{F}_q$. Assim, obtemos que $c_k = \sum_{i,j} a_i b_j t_{i,j}^{(k)}$, o que sugere que $c_k$ é uma entrada do produto de certas matrizes. De fato, considerando $A$ e $B$ como matrizes linhas e $T_k = t_{i,j}^{(k)}$ como uma matriz de ordem $n \times n$ para cada $k$, $0 \leq k \leq n-1$, temos que $c_k = A T_k B^T$ onde $B^T$ é a matriz transposta de $B$. Portanto, para multiplicarmos os elementos em $\mathbb{F}_q$ usando $N$, basta calcularmos de antemão $(T_0, \ldots, T_{n-1})$, conhecido como a *tabela de multiplicação da base normal* $N$.

**Lema 2.3.5.** *Com a notação acima, temos que* $t_{i,j}^{(k)} = t_{i-k,j-k}^{(0)}$, *para quaisquer* $i, j, k$ *com* $0 \leq i, j, k \leq n-1$, *onde os índices são tomados módulo* $n$.

*Demonstração.* Elevando ambos os lados da Equação (2.2) à potência $q$, obtemos $\alpha_{i+1}\alpha_{j+1} = \sum_{k=0}^{n-1} t_{i,j}^{(k)} \alpha_{k+1}$ onde $\alpha_n = \alpha_0$. Assim, $t_{i,j}^{(k)} = t_{i+1,j+1}^{(k+1)}$, isto é, $t_{i,j}^{(k)} = t_{i-1,j-1}^{(k-1)}$. Uma indução dá que $t_{i,j}^{(k)} = \cdots = t_{i-k,j-k}^{(0)}$, onde os índices são tomados módulo $n$. □

Esta propriedade implica que as entradas de $T_k = (t_{i,j}^{(k)})$ correspondam exatamente às entradas de $T_0 = (t_{i,j}^{(0)})$ em posições diferentes. Isto significa que precisamos apenas de $T_0$ para implementar a multiplicação. De fato, cada coeficiente $c_k$, $0 \leq k \leq n-1$, do

produto $AB$ pode ser obtido usando $T_0$ e deslocamentos cíclicos de $k$ posições dos coeficientes em $A$ e $B$. Por simplicidade, seja $T = T_0$. Temos assim a seguinte proposição.

**Proposição 2.3.6.** *Sejam $A^{[k]}$ e $B^{[k]}$ deslocamento cíclicos por $k$ posições de $A$ e $B$, respectivamente. Então, temos*

$$c_k = A^{[k]} T (B^{[k]})^T,$$

*onde $(B^{[k]})^T$ é a matriz transposta de $B^{[k]}$.*

A matriz $T$ tem um papel importante na implementação da multiplicação baseada em elementos normais. O número de elementos não nulos em $T$ dá a quantidade de portas lógicas no circuito implementando a base normal; veja por exemplo [93, 101]. Tal número é chamado a *complexidade* da base normal $N$ e é denotado por $c_N$. Mullin, Onyszchuk, Vanstone e Wilson [101] provaram a seguinte cota inferior para $c_N$.

**Teorema 2.3.7.** *Se $N$ é uma base normal de $\mathbb{F}_{q^n}$ sobre $\mathbb{F}_q$, então $c_N \geq 2n - 1$.*

Bases normais de complexidade $2n-1$ são importantes na prática, já que possuem a menor complexidade possível e assim recebem um nome especial.

**Definição 2.3.8.** Se $N$ é uma base normal de $\mathbb{F}_{q^n}$ sobre $\mathbb{F}_q$ cuja complexidade $c_N$ é $2n - 1$, então $N$ é chamada uma *base normal ótima.*

Uma multiplicação usando base normais geralmente custa $O(n^3)$ operações em $\mathbb{F}_q$, segundo o método mais ingênuo descrito acima, enquanto que esse custo cai para $O(n^2)$ operações em $\mathbb{F}_q$, usando bases normais ótimas.

### 2.3.2 Bases normais ótimas

Apesar de bases normais sempre existirem, o mesmo não acontece com bases normais ótimas. Mullin et al. [101] apresentaram condições para a existência de bases normais ótimas. Como discutiremos em seguida, na verdade, tais condições são necessárias e suficientes.

**Teorema 2.3.9.** *Sejam $q$ uma potência de um primo e $n+1$ um primo. Se $q$ é um elemento primitivo em $\mathbb{Z}_{n+1}$, então existe uma base normal ótima de $\mathbb{F}_{q^n}$ sobre $\mathbb{F}_q$.*

**Teorema 2.3.10.** *Seja $2n+1$ um primo. Suponhamos que 2 seja um elemento primitivo em $\mathbb{Z}_{2n+1}$, ou que $2n+1 \equiv 3 \pmod 4$ e 2 gere os resíduos quadráticos[2] em $\mathbb{Z}_{2n+1}$. Então $\alpha = \zeta + \zeta^{-1}$ gera uma base normal ótima para $\mathbb{F}_{2^n}$ sobre $\mathbb{F}_2$, onde $\zeta$ é uma raiz $(2n+1)$-ésima primitiva da unidade*[3]

Uma base normal ótima proveniente do Teorema 2.3.9 é chamada uma *base normal ótima do tipo I*, enquanto a proveniente do Teorema 2.3.10 é chamada uma *base normal ótima do tipo II*. Mullin et al. [101] conjecturaram que toda base normal ótima de $\mathbb{F}_{q^n}$ sobre $\mathbb{F}_q$ é equivalente a uma base construída pelo Teorema 2.3.9 ou pelo Teorema 2.3.10. Este resultado foi provado por Gao e Lenstra [50].

Na Tabela 2.2, apresentamos todos os valores de $n \leq 2000$ para os quais existe uma base normal ótima de $\mathbb{F}_{2^n}$ sobre $\mathbb{F}_2$. Esta tabela apareceu na tese de Gao [45, páginas 69-70]. As seguintes convenções foram usadas por Gao: $*$ indica que há uma base do tipo I; † indica que há uma base do tipo I e uma base do tipo II; todas as demais entradas indicam que há uma base do tipo II.

Na próxima seção, consideraremos a seguinte questão: o que fazemos quando não há uma base normal ótima?

### 2.3.3 Elementos normais de complexidade baixa

Quando nenhuma base normal ótima existe, elementos de "baixa complexidade" são usados. Apesar de não existir uma definição precisa para o termo "baixa complexidade", espera-se que o elemento normal tenha a menor complexidade possível.

Na prática, elementos normais de baixa complexidade são bastante usados. Por exemplo, o NIST[4] [105] recomenda usar elementos

[2] Sejam $a$ e $n$ inteiros coprimos. Se existe um inteiro $x$ tal que $x^2 \equiv a \pmod n$, dizemos que $a$ é um *resíduo quadrático módulo $n$*.

[3] Um gerador do grupo cíclico das $n$-ésimas raízes da unidade é chamado uma *raiz $n$-ésima primitiva da unidade*.

[4] National Institute of Standards and Technology.

normais de baixa complexidade na criptografia de curvas elípticas sobre $\mathbb{F}_{2^m}$, para $m = 163, 233, 283, 409$ e $571$. Observamos que existem várias implementações muito eficientes em software da multiplicação usando bases normais; veja [33, 113]. Para maiores detalhes sobre as implementações desses métodos, veja [34].

O problema de classificar bases normais de complexidade baixa ainda está em aberto. Na realidade, não há muitos artigos sobre este assunto, apesar de o problema ser de grande interesse. Os poucos artigos existentes só apareceram nos últimos anos.

As primeiras contruções de elementos normais de baixa complexidade apareceram em [4, 11]. Young e Panario [136] mostraram experimentalmente que, em corpos finitos de característica 2 e grau de extensão $n$, elemento normais de complexidade até $3n$ ocorrem somente em corpos finitos que possuem elementos normais ótimos. Com este resultado, eles obtiveram certas caracterizações de tais elementos. Wan e Zhou [132] estenderam parte desses resultados para corpos finitos de característica ímpar. Outros experimentos e conjecturas para elementos de baixa complexidade em $\mathbb{F}_{2^n}$ podem ser encontrados em [94]. Recentemente, uma construção de elementos de complexidade baixa em $\mathbb{F}_{q^n}$ baseada no traço de um elemento normal ótimo foi desenvolvida [22].

Tabela 2.2: Valores de $n \leq 2000$ para os quais há bases normais ótimas em $\mathbb{F}_{2^n}$ sobre $\mathbb{F}_2$.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 2† | 113 | 293 | 473 | 676∗ | 873 | 1110 | 1310 | 1533 | 1790 |
| 3 | 119 | 299 | 483 | 683 | 876∗ | 1116∗ | 1323 | 1539 | 1791 |
| 4∗ | 130∗ | 303 | 490∗ | 686 | 879 | 1118 | 1329 | 1541 | 1806 |
| 5 | 131 | 306 | 491 | 690 | 882∗ | 1119 | 1331 | 1548∗ | 1811 |
| 6 | 134 | 309 | 495 | 700∗ | 891 | 1121 | 1338 | 1559 | 1818 |
| 9 | 135 | 316∗ | 508∗ | 708∗ | 893 | 1122∗ | 1341 | 1570∗ | 1821 |
| 10∗ | 138∗ | 323 | 509 | 713 | 906∗ | 1133 | 1346 | 1583 | 1829 |
| 11 | 146 | 326 | 515 | 719 | 911 | 1134 | 1349 | 1593 | 1835 |
| 12∗ | 148∗ | 329 | 519 | 723 | 923 | 1146 | 1353 | 1601 | 1838 |
| 14 | 155 | 330 | 522∗ | 725 | 930 | 1154 | 1355 | 1618∗ | 1845 |
| 18† | 158 | 338 | 530 | 726 | 933 | 1155 | 1359 | 1620∗ | 1850 |
| 23 | 162∗ | 346∗ | 531 | 741 | 935 | 1166 | 1370 | 1626 | 1854 |
| 26 | 172∗ | 348∗ | 540∗ | 743 | 938 | 1169 | 1372∗ | 1636∗ | 1859 |
| 28∗ | 173 | 350 | 543 | 746 | 939 | 1170∗ | 1380∗ | 1649 | 1860∗ |
| 29 | 174 | 354 | 545 | 749 | 940∗ | 1178 | 1394 | 1653 | 1863 |
| 30 | 178∗ | 359 | 546∗ | 755 | 946∗ | 1185 | 1398 | 1659 | 1866† |
| 33 | 179 | 371 | 554 | 756∗ | 950 | 1186∗ | 1401 | 1661 | 1876∗ |
| 35 | 180∗ | 372∗ | 556∗ | 761 | 953 | 1194 | 1409 | 1666∗ | 1883 |
| 36∗ | 183 | 375 | 558 | 765 | 965 | 1199 | 1418 | 1668∗ | 1889 |
| 39 | 186 | 378† | 561 | 771 | 974 | 1211 | 1421 | 1673 | 1898 |
| 41 | 189 | 378∗ | 562∗ | 772∗ | 975 | 1212∗ | 1425 | 1679 | 1900∗ |
| 50 | 191 | 386 | 575 | 774 | 986 | 1218 | 1426∗ | 1685 | 1901 |
| 51 | 194 | 388∗ | 585 | 779 | 989 | 1223 | 1430 | 1692∗ | 1906∗ |
| 52∗ | 196∗ | 393 | 586∗ | 783 | 993 | 1228∗ | 1439 | 1703 | 1923 |
| 53 | 209 | 398 | 593 | 785 | 998 | 1229 | 1443 | 1706 | 1925 |
| 58∗ | 210† | 410 | 606 | 786∗ | 1013 | 1233 | 1450∗ | 1730 | 1926 |
| 60∗ | 221 | 411 | 611 | 791 | 1014 | 1236∗ | 1451 | 1732∗ | 1930∗ |
| 65 | 226∗ | 413 | 612∗ | 796∗ | 1018∗ | 1238 | 1452∗ | 1733 | 1931 |
| 66∗ | 230 | 414 | 614 | 803 | 1019 | 1251 | 1454 | 1734 | 1938 |
| 69 | 231 | 418∗ | 615 | 809 | 1026 | 1258∗ | 1463 | 1740∗ | 1948∗ |
| 74 | 233 | 419 | 618† | 810 | 1031 | 1265 | 1469 | 1745 | 1953 |
| 81 | 239 | 420∗ | 629 | 818 | 1034 | 1269 | 1478 | 1746∗ | 1955 |
| 82∗ | 243 | 426 | 638 | 820∗ | 1041 | 1271 | 1481 | 1749 | 1958 |
| 83 | 245 | 429 | 639 | 826∗ | 1043 | 1274 | 1482∗ | 1755 | 1959 |
| 86 | 251 | 431 | 641 | 828∗ | 1049 | 1275 | 1492∗ | 1758 | 1961 |
| 89 | 254 | 438 | 645 | 831 | 1055 | 1276∗ | 1498∗ | 1763 | 1965 |
| 90 | 261 | 441 | 650 | 833 | 1060∗ | 1278 | 1499 | 1766 | 1972∗ |
| 95 | 268∗ | 442∗ | 651 | 834 | 1065 | 1282∗ | 1505 | 1769 | 1973 |
| 98 | 270 | 443 | 652∗ | 846 | 1070 | 1289 | 1509 | 1773 | 1978∗ |
| 99 | 273 | 453 | 653 | 852∗ | 1090∗ | 1290∗ | 1511 | 1778 | 1983 |
| 100∗ | 278 | 460∗ | 658∗ | 858∗ | 1103 | 1295 | 1518 | 1779 | 1986∗ |
| 105 | 281 | 466∗ | 659 | 866 | 1106 | 1300∗ | 1522∗ | 1785 | 1994 |
| 106∗ | 292∗ | 470 | 660∗ | 870 | 1108∗ | 1306∗ | 1530∗ | 1786∗ | 1996∗ |

# Capítulo 3

# Polinômios irredutíveis

No Capítulo 2, vimos que podemos representar os elementos em $\mathbb{F}_q$ por meio de polinômios. Esta representação é possível, pois $\mathbb{F}_q$ é isomorfo a $\mathbb{F}_p[x]/(f)$, onde $q = p^n$ e $f$ é um polinômio irredutível de grau $n$ sobre $\mathbb{F}_p$. Isto já explica o interesse, por parte de um grande número de pesquisadores, em questões envolvendo polinômios irredutíveis. Na verdade, há inúmeras outras aplicações, algumas das quais serão apresentadas nos próximos capítulos.

Na Seção 3.1, mostraremos um resultado fundamental sobre polinômios irredutíveis. O problema de determinar se um dado polinômio é irredutível não é trivial. Discutiremos um pouco sobre essa dificuldade na Seção 3.2, mostrando que testes eficientes de irredutibilidade existem.

A aritmética usando a representação polinomial depende da redução módulo um polinômio irredutível. Para esta tarefa, polinômios irredutíveis com muitos coeficientes iguais a zero são desejados. Discutiremos alguns resultados recentes sobre este assunto na Seção 3.3.

Um tipo importante de polinômio irredutível é o polinômio primitivo. Este tipo de polinômio tem a propriedade de que todas as suas raízes são elementos primitivos. Tais elementos, por sua vez, determinam uma outra maneira de representar os elementos num corpo finito, como vimos no Capítulo 2. Apresentaremos os polinômios primitivos e uma aplicação relacionada às seqüências representadas por LSFR's na Seção 3.4.

Na Seção 3.5, polinômios minimais serão abordados brevemente, desta vez especificamente sobre corpos finitos.

Qualquer polinômio não constante sobre um corpo é um produto de polinômios irredutíveis. Fatorar um polinômio significa obter tais polinômios irredutíveis. O problema de fatorar polinômios sobre um corpo finito é de extrema importância em criptografia, teoria de códigos e outra áreas. A Seção 3.6 é dedicada a alguns métodos clássicos de fatoração.

## 3.1 Um resultado fundamental

Começaremos com um resultado fundamental[1] sobre polinômios irredutíveis. Sua importância será justificada em várias ocasiões neste capítulo.

**Teorema 3.1.1.** *O produto de todos os polinômios mônicos irredutíveis sobre $\mathbb{F}_q$ cujos graus dividem $n$ é igual a $x^{q^n} - x$.*

*Demonstração.* O Lema 1.4.6 implica que $x^{q^n} - x$ seja um produto de polinômios irredutíveis distintos sobre $\mathbb{F}_q$. Seja $f$ um desses fatores irredutíveis de grau $m$. Segue que $\mathbb{F}_{q^n}$ contém uma raiz $\theta$ de $f$ e portanto $\mathbb{F}_q(\theta) \subseteq \mathbb{F}_{q^n}$. Portanto $[\mathbb{F}_q(\theta) : \mathbb{F}_q] = m$ divide $[\mathbb{F}_{q^n} : \mathbb{F}_q] = n$.

Reciprocamente, seja $f$ um polinômio irredutível sobre $\mathbb{F}_q$ de grau $m$ onde $m \mid n$. Se $\theta$ é uma raiz de $f$ no seu corpo de decomposição, pelos Teoremas 1.4.11 e 1.4.13, concluímos que $\mathbb{F}_q(\theta) \cong \mathbb{F}_{q^m} \subseteq \mathbb{F}_{q^n}$ e $\theta^{q^n} - \theta = 0$. Logo, pelo Teorema 1.3.12, obtemos que $f$ é o polinômio minimal de $\theta$ e assim divide $x^{q^n} - x$. □

Uma fórmula para o número de polinômios irredutíveis de um certo grau é a primeira conseqüência imediata deste resultado. Este número será dado em termos da função de Möbius; para maiores informações sobre esta função, veja Apêndice A.

**Teorema 3.1.2.** *O número de polinômios mônicos irredutíveis de grau $n$ sobre $\mathbb{F}_q$ é*

$$I_n = \frac{1}{n} \sum_{d|n} \mu(d) q^{n/d},$$

[1] Contaremos um pouco sobre a história deste resultado na Seção 3.6.2.

*onde $\mu$ é a função de Möbius.*

*Demonstração.* Comparando os graus dos polinômios no enunciado do Teorema 3.1.1, obtemos que $q^n = \sum_{d|n} dI_d$. Pela fórmula de inversão de Möbius (Teorema A.0.3), temos que $nI_n = \sum_{d|n} \mu(d) q^{n/d}$, o que dá a fórmula desejada. □

**Exemplo 3.1.3.** Há $q^{24}$ polinômios mônicos de grau 24 em $\mathbb{F}_q[x]$, enquanto o número de polinômios mônicos irredutíveis sobre $\mathbb{F}_q$ de grau 24 é

$$\begin{aligned} I_{24} &= \frac{1}{24}\left(\mu(1)q^{24} + \mu(2)q^{12} + \mu(3)q^8 + \mu(4)q^6 + \mu(6)q^4 \right. \\ &\quad \left. + \mu(8)q^3 + \mu(12)q^2 + \mu(24)q\right) \\ &= \frac{1}{24}\left(q^{24} - q^{12} - q^8 + q^4\right). \end{aligned}$$

Assim, a probabilidade de que um polinômio mônico de grau 24 seja irredutível sobre $\mathbb{F}_q$ é

$$\frac{I_{24}}{q^{24}} = \frac{1}{24}\left(1 - \frac{1}{q^{12}} - \frac{1}{q^{16}} + \frac{1}{q^{20}}\right) \approx \frac{1}{24}.$$

Em geral, se o grau do polinômio é $n$, esta probabilidade é aproximadamente $1/n$. Isso dá um algoritmo probabilístico para encontrar polinômios irredutíveis, como veremos a seguir.

## 3.2 Um teste de irredutibilidade

Uma outra conseqüência do Teorema 3.1.1 é que ele pode ser usado para testar a irredutibilidade de um polinômio sobre $\mathbb{F}_q$ de uma maneira bastante simples. O primeiro passo do processo consiste em calcular o $\mathrm{mdc}(f, x^q - x)$. Se esse mdc não é 1, então $f$ tem fatores lineares e portanto é redutível. Caso contrário, calculamos o $\mathrm{mdc}(f, x^{q^2} - x)$ para verificar se $f$ tem fatores irredutíveis cujos graus dividem 2. Na verdade, como $f$ não tem nenhum fator linear, estaremos testando se $f$ tem algum fator irredutível de grau exatamente igual a 2. Continuamos calculando o $\mathrm{mdc}(f, x^{q^i} - x)$, enquanto

$i \leq \text{grau}(f)/2$ e o mdc não é 1. Essas idéias determinam o seguinte algoritmo, que também é conhecido como o Algoritmo de Ben-Or [6].

**Algoritmo 3.2.1.** [Ben-Or]
**Entrada:** $f \in \mathbb{F}_q[x]$ de grau $n$.
**Saída:** "$f$ irredutível" ou "$f$ redutível".

```
para i = 1 até ⌊n/2⌋ faça
   se mdc(f, x^{q^i} − x) ≠ 1 então
      { retorne "f redutível" e pare; }
retorne "f irredutível";
```

**Exemplo 3.2.2.**

1. Como $\text{mdc}(x^{15}-1, x^{11}-x) = x^5 - 1$, o polinômio $x^{15}-1$ é redutível sobre $\mathbb{F}_{11}$. Em particular, esse mdc mostra que $x^{15}-1$ tem cinco fatores lineares, a saber, $x-1$, $x-3$, $x-4$, $x-5$ e $x-9$.

2. Seja $f(x) = x^6 + 10x^5 + x^4 + 14x^3 + 13x^2 + x + 3$ em $\mathbb{F}_{17}[x]$. Temos que $\text{mdc}(x^{17}-x, f) = 1$ e $\text{mdc}(x^{17^2}-x, f) = 1$; portanto $f$ não tem fatores lineares, nem quadráticos. No entanto, temos que $\text{mdc}(x^{17^3}-x, f) = f$, o que mostra que $f$ não só é redutível sobre $\mathbb{F}_{17}$, mas também é o produto de dois fatores irredutíveis de grau 3. De fato, $f(x) = (x^3 + x + 3)(x^3 + 10x^2 + 1)$.

Analisaremos agora o tempo de execução do algoritmo acima no pior caso. Primeiro vamos analisar como podemos calcular $x^q \pmod f$. Usando o método da repetição de quadrados (veja Seção 2.1.4), podemos obter essa potência usando no máximo $2\lfloor \log q \rfloor$ produtos, ao invés de $q-1$ produtos como no método tradicional. Para calcular $x^{q^{i+1}} \pmod f$, tendo calculado $x^{q^i} \pmod f$, observamos que

$$(x^{q^i})^q = x^{q^i} x^{q^i} \cdots x^{q^i} = x^{qq^i} = x^{q^{i+1}}.$$

Ao invés de calcularmos $\text{mdc}(f, x^{q^i} - x)$, primeiro reduzimos $x^{q^i} - x \pmod f$, já que

$$\text{mdc}(f, x^{q^i} - x) = \text{mdc}(f, x^{q^i} - x \pmod f).$$

Isso representa um bom ganho no custo do algoritmo, pois o último mdc envolve polinômios de grau no máximo $n$.

No pior caso, este algoritmo calcula $\lfloor n/2 \rfloor$ $q$-ésimas potências e um mdc de polinômios de grau no máximo $n$. Pelos dados na Seção 2.1, concluímos que o custo é $O(n^2 \log^2 n \log\log n \log q)$, usando multiplicação baseada no FFT, e de $O(n^3 \log q)$, usando aritmética clássica. Entretanto, na média, o processo precisa apenas do valor esperado do menor grau entre os fatores irredutíveis de $f$. Como o menor grau esperado é da ordem de $\log n$ (veja [108, 109]), o custo esperado é apenas $O(n \log^3 n \log\log n \log q)$ e $O(n^2 \log n \log q)$, respectivamente.

## 3.3 Polinômios irredutíveis de baixo peso

O número de coeficientes não nulos num polinômio é chamado o *peso do polinômio*. O problema de caracterizar polinômios irredutíveis de peso baixo tem atraído vários pesquisadores nos últimos anos. Já mencionamos que os elementos de um corpo finito podem ser representados por meio de polinômios com operações módulo um polinômio irredutível. Acontece que se esse polinômio tem peso baixo, essas operações serão mais baratas. Por exemplo, vimos na Seção 2.1.1 que a divisão de um polinômio $f$ de grau $n$ por um polinômio $g$ de grau $m \leq n$ tem um custo de $m(n-m)$ operações no corpo finito, ou seja, $O(n^2)$. Porém, se o polinômio $g$ tem peso $w(g)$, então o custo é de $w(g)(n - w(g))$. Assim, se $w(g)$ é constante, o custo desta divisão é somente $O(n)$.

A maioria dos resultados existentes se concentram em $\mathbb{F}_2$, dado que provavelmente a maioria das aplicações envolvendo corpos finitos usa $\mathbb{F}_2$. Neste caso, uma observação imediata é a de que o peso de um polinômio irredutível sobre $\mathbb{F}_2$ é sempre ímpar; caso contrário, o polinômio $x+1$ seria sempre um de seus fatores.

Nesta direção, um resultado bastante conhecido é devido a Swan [129]. Ele caracteriza a paridade do número de fatores irredutíveis de qualquer trinômio da forma $f(x) = x^n + x^k + 1$ sobre $\mathbb{F}_2$. Por exemplo, ele mostra que $x^n + x^k + 1$, onde $n$ é um múltiplo de 8, sempre tem um número par de fatores irredutíveis e portanto não pode ser irredutível para qualquer $n$. Os resultados de Swan são baseados no teorema de Stickelberger [126].

Resultados experimentais como os de Seroussi [121] mostram que trinômios irredutíveis sobre $\mathbb{F}_2$ existem somente para aproximadamente 50% dos graus. Os próximos polinômios irredutíveis de menor peso são os de peso igual a cinco, também conhecidos como pentanômios. O uso de pentanômios, quando trinômios não existem, é convencional de acordo com as especificações do IEEE[2] para criptografia de chave pública [73]. Existe evidência empírica de que pentanômios irredutíveis de qualquer grau sobre $\mathbb{F}_2$ existem, mas ainda não há uma prova disso.

O estudo do número de fatores irredutíveis de tetranômios sobre $\mathbb{F}_2$, que são polinômios de peso igual a quatro, foi recentemente completado por Hales e Newhart [64]; veja também o artigo de Bluher [14].

Outros polinômios irredutíveis de peso baixo que têm recebido atenção são $x^n + g(x) \in \mathbb{F}_2[x]$, onde $\text{grau}(g) \leq \log_2 n$. Comentaremos sobre esse tipo de polinômios quando apresentarmos o algoritmo de Coppersmith para calcular logaritmos discretos na Seção 4.4.4. Eles também aparecem na geração de elementos de ordem grande em corpos finitos [46]. Não se sabe muito sobre polinômios irredutíveis desta forma (veja uma breve discussão na Seção 4.4.4).

Finalmente, citamos um sistema criptográfico, que foi recentemente desenvolvido por Aumasson, Finiasz, Meier e Vaudenay, chamado *TCHo*[3]. A segurança deste sistema é garantida pela dificuldade de se encontrar um múltiplo de um polinômio irredutível de baixo peso; veja mais detalhes em [5].

## 3.4 Polinômios primitivos

Polinômios primitivos formam uma classe particular de polinômios irredutíveis.

**Definição 3.4.1.** Um polinômio $f \in \mathbb{F}_q[x]$ de grau $m$ é *primitivo*, se $f$ é o polinômio minimal sobre $\mathbb{F}_q$ de algum elemento primitivo em $\mathbb{F}_{q^m}$.

[2]Institute of Electrical and Electronics Engineers.
[3]Trapdoor Cipher, Hardware oriented.

Em outras palavras, um polinômio primitivo de grau $m$ é um polinômio mônico e irredutível com a propriedade adicional de que se $\alpha \in \mathbb{F}_{q^m}$ é uma raiz de $f$ então a ordem de $\alpha$ é $q^m - 1$. Portanto, as raízes de um polinômio primitivo são geradores do grupo multiplicativo $\mathbb{F}_{q^m}^*$.

Na Seção 2.2, vimos que o número de elementos primitivos em $\mathbb{F}_{p^m}$ é $\phi(p^m - 1)$, onde $\phi$ é a função de Euler (veja Apêndice C). O número de polinômios primitivos em $\mathbb{F}_{p^m}[x]$ é $\phi(p^m - 1)/m$. Isto porque se $f$ é um polinômio primitivo e $\alpha$ é uma de suas raízes, então todos os conjugados de $\alpha$ têm o mesmo polinômio primitivo $f$ (veja [90, Teorema 2.14]).

Hansen e Mullen [68] conjecturaram que, com exceção de alguns poucos valores pequenos de $q$ e de $n$, dado $a \in \mathbb{F}_q$ e $j$ com $1 \leq j \leq n-1$, sempre existe pelo menos um polinômio primitivo de grau $n$ sobre $\mathbb{F}_q$, onde $a$ é o coeficiente de $x^j$. Esta conjectura foi responsável por intensa pesquisa nos últimos anos. Vários resultados parciais surgiram até que uma série de artigos de Cohen e Han finalmente completou a prova; veja [23, 24, 25, 26, 39, 65, 76, 99].

Na prática, estamos interessados em polinômios primitivos sobre $\mathbb{F}_2$ com baixo peso, seguindo a mesma linha de idéias já discutidas na Seção 3.3. Neste caso, a melhor opção tem sido usar trinômios primitivos. Não é difícil ver que se o trinômio $f(x) = x^r + x^s + 1$, $0 < s < r$, é primitivo, então $x^r + x^{r-s} + 1 = x^r f(x^{-1})$ também é primitivo. Logo, podemos supor que $s < r/2$. Trinômios primitivos de grau grande são necessários em aplicações como a geração de números aleatórios. No entanto, o problema de encontrar tais polinômios é difícil. Um grande avanço foi obtido com o algoritmo para testar a irredutibilidade de trinômios de grau ímpar sobre $\mathbb{F}_2$, desenvolvido por Brent et al. [15, 16, 17]. Usando o fato de que todo trinômio irredutível de grau $r$, onde $r$ é um expoente de Mersenne[4], é necessariamente primitivo, Brent et al. encontraram os seguintes trinômios primitivos: $x^{3021377} + x^{361604} + 1$ e $x^{3021377} + x^{1010202} + 1$.

[4]Se $2^r - 1$ é primo, dizemos que $2^r - 1$ é um *primo de Mersenne* e $r$ é *um expoente de Mersenne*.

### 3.4.1 Uma aplicação: LFSR

As seqüências em corpos finitos são usadas numa variedade de aplicações, nas áreas de engenharia e criptografia. Por exemplo, elas aparecem na emissão e na comunicação digital, na construção de geradores de números pseudo-aleatórios e nas cifras de fluxo. Na Seção 4.5.2, veremos esta última aplicação.

Normalmente, representamos uma seqüência usando um *registrador de deslocamento com realimentação linear*. Neste livro, denotaremos este registrador por LFSR, que é a abreviação das iniciais em inglês (Linear Feedback Shift Register). Cada registrador do LFSR contém um valor em $\mathbb{F}_q$. Chamamos de *estado* do LFSR a $n$-upla de valores contida no LFSR. Há então $q^n$ estados no LFSR. Denotaremos os estados do LFSR por vetores em $\mathbb{F}_q^n$, onde $n$ é o número de registradores do LFSR.

O LFSR muda de estado através de uma função linear. A saída do LFSR é usada como realimentação do LFSR para gerar o estado seguinte. O estado inicial $\overrightarrow{\mathbf{S_0}}$ do LFSR é chamado a *semente do registrador*. O LFSR resulta deterministicamente num elemento em $\mathbb{F}_q$ de cada vez. Logo, a seqüência de valores do LFSR é completamente determinada pela sua semente.

Na maioria das aplicações práticas, o corpo finito usado é $\mathbb{F}_2$ e o LFSR computa assim um bit. Porém, nesta seção, consideraremos o caso mais geral de um corpo finito de ordem arbitrária.

LFSR's são usados para calcular seqüências que satisfazem uma recorrência linear. Dizemos que uma seqüência $s_0, s_1, s_2, \ldots$ em $\mathbb{F}_q$ satisfaz uma *recorrência linear (homogênea) de ordem* $n$ se

$$s_{k+n} = a_{n-1}s_{k+n-1} + a_{n-2}s_{k+n-2} + \cdots + a_0 s_k, \quad \text{para} \quad k = 0, 1, \ldots \tag{3.1}$$

Para cada seqüência, existe um LFSR que a computa.

**Exemplo 3.4.2.** Seja a seqüência $\{s_k\}$ em $\mathbb{F}_2$ definida por

$$s_{k+4} = s_{k+1} + s_k, \quad k \geq 0, \quad s_0 = s_1 = s_2 = 0, \; s_3 = 1.$$

O LFSR correspondente tem 4 registradores. Sejam $x_0, x_1, x_2$ e $x_3$ os valores atuais dos 4 registradores do LFSR, onde $x_0$ é o valor mais à direita e $x_3$ é o valor mais à esquerda do LFSR. Então o estado

atual do LFSR é o vetor $(x_3, x_2, x_1, x_0)$. É claro que a função linear $A$ correspondente é $x_0 + x_1$ e a mudança de estado é dada por $(x_3, x_2, x_1, x_0) \mapsto (x_0+x_1, x_3, x_2, x_1)$. A semente é o vetor consistindo dos valores iniciais da seqüência, ou seja, $\overrightarrow{\mathbf{S_0}} = (s_3, s_2, s_1, s_0) = (1, 0, 0, 0)$; veja a Figura 3.1.

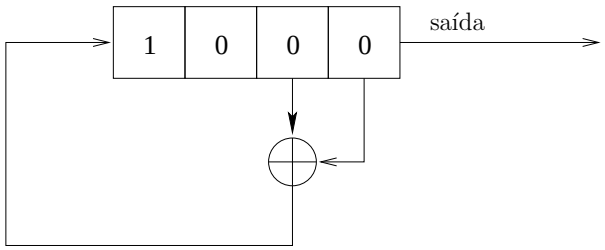



Figura 3.1: LFSR de $s_{k+4} = s_{k+1} + s_k$.

A saída de cada vez é o valor do último registrador, portanto 0 sai primeiro. Em seguida, aplica-se a função $A$ ao vetor $(1, 0, 0, 0)$ para obter $(0, 1, 0, 0)$ nos registradores. A segunda saída é então 0 de novo. No próximo passo, os registradores têm $(0, 0, 1, 0)$ e sai 0. Mais uma aplicação de $A$ produz $(1, 0, 0, 1)$ nos registradores e uma saída de 1. Continuando assim, vemos que os primeiros 15 termos da seqüência são 000100110101111.

A cada LFSR, ou seja, a cada seqüência satisfazendo uma recorrência da forma (3.1), associamos um polinômio em $\mathbb{F}_q[x]$, chamado o *polinômio característico* do LFSR da seguinte maneira. Se o comprimento (ou o número de registradores) do LFSR é $n$, o polinômio característico tem grau $n$. Além disso, seus coeficientes em $\mathbb{F}_q$ preenchem as portas lógicas do LFSR de tal maneira que se (3.1) é satisfeita, então

$$f(x) = x^n - a_{n-1}x^{n-1} - \cdots - a_1 x - a_0 \tag{3.2}$$

é o polinômio característico do LFSR. Reciprocamente, dado o polinômio (3.2), podemos construir o LFSR correspondente. A função linear

associada é dada pela matriz

$$A = \begin{pmatrix} a_{n-1} & a_{n-2} & \cdots & a_1 & a_0 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & 0 & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix},$$

que é essencialmente a matriz companheira de $f$. Se $\overrightarrow{\mathbf{S_0}}$ é a semente, então o próximo estado é $\overrightarrow{\mathbf{S_1}} = A\overrightarrow{\mathbf{S_0}}$. Em geral, temos que $\overrightarrow{\mathbf{S_k}} = A\overrightarrow{\mathbf{S_{k-1}}} = A^k\overrightarrow{\mathbf{S_0}}$. Portanto, após $k$ passos, o LFSR terá $\overrightarrow{\mathbf{S_k}}$ nos seus registradores. A $k$-ésima saída será a última entrada de $\overrightarrow{\mathbf{S_k}}$.

**Exemplo 3.4.3.** Consideramos a seqüência do Exemplo 3.4.2. O polinômio característico é $x^4 + x + 1$, a matriz companheira é

$$A = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

e a semente é $\overrightarrow{\mathbf{S_0}} = (1, 0, 0, 0)$.

O LFSR tem um número finito de estados, pois se o LFSR tem $n$ registradores, então tem no máximo $q^n$ estados. Assim, deve repetir um estado e depois daquele momento, a seqüência de saídas será repetida. O comprimento do ciclo é chamado o *período* do LFSR. Nas aplicações práticas, estamos interessados nos LFSR's com períodos maximais. *Ciclos LFSR's maximais* visitam todos os possíveis $q^n - 1$ estados do registrador de deslocamentos com a exceção do estado em que todas as entradas dos registradores são nulas. A propriedade mais importante de um LFSR é dada pelo próximo teorema.

**Teorema 3.4.4.** *Uma condição necessária e suficiente para a seqüência gerada por um LFSR, com semente não nula, ter comprimento maximal é que seu polinômio característico seja primitivo.*

Uma demonstração deste resultado pode ser encontrada no livro de Golomb e Gong [62, Seção 4.3]. O LFSR do Exemplo 3.4.2 tem comprimento maximal.

**Exemplo 3.4.5.** Considere o LFSR da Figura 3.1. O polinômio característico é $f(x) = x^4 + x + 1$. Como $f$ é um polinômio primitivo, o LFSR tem comprimento maximal. Como já vimos, se usarmos a semente 1000, a seqüência de período maximal 15 é 000100110101111. Se usarmos a semente 1111, por exemplo, a seqüência gerada será 111100010011010 (exercício). De fato, cada uma das 15 possíveis sementes não nulas gera uma seqüência de período maximal 15. A semente 0000 gera a seqüência 00000000000000, que não é interessante.

A literatura sobre LFSR é bem vasta; veja, por exemplo, Golomb [61], Berlekamp [8], Lidl e Niederreiter [89, Capítulo 8] e Jungnickel [75, Capítulo 6].

## 3.5 Polinômios minimais

Seja $\alpha \in \mathbb{F}_q$. Pelo Lema 1.4.4, temos que $\alpha^q = \alpha$, ou seja, $\alpha$ satisfaz a equação $x^q - x = 0$. Assim, $\alpha$ é algébrico sobre $\mathbb{F}_p$ e portanto tem um polinômio minimal $M \in \mathbb{F}_p[x]$.

**Exemplo 3.5.1.** Seja $q = 2^3$, $p = 2$. Então $\mathbb{F}_{2^3} \cong \mathbb{F}_2[x]/(x^3+x+1)$. Seja $\alpha = x + (x^3 + x + 1)$. Temos a sequinte tabela:

| elemento | polinômio minimal |
|---|---|
| 0 | $x$ |
| 1 | $x+1$ |
| $\alpha$ | $x^3+x+1$ |
| $\alpha^2$ | $x^3+x+1$ |
| $\alpha^3$ | $x^3+x^2+1$ |
| $\alpha^4$ | $x^3+x+1$ |
| $\alpha^5$ | $x^3+x^2+1$ |
| $\alpha^6$ | $x^3+x^2+1$ |

Observamos que, quando $k \equiv 2m \pmod 7$, os elementos $\alpha^m$ e $\alpha^k$ têm o mesmo polinômio minimal. Veremos que isto não é uma coincidência.

Polinômios minimais não são interessantes apenas por si só, mas também por causa da sua relação com códigos cíclicos, por exemplo;

veja Seção 5.2. Apresentaremos aqui algumas conseqüências imediatas do Teorema 1.3.12.

**Proposição 3.5.2.** *Seja $\alpha \in \mathbb{F}_{p^m}$ com polinômio minimal $M$ sobre $\mathbb{F}_p$. Então, temos que*

1. *$M$ é irredutível;*
2. *se $f \in \mathbb{F}_p[x]$ com $f(\alpha) = 0$, então $M \mid f$;*
3. *$M \mid (x^{p^m} - x)$;*
4. *$\text{grau}(M) \leq m$;*
5. *se $\alpha$ é um elemento primitivo de $\mathbb{F}_{p^m}$, então $\text{grau}(M) = m$ e $M$ é um polinômio primitivo.*

A seguir, provamos um lema básico na teoria de Galois que será necessário para mostrar um resultado sobre polinômios minimais.

**Lema 3.5.3.** *Sejam $K$ uma extensão de um corpo $F$, $\theta : K \to K$ um isomorfismo de anéis fixando $F$ e $f \in F[x]$. Suponhamos que $\alpha \in K$ seja uma raiz de $f$. Então $\theta(\alpha)$ também é uma raiz de $f$.*

*Demonstração.* Seja $f(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_0$ com $a_i \in F$ para $i = 1, \ldots, n$. Como $f(\alpha) = 0$ e $\theta(0) = 0$, temos que

$$\begin{aligned} 0 &= \theta(f(\alpha)) \\ &= \theta(a_n\alpha^n + a_{n-1}\alpha^{n-1} + \cdots + a_0) \\ &= \theta(a_n)\theta(\alpha)^n + \theta(a_{n-1})\theta(\alpha)^{n-1} + \cdots + \theta(a_0) \\ &= a_n\theta(\alpha)^n + a_{n-1}\theta(\alpha)^{n-1} + \cdots + a_0 \\ &= f(\theta(\alpha)), \end{aligned}$$

de onde obtemos que $\theta(\alpha)$ é uma raiz de $f$. □

Caracterizamos agora os elementos de $\mathbb{F}_q$ com o mesmo polinômio minimal.

**Proposição 3.5.4.** *Sejam $\alpha \in \mathbb{F}_q$ e $p = \text{car}(\mathbb{F}_q)$. Então, $\alpha$ e $\alpha^p$ têm o mesmo polinômio minimal sobre $\mathbb{F}_p$.*

*Demonstração.* Seja $M$ o polinômio minimal de $\alpha$ sobre $\mathbb{F}_p$. No Corolário 1.4.8, vimos que a aplicação $\theta : \mathbb{F}_q \to \mathbb{F}_q$ dada por $\theta(a) = a^p$ é um isomorfismo de anéis fixando $\mathbb{F}_p$. Resulta do Lema 3.5.3 que $\alpha^p$ é uma raiz de $M$. Como $M$ é irredutível, Teorema 1.3.12 implica que $M$ seja o polinômio minimal de $\alpha^p$. □

Agora sabemos porque $\alpha$, $\alpha^2$ e $\alpha^4$ têm o mesmo polinômio minimal no Exemplo 3.5.1. Mais geralmente, elementos conjugados têm o mesmo polinômio minimal.

**Exemplo 3.5.5.** Consideramos $\alpha \in \mathbb{F}_{2^4}$. Assim,

$$\alpha, \alpha^2, (\alpha^2)^2 = \alpha^4, (\alpha^4)^2 = \alpha^8, (\alpha^8)^2 = \alpha^{16} = \alpha.$$

Todos esses elementos têm o mesmo polinômio minimal. O mesmo acontece com os elementos

$$\alpha^3, (\alpha^3)^2 = \alpha^6, (\alpha^6)^2 = \alpha^{12}, (\alpha^{12})^2 = \alpha^{24} = \alpha^9, (\alpha^9)^2 = \alpha^{18} = \alpha^3.$$

O exemplo acima sugere que podemos fazer uma partição de $\mathbb{F}_{p^m}$ em blocos cujos elementos têm o mesmo polinômio minimal, a partir da iteração do automorfismo $\alpha \mapsto \alpha^p$. Sejamos mais precisos. Se $q = p^m$, então sabemos que o grupo cíclico $\mathbb{F}_q^*$ é isomorfo ao grupo aditivo $\mathbb{Z}_{p^m-1}$. A exponenciação com expoente $p$ em $\mathbb{F}_q$ corresponde à multiplicação por $p$ em $\mathbb{Z}_{p^m-1}$. Como $p$ e $p^m - 1$ são relativamente primos, a aplicação $s \mapsto ps$ é uma permutação de $\mathbb{Z}_{p^m-1}$. Podemos considerar então a decomposição de $\mathbb{Z}_{p^m-1}$ como sendo formada pelas órbitas da permutação.

**Definição 3.5.6.** A operação de multiplicar por $p$ particiona $\mathbb{Z}_{p^m-1}$ em conjuntos chamados as *classes laterais ciclotômicas módulo* $p^m-1$. A classe lateral ciclotômica de $s$ é

$$\{s, ps, p^2s, \ldots, p^{m_s-1}s\},$$

onde $m_s$ é o menor inteiro positivo tal que $p^{m_s}s \equiv s \pmod{p^m - 1}$.

**Exemplo 3.5.7.** Para $p = 2$, $m = 3$ e $p^m - 1 = 7$, as classes laterais ciclotômicas módulo 7 são

$$C_0 = \{0\},\ C_1 = \{1, 2, 4\},\ C_3 = \{3, 6, 5\}.$$

Estas classes correspondem aos expoentes de $\alpha$ que têm o mesmo polinômio minimal no Exemplo 3.5.1.

Usamos a seguinte convenção: a classe lateral será denotada por $C_s$, onde $s$ é o menor inteiro na classe.

Fixamos um elemento primitivo $\beta \in \mathbb{F}_{p^m}$. Pelos comentários acima, todos os elementos $\beta^i$ com $i \in C_s$ têm o mesmo polinômio minimal. Denotaremos por $M^i$ o polinômio minimal de $\beta^i$ sobre $\mathbb{F}_p$. Segue imediatamente que se $i$ pertence a classe lateral $C_s$ então

$$\prod_{j \in C_s} (x - \beta^j) \mid M^{(i)}(x).$$

É possível demonstrar que os elementos $\beta^j$ com $j \in C_s$ são todos os conjugados de $\beta^i$. Isto é uma conseqüência imediata do fato de que o grupo de Galois de $\mathbb{F}_{p^m}$ sobre $\mathbb{F}_p$ é um grupo cíclico de ordem $m$ gerado pelo automorfismo $\alpha \mapsto \alpha^p$.

**Teorema 3.5.8.** *Sejam $q = p^m$ e $\beta$ um elemento primitivo em $\mathbb{F}_q$. Então, para $0 \leq s \leq p^m - 1$ e $i \in C_s$, temos que*

$$M^{(i)}(x) = \prod_{j \in C_s} (x - \beta^j).$$

Além disso, como

$$x^{q-1} - 1 = \prod_{\alpha \in \mathbb{F}_q^*} (x - \alpha) = \prod_{0 \leq j \leq q-1} (x - \beta^j),$$

temos que

$$x^{q-1} - 1 = \prod_s M^{(s)}(x),$$

onde $s$ varia sobre os representantes das classes laterais ciclotômicas modulo $p^m - 1$.

Usaremos os resultados acima sobre polinômios minimais na Seção 5.2.

## 3.6 Fatoração de polinômios

Seja $f$ um polinômio em $\mathbb{F}_q[x]$. Fatorar $f$ significa encontrar $a \in \mathbb{F}_q$ e polinômios mônicos irredutíveis $f_1, \ldots, f_r$, dois a dois distintos, tais que $f = af_1^{e_1} \cdots f_r^{e_r}$, onde cada $e_i$ é um inteiro positivo[5]. Pelo Teorema 1.3.5, tal fatoração é única a menos da ordem dos fatores. Definimos a *parte de $f$ livre de quadrados* como sendo $f_1 \cdots f_r$. Nesta seção, vamos supor sem perda de generalidade que $f$ seja mônico.

A fatoração de polinômios é um requisito essencial em muitas aplicações na teoria de códigos, álgebra computacional, criptografia, teoria computacional de números e várias outras áreas (veja referências em [58], por exemplo). No Capítulo 4, teremos a oportunidade de ver uma dessas aplicações mais detidamente. Apesar do esforço de vários pesquisadores, fatorar polinômios sobre um corpo finito de maneira efetiva ainda é um problema em aberto. Alguns surveys recentes são os de Kaltofen [77, 78], o Capítulo 14 de von zur Gathen e Gerhard [56], e o de von zur Gathen e Panario [58].

Em seguida, mostraremos um método geral para fatorar polinômios sobre um corpo finito, que consiste de três estágios: eliminação de fatores repetidos (ERF), fatoração em graus distintos (DDF) e fatoração de graus iguais (EDF)[6]. Antes de discutirmos como cada um desses estágios funciona, exibimos um diagrama do método geral através da Figura 3.2. A Figura 3.3 ilustra um exemplo de um polinômio sendo fatorado usando o método descrito pela Figura 3.2.

### 3.6.1 Eliminação de fatores repetidos (ERF)

O primeiro estágio, no método de fatoração que vamos descrever, consiste em eliminar fatores repetidos. Uma vantagem desta redução é que o polinômio resultante freqüentemente tem grau menor, e ao mesmo tempo contém todos os fatores irredutíveis do polinômio original. Além disso, alguns métodos requerem que o dado polinômio não

[5] Chamaremos esta decomposição como sendo a *fatoração completa* de $f$; enquanto que qualquer outra decomposição é apenas uma *fatoração parcial* de $f$.

[6] Estaremos usando as iniciais dos estágios em inglês.

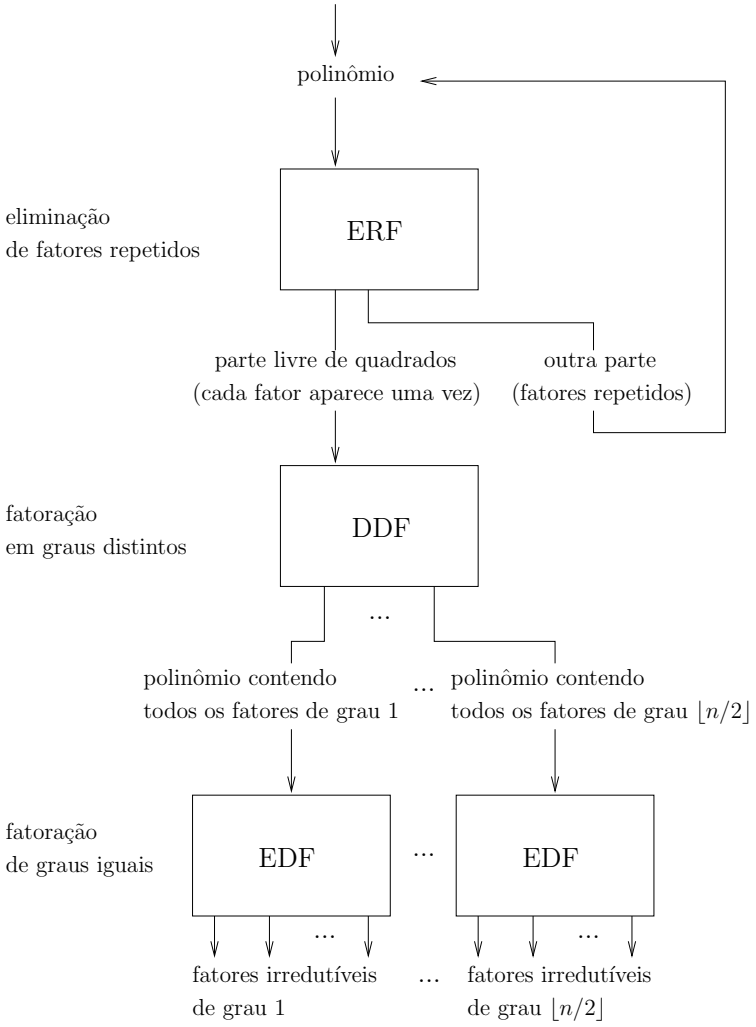



Figura 3.2: Método geral da fatoração de um polinômio através da ERF, DDF e EDF.

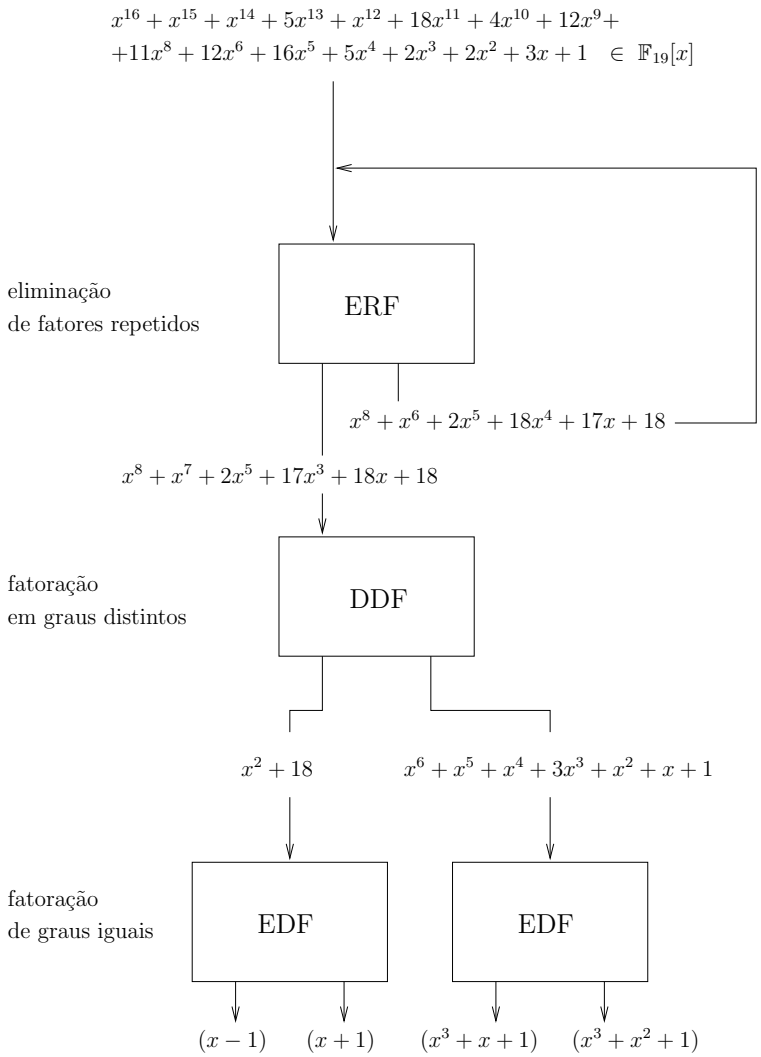



Figura 3.3: Exemplo de um polinômio sendo fatorado através da ERF, DDF e EDF.

tenha fatores repetidos e assim este estágio é também de interesse independente.

A fim de introduzir a idéia básica, vamos considerar o polinômio particular $f = AB^3$, onde $A$ é o produto dos fatores irredutíveis de $f$ que aparecem somente uma vez e $B$ é o produto dos fatores irredutíveis de $f$ que aparecem três vezes. Como $f' = A'B^3 + 3AB^2B' = B^2(A'B + 3AB')$, temos que $B^2$ aparece na decomposição de ambos $f$ e $f'$, ou seja, $B^2$ é um divisor do $\mathrm{mdc}(f, f')$. Assim, o polinômio $f/\mathrm{mdc}(f, f') = AB$ não contém fatores repetidos. Esta idéia é válida em todos os corpos; contudo, em corpos finitos, onde a característica é diferente de zero, deve-se ter um certo cuidado. Por exemplo, o polinômio $f = x^{10}$ em $\mathbb{F}_5[x]$ revela que é possível ter $f' = 0$, mesmo $f$ não sendo uma constante. Em geral, se $p = \mathrm{car}(\mathbb{F}_q)$ e $f = \sum_{i=0}^{n/p} f_{ip}x^{ip}$, temos que

$$f' = \sum_{i=1}^{n/p} ipf_{ip}x^{ip-1} = 0,$$

e assim $f/\mathrm{mdc}(f, f') = 1$. No entanto, o polinômio $f$ contém fatores repetidos, pois usando Teorema 1.4.7 podemos escrever $f$ como

$$f = \left(\sum_{i=0}^{n/p} f_{ip}^{q/p}x^i\right)^p.$$

Se $f = \prod_{i=1}^{r} f_i^{e_i}$ é a fatoração de $f$ em $\mathbb{F}_q[x]$, segue que

$$f' = \sum_{i=1}^{r} e_i f_i^{e_i-1} f_i' \prod_{j\neq i} f_j^{e_j} = \sum_{i=1}^{r} e_i f_i^{e_i-1} f_i' \frac{f}{f_i^{e_i}} = \sum_{i=1}^{r} e_i f_i' \frac{f}{f_i}$$

e $\mathrm{mdc}(f_i, f_i') = 1$, pois $f_i$ é irredutível. Observamos que $f_i' = 0$ implica que $f_i$ seja uma $p$-ésima potência, contradizendo a irredutibilidade de $f_i$. Por outro lado, $\mathrm{grau}(f_i') < \mathrm{grau}(f_i)$. Novamente, a irredutibilidade de $f_i$ implica que $\mathrm{mdc}(f_i, f_i') = 1$. Assim,

$$u = \mathrm{mdc}(f, f') = \mathrm{mdc}\left(\prod_{i=1}^{r} f_i^{e_i}, \sum_{i=1}^{r} e_i f_i' \frac{f}{f_i}\right) = \prod_{\substack{i=1 \\ p\nmid e_i}}^{r} f_i^{e_i-1} \prod_{\substack{i=1 \\ p\mid e_i}}^{r} f_i^{e_i}.$$

Obtemos agora a parte livre de quadrados de $f$ correspondente aos fatores cujos expoentes não são múltiplos de $p$:

$$v = \frac{f}{\mathrm{mdc}(f, f')} = \frac{f}{u} = \prod_{\substack{i=1 \\ p \nmid e_i}}^{r} f_i.$$

Para obtermos os outros fatores, suponhamos que $n = \mathrm{grau}(f)$ e calculamos

$$v^n = \prod_{\substack{i=1 \\ p \nmid e_i}}^{r} f_i^n.$$

Segue que

$$\mathrm{mdc}(u, v^n) = \mathrm{mdc}\left( \prod_{\substack{i=1 \\ p \nmid e_i}}^{r} f_i^{e_i - 1} \prod_{\substack{i=1 \\ p | e_i}}^{r} f_i^{e_i}, \prod_{\substack{i=1 \\ p \nmid e_i}}^{r} f_i^n \right) = \prod_{\substack{i=1 \\ p \nmid e_i}}^{r} f_i^{e_i - 1},$$

e portanto

$$w = \frac{u}{\mathrm{mdc}(u, v^n)} = \prod_{\substack{i=1 \\ p | e_i}}^{r} f_i^{e_i}.$$

Podemos chamar o algoritmo recursivamente usando $w^{1/p}$ até obtermos $\prod_{i=1}^{r} f_i$.

O próximo algoritmo descreve os passos do método que acabamos de descrever.

**Algoritmo 3.6.1.** [Eliminação de fatores repetidos (ERF)]
**Procedimento** ERF($f$)
**Entrada**: $f \in \mathbb{F}_q[x]$ mônico de grau $n$.
**Saída**: a parte de $f$ livre de quadrados.

$p = \mathrm{car}(\mathbb{F}_q)$;
$u = \mathrm{mdc}(f, f')$;
`se` $u = 1$ `então retorne` $f$;
$v = \dfrac{f}{u}$;

$w = \dfrac{u}{\mathrm{mdc}(u, v^n)}$;
$z = w^{1/p}$;
`retorne` $v.\mathrm{ERF}(z)$.

Mostramos agora que podemos obter mais informação sobre os fatores irredutíveis de $f$, além do produto de fatores distintos de $f$. A *fatoração de $f$ livre de quadrados* é formada por polinômios mônicos $g_1, \ldots, g_n \in \mathbb{F}_q[x]$, cada um dos quais é livre de quadrados, dois a dois relativamente primos, satisfazendo $f = g_1 g_2^2 \ldots g_n^n$. Cada $g_i$ é o produto de todos os fatores irredutíveis que aparecem repetidos $i$ vezes na fatoração de $f$. Isso pode ser feito usando o algoritmo de Yun [137], que será descrito a seguir.

**Algoritmo 3.6.2.** [Fatoração livre de quadrados (SFF)]
**Procedimento** SSF($f$)
**Entrada**: $f \in \mathbb{F}_q[x]$ mônico.
**Saída**: a fatoração de $f$ livre de quadrados, i.e. $f = g_1 g_2^2 \ldots g_n^n$ com cada $g_i$ mônico, livre de quadrados e dois a dois relativamente primos.

$p = \mathrm{car}(\mathbb{F}_q)$;
$i \leftarrow 1$; $S \leftarrow 1$; $g \leftarrow f'$;
`se` $g \neq 0$ `então {`
  $h \leftarrow \mathrm{mdc}(f, g)$;
  $r \leftarrow f/h$;
  `enquanto` $r \neq 1$ `faça {`
    $s \leftarrow \mathrm{mdc}(r, h)$; $t \leftarrow r/s$;
    $S \leftarrow t^i S$; $i \leftarrow i + 1$;
    $r \leftarrow s$; $h \leftarrow h/s$ `};`
  `se` $h \neq 1$ `então {`
    $h \leftarrow h^{1/p}$;
    $S \leftarrow (\mathrm{SFF}(h))^p S$`}}`
`senão {`
  $f \leftarrow f^{1/p}$;
  $S \leftarrow (\mathrm{SFF}(f))^p$`};`
`retorne(`$S$`).`

Observamos que, no algoritmo SFF, a saída do algoritmo deve ser interpretada de maneira simbólica. Acontece que as potências dos

fatores que aparecem nas chamadas recursivas (quando temos fatores que aparecem um número de vezes múltiplo da característica) devem ser combinadas para assim obtermos a potência final deste fator. No próximo exemplo, ilustraremos esta situação.

**Exemplo 3.6.3.** Suponhamos que $f = AB^2C^3D^9$ sobre $\mathbb{F}_3$, onde $A, B, C$ e $D$ são o produto de todos os fatores irredutíveis de multiplicidades $1, 2, 3$ e $9$, respectivamente. No entanto, não sabemos a fatoração de $f$ em termos de $A, B, C, D$. Temos que a derivada de $f$ é

$$\begin{aligned} f' &= A'B^2C^3D^9 + 2ABB'C^3D^9 + 3AB^2C^2C'D^9 + 9AB^2C^3D^8D' \\ &= A'B^2C^3D^9 + 2ABB'C^3D^9 \\ &= BC^3D^9(A'B + 2AB'). \end{aligned}$$

Uma execução de SSF($f$) com $f = AB^2C^3D^9$ e

$$f' = g = BC^3D^9(A'B + 2AB')$$

dá

| $i$ | $h$ | $r$ | $s$ | $t$ | $S$ |
|---|---|---|---|---|---|
| 1 | $BC^3D^9$ | $AB$ | $B$ | $A$ | $A$ |
| 2 | $C^3D^9$ | $B$ | $1$ | $B$ | $AB^2$ |
| 3 | $C^3D^9$ | $1$ | | | |
| | $CD^3$ | | | | $AB^2(\mathrm{SFF}(CD^3))^3$ |

Agora consideramos a chamada recursiva SSF($CD^3$) com $f = CD^3$ e $f' = g = C'D^3 + 3CD^2D' = C'D^3$:

| $i$ | $h$ | $r$ | $s$ | $t$ | $S$ |
|---|---|---|---|---|---|
| 1 | $D^3$ | $C$ | 1 | $C$ | $C$ |
| 2 | $D^3$ | 1 | | | |
| | $D$ | | | | $C(\mathrm{SFF}(D))^3$ |

A seguir, consideramos a chamada recursiva SFF($D$) com $f = D$ e $f' = g = D'$:

| $i$ | $h$ | $r$ | $s$ | $t$ | $S$ |
|---|---|---|---|---|---|
| 1 | 1 | $D$ | 1 | $D$ | $D$ |
| 2 | 1 | 1 | | | |

Finalmente temos que

$$\begin{aligned} f &= AB^2(\mathrm{SFF}(CD^3))^3 = AB^2(C^1(\mathrm{SFF}(D))^3)^3 \\ &= AB^2C^3(\mathrm{SFF}(D))^9 = AB^2C^3D^9, \end{aligned}$$

que é a fatoração de $f$ livre de quadrados.

### 3.6.2 Fatoração em graus distintos (DDF)

Teorema 3.1.1 desempenha um papel crucial na fatoração de um polinômio em graus distintos. Antes de vermos cada passo do método, iremos contar um pouco da história deste resultado.

Galois [44] foi o primeiro a provar e publicar o Teorema 3.1.1 em 1830. No entanto, Gauss tinha uma prova desde 1798, apesar de nunca tê-la publicado. Esta informação só foi aparecer nas suas obras póstumas. Antes de Gauss, Legendre [84] também tinha uma prova para o caso $q = p$. Essas idéias foram então reunidas para obter uma decomposição parcial de $f$ em fatores, cada um dos quais contendo o produto de todos os fatores lineares, quadráticos, ternários, etc. de $f$. Este algoritmo deu origem à *fatoração em graus distintos* (DDF). Contudo, Galois "esqueceu" um passo crucial: dividir o polinômio $f$ pelo $\mathrm{mdc}(f, x^{q^i} - x)$ e assim eliminar os fatores irredutíveis de grau $i$. Esse algoritmo de fatoração parcial foi desenvolvido por Serret [122], e então redescoberto por Arwin [3], Cantor e Zassenhaus [20], entre outros. Mais referências podem ser encontradas em [58].

A fatoração em graus distintos fornece um método para decompor $f$ em polinômios contendo todos os fatores de grau 1, grau 2, etc. Se todos os fatores de $f$ têm graus distintos, então temos assim a fatoração completa de $f$. Para um polinômio escolhido aleatoriamente de maneira uniforme, isso acontece mais da metade das vezes. De fato, em $\mathbb{F}_2[x]$, a probabilidade é 0.6656, e num corpo finito muito grande, a probabilidade cai para $e^{-\gamma} = 0.5614\ldots$ (veja [40] para mais resultados sobre polinômios aleatórios). Apresentamos a seguir o algoritmo.

**Algoritmo 3.6.4.** [Fatoração em graus distintos (DDF)]
**Procedimento** DDF($f$)
**Entrada**: um polinômio $f \in \mathbb{F}_q[x]$ de grau $n$, livre de quadrados.

**Saída**: polinômios $g_1, g_2, \ldots, g_n$ tais que cada $g_i$ é o produto de todos os fatores irredutíveis de $f$ de grau $i$.

$h_0 \leftarrow x$; $f_0 \leftarrow f$;
`para` $i = 1$ `até` $\lfloor n/2 \rfloor$ `faça {`
  `calcule` $h_i \equiv h_{i-1}^q \pmod f$ `em` $\mathbb{F}_q[x]$;
  `calcule` $g_i = \text{mdc}(h_i - x, f_{i-1})$ `em` $\mathbb{F}_q[x]$;
  $f_i = f_{i-1}/g_i$; `}`
`para` $i = \lfloor n/2 \rfloor + 1$ `até` $n$ `faça` $g_i \leftarrow 1$;
$k \leftarrow \text{grau}(f_{\lfloor n/2 \rfloor})$;
`se` $k > \lfloor n/2 \rfloor$ `então` $g_k \leftarrow f_{\lfloor n/2 \rfloor}$;
`retorne` $(g_1, g_2, \ldots, g_n)$.

Note que se o polinômio $f_{\lfloor n/2 \rfloor}$ calculado acima tem grau $k$ maior que $\lfloor n/2 \rfloor$, temos que atualizar $g_k$, já que o fator restante depois que processarmos $i = \lfloor n/2 \rfloor$ é irredutível.

**Exemplo 3.6.5.** Seja $f(x) = x^{15} - 1 \in \mathbb{F}_{11}[x]$. Temos que

$$\text{mdc}(x^{15} - 1, x^{11} - x) = x^5 - 1 \quad \text{e}$$

$$\text{mdc}(x^{11^2} - 1, x^{10} + x^5 + 1) = x^{10} + x^5 + 1.$$

A DDF é $(x^5 - 1, x^{10} + x^5 + 1, 1, \ldots, 1)$ e assim há 5 fatores irredutíveis lineares cujo produto é $x^5 - 1$ e há 5 fatores irredutíveis quadráticos cujo produto é $x^{10} + x^5 + 1$. Para completar a fatoração de $f$, precisamos de um *algoritmo de fatoração de graus iguais* (EDF), que decompõe polinômios formados por polinômios irredutíveis de mesmo grau.

No algoritmo DDF apresentado, o "loop" principal pára quando $i = \lfloor n/2 \rfloor$. De fato, poderíamos parar antes de $\lfloor n/2 \rfloor$ usando "a estratégia de abortar antes" como exemplificaremos a seguir.

**Exemplo 3.6.6.** Seja $f$ um polinômio de grau 28 formado pelo produto de 20 fatores lineares e um fator de grau 8. A execução de DDF($f$) dá:

$i = 1$: $g_1$ tem grau 20 (20 fatores lineares) e $f_1$ tem grau 8;
$i = 2$: $f_2$ tem grau 8;
$i = 3$: $f_3$ tem grau 8;

$i = 4$: $f_4$ tem grau 8;
$i = 5$: e como $5 > 8/2$, pára com um fator irredutível de grau 8.

Este exemplo mostra que não precisamos considerar todos os graus até $\lfloor n/2 \rfloor$ (14 no último exemplo), mas até o grau dos fatores restantes dividos por 2. Isto aceleraria consideravelmente os cálculos sem nenhum custo extra (veja [40] para mais detalhes).

### 3.6.3 Fatoração de graus iguais (EDF)

O algoritmo que será apresentado nesta seção foi desenvolvido por Cantor e Zassenhaus [20]. Outros algoritmos de fatoração de graus iguais podem ser encontrados em [6, 112]. Em contraste com os algoritmos anteriores, todos os algoritmos EDF rápidos são probabilísticos. De fato, o problema de encontrar um algoritmo determinístico em tempo polinomial para EDF continua em aberto.

Inicialmente suponhamos que $q$ seja ímpar, $r \geq 2$ e $f = f_1 \cdots f_r$, onde $\text{grau}(f_i) = d$ para cada $i$, $1 \leq i \leq r$, e $\text{grau}(f) = n = dr$. Nesta altura do processo, estaremos supondo que não temos esta fatoração. Pelo teorema chinês dos restos (veja Apêndice B), temos que

$$\begin{array}{rcl} \mathbb{F}_q[x]/(f) & \rightarrow & \mathbb{F}_q[x]/(f_1) \times \cdots \times \mathbb{F}_q[x]/(f_r). \\ c \pmod f & \mapsto & (c \pmod{f_1}, \ldots, c \pmod{f_r}) \end{array}$$

é um isomorfismo. A idéia crucial é que se $c \in \mathbb{F}_q[x]$ é tal que $c \equiv 0 \pmod{f_i}$ e $c \not\equiv 0 \pmod{f_j}$, para certos $i$ e $j$ distintos, então o $\text{mdc}(c, f)$ decompõe $f$. Para qualquer $i$, $1 \leq i \leq r$, temos

$$\mathbb{F}_q \subseteq \mathbb{F}_q[x]/(f_i) \cong \mathbb{F}_{q^d}.$$

Tomamos $q$ ímpar e $m = (q^d - 1)/2$. Escolha aleatoriamente um elemento $a \in \mathbb{F}_q[x]/(f)$ não nulo de grau menor que $\text{grau}(f)$ com $a \mapsto (a_1, \ldots, a_r)$, através do teorema chinês dos restos. Os elementos $a_i$'s, $1 \leq i \leq r$, são independentes e uniformemente distribuídos em $\mathbb{F}^*_{q^d}$. Temos que $a_i^m = \pm 1$ com probabilidade $1/2$ cada. As únicas duas $r$-uplas que não contribuem na fatoração de $f$ são $(0, \ldots, 0)$ e $(-2, \ldots, -2)$. Logo, o $\text{mdc}(a^m - 1, f)$ não decompõe $f$ com probabilidade $2(\frac{1}{2})^r \leq \frac{1}{2}$, isto é, o $\text{mdc}(a^m - 1, f)$ fatora $f$ com probabilidade igual a pelo menos $1/2$.

Depois de obter $f = g_1 g_2$, podemos proceder de duas maneiras: aplicando o algoritmo recursivamente para $g_1$ e $g_2$, ou "refinando" uma fatoração já calculada $f = \prod_{i=1}^{s} u_i$ pelo $\mathrm{mdc}(u_i, h^m - 1)$ para $h$ aleatório. A primeira opção é óbvia de modo que descreveremos o algoritmo EDF para refinar o processo. Pode-se mostrar que para qualquer $\varepsilon$, $0 < \varepsilon < 1$, com $2\lceil \log_2 \frac{r^2}{\varepsilon} \rceil$ tais escolhas aleatórias, obtemos a fatoração completa de $f$ com probabilidade igual a pelo menos $1 - \varepsilon$.

**Algoritmo 3.6.7.** [Fatoração de graus iguais (EDF) de Cantor-Zassenhaus]

**Procedimento** EDF$(f, \varepsilon)$

**Entrada**: $q$ uma potência de um primo ímpar, um polinômio mônico $f \in \mathbb{F}_q[x]$ de grau $n = rd$ livre de quadrado com $r \geq 2$ fatores irredutíveis, cada um de grau $d$, e um parâmetro de confiança $\varepsilon$.

**Saída**: um conjunto de fatores mônicos irredutíveis de $f$ ou "falha".

Fatores $\leftarrow \{f\}$; $k \leftarrow 1$; $t \leftarrow 2\lceil \log \frac{r^2}{\varepsilon} \rceil$;
`enquanto` $k \leq t$ `faça {`
  `escolha aleatoriamente` $h \in \mathbb{F}_q[x]$ `com` $\mathrm{grau}(h) < n$;
  $g \leftarrow \mathrm{mdc}(h, f)$;
  `se` $g = 1$ `então` $g \leftarrow h^{(q^d-1)/2} - 1 \pmod f$;
  `para cada` $u \in$ Fatores `com` $\mathrm{grau}(u) > d$ `faça {`
    `se` $\mathrm{mdc}(g,u) \neq 1$ `e` $\mathrm{mdc}(g,u) \neq u$ `então {`
      Fatores $\leftarrow$ Fatores $\setminus \{u\}$
        $\cup \{\mathrm{mdc}(g,u), u/\mathrm{mdc}(g,u)\}$; `}};`
  `se` #Fatores $= r$ `então retorne` Fatores;
  $k \leftarrow k + 1$; `}`
`retorne "falha"`.

O procedimento anterior aplica-se somente a corpos finitos de característica ímpar. Cantor e Zassenhaus também propõem um procedimento para corpos finitos de característica 2. Porém, quando a característica é 2, há métodos mais simples baseados no cálculo do traço de elementos aleatórios em $R = \mathbb{F}_q[x]/(f)$. Esta idéia aparece em [6, 112] para ambas as características, pares e ímpares, e funciona da seguinte maneira. Escolhemos $h \in R$ aleatoriamente, então

calculamos seu traço,

$$g = \sum_{i=0}^{d-1} h^{q^i}.$$

A função traço tem imagem $\mathbb{F}_q^r$, elevando $g$ à potência $(q-1)/2$ no caso da característica ser ímpar, ou calculando $\sum_{i=0}^{\ell-1} g^{2^i}$ quando $q = 2^\ell$, leva a uma fatoração não trivial de $f$ de maneira análoga, e com probabilidades similares, como no algoritmo acima [20].

Observamos que a EDF pode ser calculada de maneira muito rápida usando repetidamente qualquer um dos métodos acima (Cantor-Zassenhauss quando a característica é ímpar, ou a função traço para qualquer característica). De fato, como vimos, se o número de fatores irredutíveis é $r$, a probabilidade de sucesso é

$$\frac{2^r - 2}{2^r} = 1 - \left(\frac{1}{2}\right)^{r-1}.$$

Então, quando $r = 2$, a probabilidade de sucesso é $1/2$. Assim, espera-se que um polinômio com dois fatores seja fatorado completamente com duas iterações.

### 3.6.4 Tempo de execução

Os tempos de execução dos algoritmos de fatoração apresentados nas seções anteriores dependem da aritmética usada e da variante considerada em cada estágio. O gargalo da execução do algoritmo é o DDF. Vários pesquisadores propuseram variantes do algoritmo DDF básico [55, 59, 81, 124]. Kaltofen e Shoup [81] têm um algoritmo com tempo de execução essencialmente $O(n^{1.815}(\log q)^{0.407})$ operações em $\mathbb{F}_q$. Este é o primeiro algoritmo de tempo subquadrático em $n$ para o estágio de fatoração em graus distintos e para o processo geral de fatoração. Contudo, o algoritmo usa multiplicação rápida de matrizes, de maneira que sua praticidade não é clara. Neste sentido, Shoup [124] dá uma versão adaptada deste algoritmo com tempo de execução essencialmente $O(n^{2.5} + n^{1+o(1)} \log q)$ operações em $\mathbb{F}_q$, usando um espaço de $O(n^{1.5})$ elementos em $\mathbb{F}_q$. Uma comparação mais precisa dos tempos de execução de várias versões de cada estágio pode ser encontrada em [58].

### 3.6.5 Algoritmo de Berlekamp

Um dos algoritmos mais clássicos para fatorar polinômios de uma variável sobre um corpo finito foi desenvolvido por Berlekamp [7, 8, 9]. A única condição imposta pelo algoritmo é que o polinômio a ser fatorado deve ser livre de quadrados. Lembramos ao leitor de que um algoritmo para eliminar fatores repetidos foi apresentado na Seção 3.6.1.

Descreveremos o método brevemente. Os detalhes completos, bem como as provas dos teoremas envolvidos, podem ser encontrados em [7, 8, 9].

Seja $f$ um polinômio mônico de grau $n$ e livre de quadrados, digamos $f = f_1 \cdots f_r$. Sejam $R = \mathbb{F}_q[x]/(f)$ e $R_i = \mathbb{F}_q[x]/(f_i)$, $1 \leq i \leq r$. O teorema chinês dos restos garante que $R \cong R_1 \times \cdots \times R_r$.

A seguinte aplicação é crucial no método de Berlekamp.

**Definição 3.6.8.** A aplicação

$$\begin{array}{rccc} \Phi: & R & \longrightarrow & R, \\ & h & \longmapsto & h^q \end{array}$$

é chamada o *automorfismo de Frobenius* em $R$.

Seja $B = \{h \in R\colon h^q = h\}$ o conjunto dos pontos fixos de $\Phi$. Não é difícil ver que $B$ é um subanel[7] de $R$ que contém $\mathbb{F}_q$. Observamos que, para $g \in \mathbb{F}_{q^n}$, temos $g^q = g$ se e somente se $g \in \mathbb{F}_q$. Assim, pelo teorema chinês dos restos, obtemos

$$B \cong \underbrace{\mathbb{F}_q \times \cdots \times \mathbb{F}_q}_{r \text{ vezes}}.$$

**Teorema 3.6.9.** *Seja $h \in B$. Então $f = \prod_{\alpha \in \mathbb{F}_q} \operatorname{mdc}(f, h - \alpha)$.*

A observação fundamental aqui é que, para qualquer $h \in B$ com $\operatorname{grau}(h) \geq 1$, uma fatoração não trivial de $f$ é

$$\prod_{\alpha \in \mathbb{F}_q} \operatorname{mdc}(f, h - \alpha),$$

já que $\operatorname{grau}(h) < n$.

[7] $B$ também é conhecido como a *subálgebra de Berlekamp*.

**Definição 3.6.10.** Um conjunto $S = \{h_1, \ldots, h_m\}$ é chamado um *conjunto separador de* $f$ se para quaisquer dois fatores irredutíveis de $f$, digamos, $f_i$ e $f_j$, existe $h_k \in S$ tal que $\prod_{\alpha \in \mathbb{F}_q} \mathrm{mdc}(f, h_k - \alpha)$ separa $f_i$ de $f_j$.

Berlekamp finalmente demonstrou que uma base de $B$ é um conjunto separador.

**Teorema 3.6.11.** *Se* $\{1, v_2, \ldots, v_r\}$ *é uma base para* $B$*, então*

$$\{v_2, \ldots, v_r\}$$

*é um conjunto separador.*

Podemos agora descrever o algoritmo. Estaremos usando $I$ para representar a aplicação identidade.

**Algoritmo 3.6.12.** [Método de fatoração de Berlekamp]
**Entrada**: um polinômio mônico $f \in \mathbb{F}_q[x]$ de grau $n$, livre de quadrados.
**Saída:** os fatores irredutíveis de $f$.

1. `Obtenha a matriz da aplicação` $\Phi - I$ `em termos da base` $\{1, x, \ldots, x^{n-1}\}$ `de` $R$`;`
2. `obtenha uma base` $\{1, v_2, \ldots, v_r\}$ `do` $\mathrm{Nucleo}(\Phi - I) = B$ `usando eliminação Gaussiana;`
3. `calcule` $\prod_{\alpha \in \mathbb{F}_q} \mathrm{mdc}(f, v_2 - \alpha)$ `e refine a fatoração sucessivamente usando` $v_3, \ldots, v_r$ `até obter a fatoração completa de` $f$.

Como podemos ver, o algoritmo de Berlekamp é baseado na solução de um sistema de equações. O número de operações em $\mathbb{F}_q$ deste algoritmo é essencialmente cúbico no grau $n$, usando a eliminação Gaussiana com a aritmética clássica para encontrar uma base de $B$.

Se o grau $n$ e o tamanho do corpo $q$ são grandes, o cálculo da base no algoritmo de Berlekamp pode ser calculado mais rapidamente usando o método para resolver sistemas lineares esparsos de Wiedemann [134]. Kaltofen e Lobo [80] implementaram o algoritmo de

Berlekamp no tempo esperado essencialmente $O(n^2 + n \log q)$ operações aritméticas em $\mathbb{F}_q$. Eles usaram um algoritmo probabilístico e o método de Wiedemann para encontrar elementos não nulos no núcleo de $\Phi - I$; veja mais detalhes em [29, 79].

Outros métodos existentes usam idéias similares às introduzidas por Berlekamp (veja, por exemplo, o algoritmo de Niederreiter [104]).

# Capítulo 4

# Criptografia

A *criptografia* é o estudo de técnicas para a segurança e a proteção de dados. Em particular, as técnicas de encriptação permitem que dois ou mais usuários possam comunicar-se de forma segura através de um canal de comunicação como a internet, por exemplo. O lema básico da criptografia é garantir que uma certa mensagem seja enviada com segurança para os destinatários desejados, sem a intercepção de nenhum outro usuário. Há também que se garantir que os destinatários tenham alguma prova de autenticidade quanto à mensagem recebida, assim como quanto à identidade do remetente.

*Cifragem* é a terminologia usada para o processo de converter um texto plano num *texto cifrado*. O processo que inverte esta operação é chamado *decifragem*. *Cifras* são os algoritmos de cifragem e decifragem. As cifras, em geral, dependem de um parâmetro chamado *chave*. Alguns métodos criptográficos são baseados numa *chave pública*, o que significa que todos têm acesso à chave e podem usá-la para encriptar uma mensagem. No entanto, para decifrar uma mensagem, é necessário ter uma *chave privada*.

Na prática, a criptografia anda cada vez mais presente no nosso dia-a-dia. Uma simples transação bancária, o comércio eletrônico e o uso de cartões inteligentes são apenas alguns exemplos.

Nos últimos trinta anos, a criptografia tem sido responsável pelo desenvolvimento de várias áreas na matemática. Em particular, corpos finitos têm recebido uma atenção especial, já que a maioria dos

sistemas criptográficos (também conhecidos como *criptossistemas*) baseados em corpos finitos tem mostrado serem eficientes. Os mais conhecidos são Diffie-Hellman [36], ElGamal [38], RSA [86, 87, 88], o criptossistema baseado em códigos de Goppa [89, 96], Chor-Rivest [21], o powerline de Lenstra [85] e o criptossistema baseado em curvas elíticas [13, 27, 98]. Os criptossistemas Chor-Rivest, Diffie-Hellman e ElGamal serão introduzidos nas Seções 4.1, 4.2 e 4.3.

Nenhum criptossistema é perfeito, porque a segurança de todos eles está baseada em algum tipo de limitação matemática e computacional. Quando este limite deixa de existir, dizemos que o criptossistema é *quebrado*. Por exemplo, o criptossistema RSA é um dos mais antigos e conhecidos, tendo sido inventado em 1978 por Rivest, Shamir e Adleman (as iniciais de RSA). Um dos fatores que garantem a segurança do RSA é o fato de que não se conhece um método eficiente para fatorar um número que é um produto de dois primos grandes. Claramente, para que esse seja de fato um problema, os dois primos devem ser convenientemente escolhidos.

A segurança do RSA é garantido pelo fato de que não se conhece um método eficiente para fatorar um número muito grande que é o produto de apenas dois primos grandes

Neste sentido, um conceito importante na segurança de um criptossistema é o de função *unidirecional*. Este tipo de função tem a propriedade de que é fácil de usá-la, mas é difícil de invertê-la. A segurança da criptografia de chave pública depende da existência deste tipo de função. No entanto, não sabemos se funções unidirecionais existem. Há, sim, candidatos a este tipo de função. O exemplo do RSA discutido acima corresponde à função que multiplica dois primos. Neste caso, a função inversa consiste em fatorar um dado inteiro como um produto de dois primos. Dependendo dos primos, esta tarefa pode ser muito difícil. Uma outra função deste gênero é a exponenciação, cuja função inversa consiste em encontrar o *logaritmo discreto*. Discutiremos esta função na Seção 4.4, onde também mencionaremos o método do cálculo de índices, o algoritmo de Waterloo e o método de Coppersmith.

Uma *cifra de chave simétrica* usa a mesma chave para a cifragem e a decifragem. Existem dois tipos de cifras de chave simétrica: as *cifras de blocos* e *de fluxo*. As cifras de blocos quebram um texto em blocos e retornam blocos cifrados do mesmo tamanho. Elas são

usadas nas operações práticas e também na defesa de alguns ataques criptográficos. As cifras de blocos padronizadas modernas são DES[1] e AES[2], embora a cifra DES não é mais considerada segura. As cifras de fluxo recebem um fluxo contínuo de texto e retornam um fluxo de dados cifrados. O RC4[3], criado por Rivest em 1987, é um exemplo de uma cifra de fluxo bem conhecida, pois é simples e é bastante rápido. Na Seção 4.5, discutiremos brevemente algumas idéias das cifras de blocos Rijndael e da cifra de fluxo WG[4].

Cifras de chave simétrica estudam principalmente cifras de blocos e de fluxo. *Cifras de blocos* toma um bloco de plaintext data e uma chave, e retorna um bloco de dados com texto cifrado do mesmo tamanho. Eles são usados em operações práticas e também na defesa de alguns ataques criptgráficos. As cifras de blocos clássicas standards são DES e AES (embora DES é não é mais standard).

Ao descrevermos os métodos criptográficos, consideramos as mensagens como sendo $n$-uplas em $R^n$. No sistemas que veremos neste capítulo, $R$ é um conjunto de inteiros ou um corpo finito. Este processo de conversão entre letras e símbolos pode ser feito de muitas maneiras, tanto pela pessoa que envia como pela que recebe a mensagem. Omitimos esses dois passos pois eles não são interessantes do ponto de vista matemático.

Os processos de encriptação e decriptação de uma mensagem $M$ podem ser visto como funções $E$ e $D$, respectivamente, tais que $E(D(M)) = M$ e $D(E(M)) = M$. O que torna o criptossistema seguro é a dificuldade de se obter a função $D$.

## 4.1 Criptossistema de Chor-Rivest

Sejam $q = p^m$, $m \geq 2$, $\mathbb{F}_q \cong \mathbb{F}_p[x]/(f)$ e $g$ um elemento primitivo de $\mathbb{F}_q$ representado por um polinômio em $\mathbb{F}_p[x]$. Cada elemento em $\mathbb{F}_q$ é um polinômio de grau menor que $m$, que também pode ser representado como uma potência de $g$. Logo, para cada $0 \leq i < p$,

[1] Data Encryption Standard.
[2] Advanced Encryption Standard.
[3] Rivest Code.
[4] Welch-Gong.

existe um inteiro $a_i$ tal que $0 < a_i < p^m - 1$ e

$$x - i \equiv g^{a_i} \pmod{f}.$$

Dada uma permutação $\Pi$ de $\{0, 1, \ldots, p-1\}$, definimos $b_i$, $0 \leq i < p$, por $b_i = a_{\Pi(i)}$.

Usando os dados acima, descrevemos agora o método de Chor-Rivest.

**Mensagem:** $M = (M_0, \ldots, M_{p-1})$ tal que cada $M_i \in \mathbb{N} \cup \{0\}$ e

$$\sum_{i=0}^{p-1} M_i < m$$

**Chave pública:** $\{b_0, b_1, \ldots, b_{p-1}\}$

**Chave privada:** polinômios $f, g$ e uma permutação $\Pi$

**Encriptação:** $E(M) \equiv \sum_{i=0}^{p-1} b_i M_i \pmod{q-1}$

**Decriptação:** dado o texto $c$, a mensagem $M$ é formada pelos expoentes dos fatores irredutíveis da redução de $g^c$ módulo $f$.

Vamos justificar a decriptação. Suponhamos que $c \equiv \sum_{i=0}^{p-1} b_i M_i \pmod{q-1}$. Seja $t \equiv g^c \pmod{f}$. Usando o Lema 1.4.5, temos que

$$\begin{aligned} g^c &\equiv g^{\sum_{i=0}^{p-1} b_i M_i} \equiv \prod_{i=0}^{p-1} g^{b_i M_i} \\ &\equiv \prod_{i=0}^{p-1} g^{a_{\Pi(i)} M_i} \equiv \prod_{i=0}^{p-1} (x - \Pi(i))^{M_i} \pmod{f}. \end{aligned}$$

Como $\sum_{i=0}^{p-1} M_i < m$, segue que $t = \prod_{i=0}^{p-1} (x - \Pi(i))^{M_i}$. Portanto, a mensagem é obtida fatorando $t$. Isto pode ser feito usando um dos algoritmos do Capítulo 3. As multiplidades $M_i$ dos fatores determinarão a mensagem $M$.

Um outro aspecto do criptossistema de Chor-Rivest depende do *problema da mochila*[5]. Na verdade, há outros criptossistemas que dependem deste problema.

[5] O problema da mochila é um problema famoso na otimização combinatória e é conhecido como "knapsack problem" em inglês.

O método de Chor-Rivest já não é mais seguro. De fato, Vaudenay [131] quebrou este sistema usando os parâmetros sugeridos, $\mathbb{F}_{p^{24}}$ e $\mathbb{F}_{256^{25}}$, na literatura. Para informações sobre aspectos de implementação, veja [71].

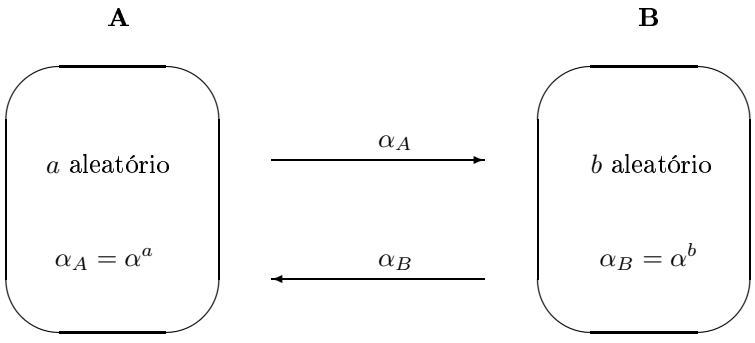
Baseado no trabalho de Chor e Rivest, Lenstra [85] introduziu o *criptossistema powerline*. Ao invés de usar o problema da mochila, ele usa a estrutura multiplicativa do corpo. Este método ainda não foi quebrado.

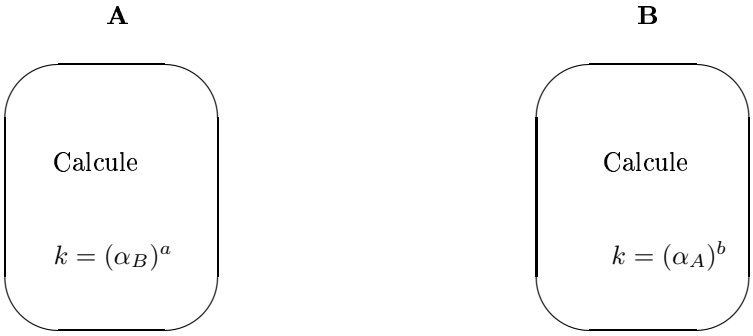
## 4.2 Esquema Diffie-Hellman

Apresentamos nesta seção o método de Diffie-Hellman para compartilhar uma chave.

Suponhamos que Alice (**A**) e Bob (**B**) queiram ter uma chave em comum. Suponhamos que **A** calcula um valor aleatório $a$, **B** calcula um valor aleatório $b$ e vamos supor que $\alpha$ seja um elemento primitivo em $\mathbb{F}_{q^n}$. Então **A** calcula $\alpha_A = \alpha^a$ e manda para **B**, enquanto **B** calcula $\alpha_B = \alpha^b$ e transmite para **A**. Agora, **A** pode calcular $(\alpha_B)^a = \alpha^{ab}$ e **B** pode calcular $(\alpha_A)^b = \alpha^{ab}$, e então eles compartilham o valor da chave $k = \alpha^{ab}$.

**Público:** $\mathbb{F}_q$ e um elemento primitivo $\alpha$.






Mesmo se um intruso obtiver $\alpha^a$ ou $\alpha^b$, acredita-se que será computacionalmente difícil para ele calcular $a$, $b$ ou $\alpha^{ab}$.

Como veremos na Seção 4.4, há métodos rápidos para calcular logaritmos em extensões binárias. Os padrões internacionais para o Diffie-Hellman são os corpos primos. Quando o Diffie-Hellman é usado em criptografia de curvas elíticas, o padrão é corpos primos ou binários. Por exemplo, a recomendação IEEE para o método de Diffie-Hellman é usar primos de 1024 a 8192 bits; veja [74].

Finalmente, é interessante comentar que o método de Diffie-Hellman é muito usado na prática. Uma aplicação recente apareceu em telefones celulares com capacidade de encriptação de vozes. Nessas aplicações, o Diffie-Hellman é usado para autenticar, enquanto o AES é usado para encriptar.

## 4.3 O sistema ElGamal

O sistema que apresentamos nesta seção foi introduzido por ElGamal [38] em 1984 para assinatura digital. O método ElGamal é baseado na chave pública. Fixamos $p$ primo e $g$ um elemento primitivo de $\mathbb{F}_p$. Alice escolhe uma chave privada $a$ onde $0 < a < p-1$ e determina que a chave pública é $(g, g^a \pmod p, p)$. Os processos de encriptação e decriptação estão descritos abaixo.

**Chave privada:** $a$ onde $0 < a < p-1$ onde $p$ é primo

**Chave pública:** $K = (g, b, p)$ onde $p$ é primo, $g$ é um elemento primitivo em $\mathbb{F}_p$ e $b \equiv g^a \pmod p$

**Mensagem:** $M$ onde $0 < M < p$

**Encriptação:** Escolha um número aleatório $r$, $1 < r < p-1$. Em seguida, calcule $e_1 \equiv g^r \pmod{p}$ e $e_2 \equiv Mb^r \pmod{p}$. Retorne $E_K(M) = (e_1, e_2)$.

**Decriptação:** $D_K(M)(e_1, e_2) \equiv e_1^{-a} e_2 \pmod{p}$

Para verificarmos a decripação, observamos que

$$D_K(M)(e_1, e_2) \equiv (e_1)^{-a} e_2 \equiv g^{-ar} e_2 \equiv Mb^r b^{-r} \equiv M \pmod{p}.$$

## 4.4 O problema do logaritmo discreto

Na Seção 4.2, comentamos brevemente sobre o problema do logaritmo discreto. Nesta seção, mostraremos métodos subexponenciais para calcular logaritmos discretos nas extensões de corpos finitos. Como conseqüência destes resultados, calcular logaritmos nas extensões binárias é mais rápido do que nos corpos primos de mesmo tamanho. Assim sendo, a segurança do logaritmo discreto em extensões binárias está comprometida quando comparada com a segurança do logaritmo discreto com corpos primos.

**Definição 4.4.1.** *Seja $g$ um gerador de $\mathbb{F}_q^*$. Para todo $h \in \mathbb{F}_q^*$, existe um inteiro $x$, $0 \leq x \leq q-2$, tal que $h = g^x$. Neste caso, dizemos que $x$ é o* logaritmo discreto de $h$ na base $g$ *e escrevemos $x = \log_g h$.*

O *problema do logaritmo discreto* consite em encontrar um algoritmo computacionalmente viável para calcular o logaritmo discreto de um dado elemento $h \in \mathbb{F}_q^*$.

Na prática, os corpos finitos mais interessantes são $\mathbb{F}_{2^n}$ ou, em geral, $\mathbb{F}_{p^n}$, onde $p$ é um primo pequeno e $n$ é grande.

A segurança de muitos criptossistemas de chave pública se baseia na dificuldade de resolver o problema do logaritmo discreto. Um avanço importante nesta área se deve ao desenvolvimento do *método do cálculo de índices*, criado por Western e Miller [133]. Em seguida, mostraremos como este método funciona e então veremos sua análise, que foi rigorosamente estudada por Adleman e Odlyzko [1, 106].

### 4.4.1 Método do cálculo de índices

Seja $S$ um conjunto de polinômios irredutíveis sobre $\mathbb{F}_p$, onde $p = \text{car}(\mathbb{F}_q)$. Digamos que todos os polinômios em $S$ têm grau menor

ou igual a $m$. Estaremos interessados nos polinômios que fatoram-se completamente em $S$. Tais polinômios são conhecidos como *polinômios $m$-suaves*. Sejam $\mathbb{F}_q \cong \mathbb{F}_p[x]/(f)$ e $g$ um gerador de $\mathbb{F}_q^*$. Além disso, seja $h^*$ o elemento cujo logaritmo queremos calcular.

O método do cálculo de índices consiste de duas partes: a construção de uma grande base de dados contendo logaritmos e o cálculo de logaritmos individuais.

**Passo 1:** Escolha um inteiro $s$, $1 \leq s \leq q-1$, aleatoriamente de maneira uniforme. Em seguida, determine $h$ satisfazendo $h \equiv g^s \pmod{f}$ com $\text{grau}(h) < n$. Verifique se $h$ fatora-se completamente como um produto de irredutíveis em $S$. Se não, descarte $s$ e repita o processo escolhendo um novo inteiro $s$ até encontrar $h$ que fatora-se em $S$; digamos,

$$h = \prod_{v \in S} v^{e_v(h)}.$$

Então, salve a equação

$$s \equiv \sum_{v \in S} e_v(h) \log_g v \pmod{q-1}.$$

Repita os cálculos acima até obter "um pouco mais" que $\#S$ congruências. Depois resolva o sistema para determinar $\log_g v$ para todo $v \in S$.

**Passo 2:** Escolha um inteiro $s$, $1 \leq s \leq q-1$, aleatoriamente de maneira uniforme. Então calcule $h$ tal que $h \equiv h^* g^s \pmod{f}$ com $\text{grau}(h) < n$. Verifique se $h$ se fatora completamente em irredutíveis sobre $S$. Se não, descarte $s$ e repita o processo escolhendo um novo inteiro $s$. Se $h$ se fatora em $S$, digamos,

$$h = \prod_{v \in S} v^{e_v(h)},$$

temos que o logaritmo discreto desejado é

$$\sum_{v \in S} e_v(h) \log_g v \pmod{q-1}.$$

O algoritmo apresentado dá uma estrutura básica na qual métodos diferentes podem ser empregados para calcular as tarefas intermediárias. Odlyzko [106] determinou algumas variantes desses cálculos. Uma tarefa crucial no Passo 1 é determinar o tamanho do sistema para obter uma base de dados de logaritmos discretos; veja [106, páginas 238-243]. A base de dados é então usada no Passo 2 para calcular logaritmos discretos particulares.

### 4.4.2 Análise do método do cálculo de índices

A análise do método do cálculo de índices requer o estudo da probabilidade de que um polinômio é $m$-suave. De fato, em ambos os passos descritos na seção anterior, tomamos polinômios aleatoriamente de maneira uniforme até obter um polinômio $m$-suave. Assim, esta probabilidade daria as condições para parar o algoritmo.

A estimativa desta probabilidade é feita da seguinte maneira. Primeiro, obtemos uma representação do problema de enumeração de polinômios suaves, em termos de uma função geradora. Então, usando a fórmula de Cauchy com métodos de integração de contorno e com o método do *ponto de sela*, derivamos resultados assintóticos. Esta estratégia foi desenvolvida por Odlyzko em [106] e tem sido usada com sucesso em muitos problemas; veja [42, 119]. Outras variantes desta estratégia seguem o mesmo esquema [37, 52, 53].

Continuaremos com a análise de Odlyzko. O número de polinômios mônicos de grau $n$ sobre $\mathbb{F}_q$ é $q^n$. Assim, a função geradora de polinômios sobre $\mathbb{F}_q$ é

$$P(z) = \sum_{k \geq 0} q^k z^k = \frac{1}{1 - qz}.$$

Uma outra maneira de contar o número de polinômios sobre $\mathbb{F}_q$ é através da fatoração única de polinômios em fatores irredutíveis. Seja $I_k$ o número de polinômios mônicos irredutíveis sobre $\mathbb{F}_q$ de grau $k$. Um enumerador de um fator irredutível fixo de grau $k$ é dado por

$$1 + z^k + z^{2k} + \cdots,$$

no sentido de que ele conta o número de vezes em que este polinômio irredutível aparece na decomposição de um polinômio. Usando agora

a fatoração única de polinômios, segue que a função geradora de polinômios sobre $\mathbb{F}_q$ pode também ser escrita como

$$P(z) = \prod_{k\geq 1} \left(1 + z^k + z^{2k} + \cdots\right)^{I_k} = \prod_{k\geq 1} \left(\frac{1}{1-z^k}\right)^{I_k}.$$

Concluímos que

$$P(z) = \prod_{k\geq 1} \left(\frac{1}{1-z^k}\right)^{I_k} = \frac{1}{1-qz}.$$

Seja $N_q(m;n)$ o número de polinômios mônicos sobre $\mathbb{F}_q$ de grau $n$ que são $m$-suaves. Usando o argumento anterior, devemos considerar fatores irredutíveis de grau até $m$. Neste caso, a função geradora é

$$S_m(z) = \sum_{n\geq 0} N_q(m;n) z^n = \prod_{k=1}^{m} \left(\frac{1}{1-z^k}\right)^{I_k}.$$

O número desejado de polinômios $m$-suaves em $\mathbb{F}_q[x]$ é o coeficiente de $z^n$ em $S_m(z)$, denotado por $[z^n]S_m(z)$. Se $m$ é constante, a extração do coeficiente é simples, usando o método da análise de singularidades desenvolvido por Flajolet e Odlyzko [41, 107]. No entanto, para a aplicação do método do cálculo de índices, precisamos de $m \to \infty$ quando $n \to \infty$ e isso complica consideravelmente a análise.

Odlyzko [106] conseguiu uma estimativa assintótica de $N_q(m;n)$, quando $n \to \infty$, uniformemente para $m$ no intervalo

$$n^{1/100} \leq m \leq n^{99/100}.$$

Observamos que os exponentes são arbitrários e de fato os resultados são válidos para $n^\delta \leq m \leq n^{1-\delta}$ e $\delta > 0$.

A prova baseia-se na fórmula de Cauchy

$$\begin{aligned} N(m;n) &= \frac{1}{2\pi i} \int_{|z|=r} S_m(z) z^{-n-1}\, dz \\ &= \frac{1}{2\pi} \int_{-\pi}^{\pi} S_m\left(re^{i\theta}\right) r^{-n} e^{-in\theta}\, d\theta, \end{aligned}$$

para um número real $r$, $0 < r < 1$. Odlyzko estimou esta integral assintoticamente através de uma aplicação usual do método do ponto de sela. Apresentaremos a derivação dele para um $q$ geral, embora a prova original de Odlyzko é para $\mathbb{F}_2$; veja [106, Apêndice A].

Seja $r_0 = r_0(m,n)$ a única solução da equação

$$\frac{rS'_m(r)}{S_m(r)} = n.$$

Quando $n \to \infty$ (e uniformemente para $m$ no intervalo $n^\delta \leq m \leq n^{1-\delta}$), temos

$$N(m;n) \sim \frac{S_m(r_0)r_0^{-n}}{\sqrt{2\pi b(r_0)}},$$

onde

$$b(r) = \left(\frac{S'_m(r)}{S_m(r)}\right)'.$$

Odlyzko também mostrou que $r_0 = r_0(m,n)$ é assintoticamente dado por

$$r_0 = \frac{1}{q}\exp\left(\frac{\log(n/m) + \log\log(n/m) + o(1)}{m}\right),$$

então no intervalo dado obtemos que

$$r_0 - \frac{1}{q} = O\left(\frac{\log n}{n^\delta}\right).$$

Além disso, ele mostrou que $b(r_0) \sim 4mn$. Combinando esses resultados, concluímos que

$$N_q(m;n) \quad = \quad q^n e^{-(1+o(1))\frac{n}{m}\log\left(\frac{n}{m}\right)}.$$

Podemos acelerar o Passo 2, aumentando $m$. Contudo, isso implicaria num armazenamento maior de base de dados e um tempo de execução maior do Passo 1. Otimizando todos os parâmetros do algoritmo, Odlyzko obtém, para $q = 2$, o melhor valor para $m$:

$$m = \sqrt{\frac{n\log n}{2\log 2}}.$$

Finalmente, se usamos $L(n) = e^{(1+o(1))\sqrt{n \log n}}$, então o tempo de execução do algoritmo básico, que calcula os índices, é limitado por $L(n)^c$, onde $c$ é uma constante calculável que, para $q = 2$, é $\sqrt{2 \log 2} = 1.1774\ldots$.

### 4.4.3 Algoritmo de Waterloo

Blake et al. [10, 12] propuseram uma variante do método do cálculo de índices, agora conhecido como o *algoritmo de Waterloo*. Esta variante melhora o tempo de execução do método, introduzindo um argumento heurístico que faz a análise não ser rigorosa. O melhoramento não é de ordem assintótica. Contudo, acelera o algoritmo consideravelmente na prática, com um aumento entre duas e dez vezes; veja [106].

O algoritmo é similar à versão básica apresentada na seção anterior. Exemplificaremos a mudança, usando apenas o segundo estágio do algoritmo, isto é, o cálculo dos logaritmos individuais, uma vez que calculamos a base de dados de logaritmos de polinômios irredutíveis de grau menor ou igual a $m$. Um argumento similar pode ser usado no primeiro estágio; veja [10, 12, 106].

O algoritmo depende do cálculo do logaritmo de um elemento em $\mathbb{F}_q \cong \mathbb{F}_p[x]/(f)$, onde $f$ é um polinômio mônico irredutível de grau $n$ sobre $\mathbb{F}_p$.

A idéia principal do algoritmo é baseada no seguinte fato: é mais provável que dois polinômios de grau no máximo $(n-1)/2$ tenham todos os seus fatores irredutíveis de grau menor ou igual a $m$ do que um polinômio de grau no máximo $n-1$ tenha todos os seus fatores irredutíveis de grau menor ou igual a $m$. Blake et al. propuseram encontrar dois polinômios de grau aproximadamente em torno da metade de $n$, executando o algoritmo euclideano estendido (Seção 1.3) para os polinômios $h$ e $f$. Acontece que se aplicarmos este algoritmo para $h$ e $f$, existe um passo, em que ambos os polinômios $r$ e $t$ são tais que $th \equiv r \pmod{f}$ e têm grau menor ou igual a $(n-1)/2$. Observamos que esta propriedade era conhecida mesmo antes do trabalho de Blake et al. (veja [95]). Apresentamos agora o segundo passo do algoritmo de Waterloo.

**Variante de Waterloo:**

**Passo 1:** Defina $A$ como sendo 0. Se $\text{grau}(h) \leq m$ e $h(x) =$

$\prod_i h_i(x)^{e_i}$, então $\log_g h(x) \equiv \sum_i e_i \log_g h_i(x) \pmod{q-1}$. Pare.

**Passo 2:** Gere um inteiro aleatório $a$, atualize $A$ por $A+a$ e $h$ por $hg^a$. Aplique o algoritmo euclideano estendido para $h$ e $f$. Obtenha polinômios $r$ e $t$ tais que

$$t(x)h(x) \equiv r(x) \pmod{f}$$

e $\text{grau}(r), \text{grau}(t) \leq (n-1)/2$.

**Passo 3:** Fatore $t(x) = \prod_i p_i(x)^{d_i}$ e $r(x) = \prod_j p_j(x)^{d_j}$. Se $\text{grau}(p_i) \leq m$ e $\text{grau}(p_j) \leq m$ para quaisquer $i, j$, então calcule o logaritmo discreto desejado através de

$$\log_g h(x) = \sum_j d_j \log_g p_j(x) - \sum_i d_i \log_g p_i(x) - A$$

e pare. Caso contrário, volte ao Passo 2.

A análise do tempo de execução do algoritmo de Waterloo requer o estudo da probabilidade de que dois polinômios mônicos de grau menor ou igual a $(n-1)/2$, escolhidos aleatoriamente de maneira uniforme, sejam relativamente primos e se fatorem como um produto de polinômios irredutíveis de grau menor que $m$. No artigo original, Blake et. al supuseram que os polinômios $r$ e $t$ de grau menor ou igual a $(n-1)/2$ podiam ser considerados como sendo relativamente primos. Se este é o caso, então podemos imediatamente obter a probabilidade desejada e o tempo de execução dos algoritmos, aplicando a análise de Odlyzko para a versão básica. De fato, se $P(m;(n-1)/2)$ é a probabilidade de que um polinômio de grau $(n-1)/2$ é $m$-suave e $P(m;(n-1)/2,(n-1)/2)$ é a probabilidade de que um par de polinômios, cada um de grau $(n-1)/2$, é $m$-suave, podemos estimar $P(m;(n-1)/2,(n-1)/2)$ por $P^2(m;(n-1)/2)$.

A análise rigorosa do tempo de execução do algoritmo implica em dar estimativas para o par de polinômios suaves com a condição adicional de que os polinômios sejam relativamente primos. Drmota e Panario [37] provaram que a aproximação de Blake et al. é correta, em termos assintóticos, e dão uma estimativa precisa para esta relação, provando rigorosamente o tempo de execução do algoritmo de Waterloo. Eles usaram a estratégia do ponto de sela duplo análoga ao ponto de sela de uma variável de Odlyzko.

### 4.4.4 Variante de Coppersmith

O método mais rápido para o cálculo de logaritmos discretos em $\mathbb{F}_{2^n}$ é devido a Coppersmith [28]; veja também [106]. Em seguida, descreveremos os principais ingredientes desta variante.

O método de Coppersmith foi desenvolvido para corpos finitos de característica par e depende de hipóteses que não foram provadas, mas mesmo assim têm demonstrado serem razoáveis, pelo menos na prática. Este foi o primeiro método a ser conjecturado com tempo de execução

$$e^{(c+o(1))n^{1/3}(\log n)^{2/3}},$$

onde $c$ é uma constante efetivamente calculável [28, 106].

O método assume que o polinômio mônico irredutível $f$ de grau $n$ usado para definir $\mathbb{F}_{2^n}$ é da forma $f(x) = x^n + f_1(x)$, onde $\text{grau}(f_1) \leq \log_2 n$. Pouco se sabe sobre a irredutibilidade deste tipo de polinômio; veja [49] e a Seção 3.3. Existem polinômios irredutíveis desta forma para alguns valores de $n$. Por outro lado, se consideramos $f_1$ de grau no máximo $\log_2 n + c_1$, onde $c_1$ é um inteiro positivo pequeno, então há polinômios irredutíveis desta forma para todos os valores de $n$ que tenham interesse prático [49]. Contudo, mesmo neste caso, não há provas quanto à existência de polinômios irredutíveis desta forma. De fato, a melhor cota foi encontrada por Hsu [72], que permite que a metade superior ou inferior do polinômio seja fixa, o que produz um número muito menor de coeficientes fixos do que os comentados acima.

No primeiro estágio do algoritmo, dois polinômios $w_1$ e $w_2$ são construídos de maneira que $w_2 \equiv w_1^{2^k} \pmod{f}$, onde $k$ é um parâmetro escolhido tal que $2^k$ é da ordem $n^{1/3}(\log n)^{-1/3}$. A construção de $w_1$ e $w_2$ é baseada em manipulações algébricas que são independentes do conjunto $S$. Em seguida, a suposição heurística feita é que $w_1$ e $w_2$ se comportam como polinômios aleatórios *independentes* de grau no máximo da ordem $n^{2/3}(\log n)^{1/3}$. A análise segue usando essencialmente os mesmos argumentos de Odlyzko. A análise do segundo estágio é feita de maneira similar.

Conjectura-se que o método de Coppersmith tem o melhor tempo de execução para o problema do logaritmo discreto em $\mathbb{F}_{2^n}$. Para uma descrição detalhada do algoritmo, veja o artigo original de Coppersmith [28] e o survey de Odlyzko [106]. Um outro algoritmo para

o cálculo de logaritmos discretos em corpos finitos não primos foi desenvolvido por Semaev [120].

## 4.5 Cifras

Nesta seção, apresentaremos brevemente as cifras de blocos Rijndael e a cifra de fluxo WG.

A cifra de blocos AES é *Rijndael* e foi desenvolvida por Daemen e Rijmen [32].

Rijndael tem comprimentos de bloco e de chave variáveis, ambos especificados para serem de tamanho 128, 192 ou 256 bits. A maioria das operações em Rijndael envolve bytes (8 bits), que são representados por elementos em $\mathbb{F}_{2^8}$. Algumas outras operações funcionam com palavras formadas por 4 bytes (32 bits). As operações em $\mathbb{F}_{2^8}$ podem ser eficientemente implementadas tanto em hardware como em software. Uma vantagem deste projeto é que ele é muito rápido e fácil de ser implementado. Além disso, do ponto de vista de corpos finitos, não é muito sofisticado. Os elementos chaves de Rijndael são os seguintes.

- A maioria da aritmética é feita em $\mathbb{F}_{2^8}$, usando o polinômio irredutível $x^8 + x^4 + x^3 + x + 1$.

- Todas as multiplicações são rápidas, consistindo apenas de multiplicações por $x$, que são simples deslocamentos, ou de multiplicações por $x$ seguidas por uma soma "+1". Tais operações são rapidamente implementadas em hardware.

- O cálculo de inversos é feito usando o algoritmo euclideano estendido (Seção 1.3).

- Há algumas outras funções que operam em $\mathbb{F}_{2^8}$ e também várias outras operações que funcionam com bits e bytes para misturar a mensagem encriptada.

Os detalhes de implementação e a justificativa criptográfica para a seleção do parâmetro podem ser encontrados em [32].

### 4.5.1 Cifras de blocos Rijndael

A cifra de blocos AES (*Rijndael*) foi desenvolvida por Daemen e Rijmen) e é padronizada por NIST [32]. Rijndael usa blocos de 128 bits e chaves de tamanho 128, 192 ou 256 bits. A maioria das operações em Rijndael envolve bytes (8 bits), que são representados por elementos em $\mathbb{F}_{2^8}$. Algumas outras operações funcionam com palavras formadas por 4 bytes (32 bits). As operações em $\mathbb{F}_{2^8}$ podem ser eficientemente implementadas tanto em hardware como em software. Uma vantagem deste projeto é que ele é muito rápido e fácil de ser implementado. Além disso, do ponto de vista de corpos finitos, não é muito sofisticado. Os elementos chaves de Rijndael são os seguintes.

- A maioria da aritmética é feita em $\mathbb{F}_{2^8}$, usando o polinômio irredutível $x^8 + x^4 + x^3 + x + 1$.
- Todas as multiplicações são rápidas, consistindo apenas de multiplicações por $x$, que são simples deslocamentos, ou de multiplicações por $x$ seguidas por uma soma "+1". Tais operações são rapidamente implementadas em hardware.
- Os blocos de 128 bits são interpretados como matrizes de 4x4 bytes.
- O cálculo de inversos é feito usando o algoritmo euclideano estendido (Seção 1.3).
- Há algumas outras funções que operam em $\mathbb{F}_{2^8}$ e também várias outras operações que funcionam com bits e bytes para misturar a mensagem encriptada.

Os detalhes de implementação e a justificativa criptográfica para a seleção do parâmetro podem ser encontrados em [32]. Para mais informações sobre a aritmética em corpos binários, veja [67].

### 4.5.2 Cifra de fluxo WG

Uma cifra muito interessante do ponto de vista de corpos finitos é chamada WG e foi recentemente proposta por Nawaz e Gong para possível padronização [102]. A lista completa de cifras de fluxo submetidas pode ser encontrada em http://www.ecrypt.eu.org/stream/.

Os elementos chaves da cifra WG são os seguintes.

- A aritmética é feita em $\mathbb{F}_{2^{29}}$, usando o polinômio primitivo

$$\begin{aligned} g(x) &= x^{29}+x^{28}+x^{24}+x^{21}+x^{20}+x^{19}+x^{18}+x^{17}+ \\ &\quad x^{14}+x^{12}+x^{11}+x^{10}+x^7+x^6+x^4+x+1. \end{aligned}$$

- Se $\beta$ é uma raiz de $g$, definimos $\gamma \in \mathbb{F}_{2^{29}}$ como sendo $\gamma = \beta^{464730077}$. Assim, podemos verificar que $\gamma$ é um elemento normal em $\mathbb{F}_{2^{29}}$.

- Seja

$$t(x) = x + x^{2^{10}+1} + x^{2^{19}+2^9+1} + x^{2^{19}-2^9+1} + x^{2^{19}+2^{10}-1}.$$

  A *transformada Welch-Gong* é uma função $f\colon \mathbb{F}_{2^{29}} \to \mathbb{F}_2$ definida por

$$f(x) = \mathrm{Tr}(t(x+1)+1),$$

  onde Tr é a função traço definida em $\mathbb{F}_{2^{29}}$ sobre $\mathbb{F}_2$.

- O fluxo da chave é formado pelos seguintes componentes:

  1. um LFSR sobre $\mathbb{F}_{2^{29}}$, definido pelo polinômio primitivo

$$p(x) = x^{11}+x^{10}+x^9+x^6+x^3+x+\gamma,$$

     fornecendo assim uma seqüência de comprimento máximo, e
  2. de uma *transformada Welch-Gong*.

- Todas as operações nas transformadas LFSR e WG são realizadas rapidamente com o uso de bases normais; veja Seção 2.3. Por exemplo, uma exponenciação é apenas um deslocamento à direita, o cálculo do traço de um elemento pode ser feito adicionando seus bits (o traço dos elementos da base é 1), etc. Eles são essencialmente deslocamentos e XORs.

- Há poucas outras funções que operam em $\mathbb{F}_{2^{29}}$ e várias outras operações que funcionam com bits. Elas são importantes tanto para esconder informações, como também para operar com elas.

A extensão de grau 29 foi escolhida, já que ela contém elementos normais ótimos (veja Tabela 2.2), a transformada WG existe em $\mathbb{F}_{2^{29}}$ e 29 é perto do tamanho da palavra usada (32 bits).

Os autores mostraram que a cifra WG produz um fluxo da chave com boas propriedades de aleatoriedade tais como balanço, período longo, complexidade linear grande, autocorrelação aditiva de nível 3 e a autocorrelação ideal multiplicativa de nível 3. Para uma explicação desses conceitos, veja por exemplo o livro de Golomb e Gong [62]. Essas propriedades criptográficas são derivadas da transformada WG [63]. Eles também mostraram que WG é resistente a vários ataques. Contudo, Wu e Preneel mostraram recentemente que a cifra de fluxo WG é vulnerável a certos ataques criptográficos; veja detalhes em [103, 135].

# Capítulo 5

# Teoria de Códigos

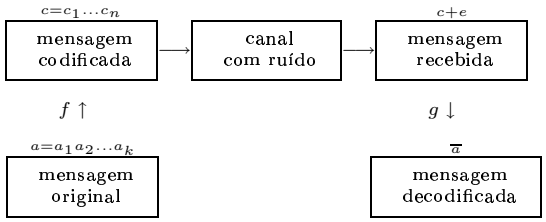
Com a crescente inovação tecnológica, a preocupação com a transmissão de informação com confiança tem aumentado consideravelmente. O problema é que dependemos de instrumentos eletrônicos que não são inteiramente confiáveis. Assim, estamos sempre sujeitos a erros como, por exemplo, a perda e o deslocamento de símbolos na mensagem enviada. Problemas como esses podem causar um não entendimento da mensagem recebida e, no pior dos casos, a perda total da mensagem. A *Teoria de Códigos* estuda justamente técnicas que tentam detectar e corrigir tais erros. Essas duas tarefas são independentes, no sentido de que um modelo de transmissão de dados pode ser mais eficiente para uma tarefa do que para a outra. Por exemplo, veremos alguns modelos que conseguem detectar erros, mas que não os conseguem corrigir.

Introduziremos agora algumas terminologias. Queremos transmitir uma mensagem através de um *canal com ruído*. A mensagem consiste de uma seqüência finita de $k$ símbolos de um certo alfabeto. Consideramos o alfabeto como sendo um corpo finito. Suponhamos que a mensagem $a$ seja *codificada* por uma *palavra-código* $c$ de $n$ símbolos, através de uma função $f$ chamada o *esquema de codificação*:

$$\begin{array}{rccc} f: & \mathbb{F}_q^k & \longrightarrow & \mathbb{F}_q^n, \\ & a=(a_1,\ldots,a_k) & \longmapsto & c=(c_1,\ldots,c_n) \end{array}$$

onde $n > k$. A idéia é enviar a palavra $c = f(a)$ ao canal de maneira

que a mensagem recebida, digamos, $c+e$, seja tal que o erro $e$ possa ser detectado e/ou corrigido. O *código* é o conjunto imagem de $f$, ou seja, o conjunto de todas as palavras-código. A função $g : \mathbb{F}_q^n \to \mathbb{F}_q^k$, que associa a mensagem recebida à mensagem *decodificada*, é chamada um *esquema de decodificação*. O seguinte diagrama reúne todos esses conceitos:




Canais perfeitos não existem, mas isto não quer dizer que haverá erros constantemente, apenas que não são 100% confiáveis. Assim, ao projetar um modelo de codificação e decodificação, é importante considerar também o caso em que a mensagem é transmitida sem nenhuma alteração.

A teoria de códigos foi uma das primeiras áreas de aplicação de corpos finitos, sendo $\mathbb{F}_2$ o mais utilizado. Neste capítulo, introduziremos algumas idéias clássicas de modelos algébricos usados na codificação e decodificação de mensagens. Na Seção 5.1, apresentaremos os *códigos lineares*. Os *códigos cíclicos* são tipos especiais de códigos lineares e serão estudados na Seção 5.2. Em particular, veremos os *códigos de Hamming* e os códigos *BCH*[1] *que corrigem dois erros*. Finalmente, a Seção 5.3 é dedicada aos *códigos BCH que corrigem $t$ erros* e os códigos *Reed-Solomon*.

As referências clássicas da teoria de códigos relacionadas aos corpos finitos são os livros de Belekamp [8], de Hankerson et al. [66], de Lidl e Niederreiter [89, Capítulo 8], de MacWilliams e Sloane [92], de McEliece [95] e de Pless [110]. Veja também os livros de Costa et al. [30] e de Hefez e Villela [69].

A nossa abordagem neste capítulo será um pouco diferente dos últimos três capítulos, uma vez que estaremos discutindo modelos

[1] as iniciais de seus criadores: Bose, Chaudhuri e Hocquenghem.

clássicos de códigos e isso nos afasta um pouco dos problemas em aberto nesta área. Baseamos essas notas nos livros de Lidl e Niederreiter [89], e de MacWilliams e Sloane [92].

## 5.1 Códigos lineares

Como o próprio nome sugere, os *códigos lineares* estão baseados em várias ferramentas da álgebra linear. Assim, é importante visualizar $\mathbb{F}_{q^n}$ não apenas como um corpo finito, mas também como um espaço vetorial sobre $\mathbb{F}_q$.

**Definição 5.1.1.** Seja $H$ uma matriz $(n-k) \times n$ com entradas em $\mathbb{F}_q$. Dizemos que o conjunto

$$C = \{c \in \mathbb{F}_q^n \colon Hc^T = 0\}$$

é um *código linear* sobre $\mathbb{F}_q$, também denotado por $C(n,k)$, enquanto que $n$ é o *comprimento* e $k$ é a *dimensão* do código. Os elementos de $C$ são chamados *palavras-código* (ou *vetores-código*) e $H$ é chamada a *matriz de paridade* de $C$. Se $q = 2$, dizemos que $C$ é um *código binário*. Se $G$ é uma matriz $k \times n$ cujo espaço gerado pelas linhas é igual a $C$, então dizemos que $G$ é uma *matriz geradora* de $C$.

O código $C(n,k)$ é um subespaço $k$-dimensional de $\mathbb{F}_q^n$, pois é o núcleo de $H$ e assim é fechado para a adição e para a multiplicação por escalares. De fato, se $c_1, c_2 \in C$ e $a \in \mathbb{F}_q$, temos que

$$H(c_1 + c_2)^T = Hc_1^T + Hc_2^T = 0 \quad \text{e} \quad H(ac)^T = aHc^T = 0.$$

Em particular, o vetor nulo é uma palavra-código.

Pela definição da matriz $G$, qualquer $c = aG$ com $a \in \mathbb{F}_q^k$ é uma palavra-código. Portanto, ambas as matrizes $G$ e $H$ podem ser usadas para obter $C$.

Em seguida, veremos exemplos de códigos lineares.

**Exemplo 5.1.2.** [Código verificador de paridade]

Este é um dos códigos mais básicos e usados. Definimos o esquema de codificação binária $f$ da seguinte maneira:

$$(a_1 a_2 \cdots a_k) \quad \longmapsto \quad (c_1 c_2 \cdots c_k c_{k+1}),$$

onde $a_i = c_i$ para $i = 1, \ldots, k$ e $c_{k+1} = \sum_{i=1}^{k} a_i$. O último dígito $c_{k+1}$ é chamado o *símbolo de controle* e serve como um verificador de paridade pois, em $\mathbb{F}_2$,

$$\sum_{i=1}^{k+1} c_i = \sum_{i=1}^{k} a_i + c_{k+1} = 2\sum_{i=1}^{k} a_i = 0.$$

Assim, a soma dos dígitos recebidos deve ser 0; caso contrário, há um erro. Este tipo de código detecta um erro, mas não o corrige, pois não há como identificar a coordenada do erro. Se há dois erros, eles compensarão a paridade. Logo, este código não pode detectar dois ou mais erros.

Este modelo é um código linear binário $(k+1, k)$ com $H = (11\cdots1)$ e $G = (I_k \ \mathbf{1})$.

**Exemplo 5.1.3.** [Código de repetição]

A proposta deste código é oposta à dos códigos verificadores de paridade. No *código de repetição*, cada palavra-código contém um símbolo de mensagem e $n-1$ símbolos de controle, de maneira que

$$c_2 = c_3 = \cdots = c_n = a_1.$$

Logo, o esquema de codificação $f\colon \mathbb{F}_q \to \mathbb{F}_q^n$ é dada por

$$a_1 \quad \longmapsto \quad (a_1 a_1 \cdots a_1).$$

Este é um código linear $(n, 1)$ com matriz de paridade da forma $H = (-\mathbf{1}, I_{n-1})$ e $G = (11\cdots1)$.

O sistema $Hc^T = 0$ é formado pelas equações

$$\begin{array}{rcl} -c_1 + c_2 & = & 0 \\ -c_1 + c_3 & = & 0 \\ \vdots & & \vdots \\ -c_1 + c_n & = & 0, \end{array}$$

o que implica que $c_1 = c_2 = \cdots = c_n$.

Este código detecta até $n-1$ erros, já que se há quaisquer dois símbolos com valores diferentes, deve haver algum erro. Se todos os símbolos são os mesmos, então não podemos detectar se houve ou

não mudanças na transmissão, isto é, se todos os $n$ símbolos foram trocados. Assim não podemos detectar $n$ erros.

Por outro lado, o código de repetição pode corrigir até $\lfloor (n-1)/2 \rfloor$ erros. De fato, se há no máximo $\lfloor (n-1)/2 \rfloor$ erros, então é possível deduzir a mensagem original corretamente, considerando o símbolo mais comum na seqüência recebida como sendo o símbolo enviado.

Por exemplo, suponhamos que $n = 10$ e que a mensagem original seja 1. Então se recebermos três 0's e sete 1's, saberemos que houve um erro, simplesmente pelo fato de haver símbolos diferentes. Como há no máximo $\lfloor (10-1)/2 \rfloor = 4$ erros, sabemos que a mensagem original é 1. Se há mais que quatro erros, o erro não pode ser identificado corretamente, mas a nossa hipótese é que não há mais que quatro erros.

### 5.1.1 Distância de Hamming

Para corrigir erros de códigos em geral, precisamos de uma noção de *distância* entre vetores.

**Definição 5.1.4.** Sejam $x$ e $y$ vetores em $\mathbb{F}_q^n$. A *distância de Hamming* $d(x, y)$ é o número de coordenadas nas quais $x$ e $y$ se diferem.

A distância de Hamming é uma *métrica* em $\mathbb{F}_q^n$, isto é, satisfaz as seguintes propriedades: para quaisquer $x, y, z \in \mathbb{F}_q^n$, temos

1. $d(x, y) \geq 0$,
2. $d(x, y) = 0$ se e somente se $x = y$,
3. $d(x, y) = d(y, x)$,
4. $d(x, z) \leq d(x, y) + d(y, z)$.

Um conceito relacionado com a distância de Hamming é o *peso de Hamming*. O peso de Hamming é o número de coordenadas não nulas de $x$ e é denotado por $w(x)$.

**Exemplo 5.1.5.** $d((1,0,1,0,1),(0,1,1,1,0)) = 4$ *e* $w(1,0,0,1,1) = 3$.

A distância e o peso de Hamming estão relacionados pela identidade $d(x,y) = w(x-y)$.

A seguir definimos *erro* e *códigos que corrigem* $t$ *erros*.

**Definição 5.1.6.** Se $c$ é uma palavra-código e $y$ é uma palavra recebida, então o *erro* é a diferença $e = y - c$.

**Definição 5.1.7.** Seja $t$ um inteiro positivo. Um código $C \in \mathbb{F}_q^n$ *corrige* $t$ *erros* se, para cada palavra recebida $y \in \mathbb{F}_q^n$, há no máximo um $c \in C$ tal que $d(c,y) \leq t$.

Dito de outra maneira, se uma bola fechada de raio $t$ centrada em $y$ contém pelo menos duas palavras-código, o código não corrige $t$ erros.

A correção do erro é feita por *verossimilhança*, que simplesmente consiste em procurar pela palavra-código mais próxima possível. Esta noção de proximidade será formalizada com a seguinte definição.

**Definição 5.1.8.** A *distância mínima* de um código $C$ é definida por

$$d_C = \min_{\substack{u,v \in C \\ u \neq v}} d(u,v) = \min_{\substack{c \in C \\ c \neq 0}} w(c). \tag{5.1}$$

Agora podemos determinar o número máximo de erros que um código pode corrigir.

**Teorema 5.1.9.** *Um código* $C$ *com distância mínima* $d_C$ *pode corrigir até* $t$ *erros, se* $d_C \geq 2t+1$.

*Demonstração.* A bola fechada $B_t(x)$ de raio $t$ e centro $x \in \mathbb{F}_q^n$ é formada por todos os vetores $y \in \mathbb{F}_q^n$ tais que $d(x,y) \leq t$. A regra de decodificação por verossimilhança garante que cada palavra recebida com no máximo $t$ erros deve estar numa bola com centro na palavra transmitida e raio $t$. Suponhamos que $u \in B_t(x) \cap B_t(y)$ onde $x, y \in C$. Então $d(x,y) \leq d(x,u) + d(u,y) \leq 2t$, o que é uma contradição, pois $d_C \geq 2t+1$. □

Para sabermos se um código corrige $t$ erros, devemos investigar $d_C$. Calcular $d_C$ percorrendo as $q^k$ palavras-código pode ser uma

tarefa bem cara. Uma alternativa é proposta com o seguinte resultado.

**Teorema 5.1.10.** *Um código linear $C$ com matriz de paridade $H$ tem distância mínima $d_C \geq s+1$ se e somente se quaisquer $s$ colunas de $H$ são linearmente independentes.*

*Demonstração.* Suponhamos que $H$ tenha $s$ colunas linearmente dependentes, digamos, $D_{i_1}, D_{i_2}, \ldots, D_{i_s}$. Então existem constantes $\alpha_{i_1}$, $\alpha_{i_2}, \ldots, \alpha_{i_s}$, não todas nulas, tais que $\sum_{j=1}^{s} \alpha_{i_j} D_{i_j} = 0$.

Seja $c \in \mathbb{F}_q^n$ tal que $c_{i_j} = \alpha_{i_j}$ para $j = 1, 2, \ldots, s$ e todas as outras coordenadas de $c$ são nulas. Temos que $Hc^T = \sum_{j=1}^{s} \alpha_{i_j} D_{i_j} = 0$. Logo $c \in C$. Além disso, $c \neq 0$ e $w(c) \leq s$, ou seja, $d_C \leq s$.

Por outro lado, se quaisquer $s$ colunas de $H$ são linearmente independentes, então não existe $c \in C$ não nulo de peso no máximo $s$. De fato, se existe uma palavra-código $c \neq 0$ com $w(c) \leq s$, então $\sum_{j=1}^{s} c_{i_j} H_{i_j} = 0$ implica que $c_{i_j} = 0$ para todo $j = 1, 2, \ldots, s$. Assim, o único $c \in C$ com $w(c) \leq s$ é $c = 0$. □

No próximo exemplo, combinaremos os dois últimos resultados para achar o número máximo de erros que o código pode corrigir.

**Exemplo 5.1.11.** Vamos considerar a matriz de paridade sobre $\mathbb{F}_2$:

$$H = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}. \tag{5.2}$$

Quaisquer duas colunas de $H$ são linearmente independentes. Pelo Teorema 5.1.10, $d_C \geq 3$. No entanto, como

$$\text{coluna } 4 = \text{coluna } 3 + \text{coluna } 5,$$

temos que $d_C < 4$, ou seja, $d_C = 3$. Logo, pelo Teorema 5.1.9, o código $C$ corrige um erro.

### 5.1.2 Decodificação de códigos lineares

Consideramos agora o problema de decodificar a palavra recebida $y$. Uma possibilidade é estudar a distância de $y$ a cada palavra-código e corrigir a mensagem pela palavra-código de menor distância.

Este método é impraticável para códigos com dimensão $k$ grande já que requer $q^k$ computações. Nesta seção, veremos um método mais eficiente para decodificar palavras recebidas.

Consideramos um código linear $C(n,k)$ sobre $\mathbb{F}_q$. Sejam $0 = c^{(1)}$, $c^{(2)}, \ldots, c^{(q^k)}$ as palavras-código. Observamos que $C$ é um ideal de $\mathbb{F}_q^n$. O anel quociente $\mathbb{F}_q^n/C$ é formado pelas classes laterais $a + C = \{a + c : c \in C\}$, onde $a \in \mathbb{F}_q^n$. O número de elementos em cada classe lateral é $\#C = q^k$ e o número de classes laterais é $q^{n-k}$. Particionamos $\mathbb{F}_q^n$ como

$$\mathbb{F}_q^n = (a^{(0)} + C) \uplus (a^{(1)} + C) \uplus \cdots \uplus (a^{(s)} + C),$$

onde $a^{(0)} = 0$ e $s = q^{n-k} - 1$. Seja $y \in \mathbb{F}_q^n$ uma palavra recebida, digamos, $y \in a^{(i)} + C$, ou mais especificamente, $y = a^{(i)} + c^{(j)}$, onde $0 \leq i \leq s$ e $1 \leq j \leq q^k$. Suponhamos que a palavra transmitida seja $c$. Então o erro é $e = y - c = a^{(i)} + c^{(j)} - c$. Como $C$ é um espaço vetorial, é fechado para a adição. Portanto, o erro pertence à mesma classe lateral que a mensagem recebida.

A seguir veremos como achar as classes laterais. Primeiro introduzimos o seguinte conceito.

**Definição 5.1.12.** Sejam $H$ uma matriz de paridade de um código linear $C(n,k)$ e $y \in \mathbb{F}_q^n$. O vetor $S(y) = Hy^T \in \mathbb{F}_q^{n-k}$ é chamado a *síndrome* de $y$.

O próximo resultado será fundamental para os nossos propósitos.

**Teorema 5.1.13.** *Para $y, z \in \mathbb{F}_q^n$, temos que*

1. *$S(y) = 0$ se e somente se $y \in C$,*
2. *$S(y) = S(z)$ se e somente se $y + C = z + C$.*

*Demonstração.* A afirmação (1) segue imediatamente da definição de síndrome. Para provar (2), observamos que $S(y) = S(z)$ se e somente se $H(y-z)^T = 0$. Isto significa que $y - z \in C$, i.e. $y + C = z + C$. □

Pela regra de decodificação por verossimilhança, queremos que o erro seja um elemento de menor peso na classe lateral da mensagem recebida. Isto ficará mais claro mais adiante.

**Definição 5.1.14.** Seja $C(n,k)$ um código linear sobre $\mathbb{F}_q$. Um elemento de peso mínimo numa classe lateral $a+C \in \mathbb{F}_q^n/C$ é chamado um *líder da classe lateral*. Se há mais de um vetor com peso mínimo em $a+C$, escolhemos qualquer um deles para ser o líder.

Vamos supor que os elementos $a^{(1)},\ldots,a^{(s)}$ considerados anteriormente são na verdade os líderes das classes laterais. Usando a notação padrão de vetores, descrevemos a seguir os elementos de $\mathbb{F}_q^n$ distribuídos pelas classes laterais.

| linha / mensagem | $00\ldots0$ | $00\ldots01$ | $\ldots$ | $(q-1)\ldots(q-1)$ |
|---|---|---|---|---|
| palavras-código | $c^{(1)}$ | $c^{(2)}$ | $\ldots$ | $c^{(q^k)}$ |
| | $a^{(1)}+c^{(1)}$ | $a^{(1)}+c^{(2)}$ | $\ldots$ | $a^{(1)}+c^{(q^k)}$ |
| classes | $a^{(2)}+c^{(1)}$ | $a^{(2)}+c^{(2)}$ | $\ldots$ | $a^{(2)}+c^{(q^k)}$ |
| laterais | $\vdots$ | $\vdots$ | $\ddots$ | $\vdots$ |
| restantes | $\underbrace{a^{(s)}+c^{(1)}}_{\text{coluna dos líderes}}$ | $a^{(s)}+c^{(2)}$ | $\ldots$ | $a^{(s)}+c^{(q^k)}$ |

Como o erro é o líder da classe lateral de $y$, temos que $e=a^{(i)}$ e a mensagem transmitida é

$$x = y - e = a^{(i)} + c^{(j)} - a^{(i)} = c^{(j)}.$$

Se soubermos como calcular os líderes da classe lateral, teremos um método eficiente de decodificação. O problema de encontrar um líder de uma classe lateral em $\mathbb{F}_q^n/C$ pode ser resolvido usando a noção de síndrome. Para corrigir erros na mensagem recebida $y$, calculamos $S(y)$ e encontramos os líderes da classe lateral $a^{(i)}$ com síndrome igual a $S(y)$. Então, decodificamos $y$ através de $x=y-a^{(i)}$, onde $x$ é a palavra-código que está a uma distância de $y$ igual à distância mínima do código.

**Exemplo 5.1.15.** Seja $C(4,2)$ um código binário linear com matriz geradora

$$G = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

e matriz de paridade

$$H = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}.$$

Há 4 mensagens distintas, 4 palavras-código e 4 classes laterais. A seguinte tabela contém os elementos de $\mathbb{F}_{2^4}$ particionados em classes segundo este código. Em seguida, explicaremos a distribuição desta tabela.

| linha mensagem | 00 | 01 | 10 | 11 | Síndrome |
|---|---|---|---|---|---|
| palavras-código | 0000 | 0110 | 1011 | 1101 | $\binom{0}{0}$ |
| | 0001 | 0111 | 1010 | 1100 | $\binom{0}{1}$ |
| classes | 0010 | 0100 | 1001 | 1111 | $\binom{1}{0}$ |
| laterais restantes | $\underbrace{1000}_{\text{coluna dos líderes}}$ | 1110 | 0011 | 0101 | $\binom{1}{1}$ |

Calculamos as palavras-código usando $aG = c$ ou $Hc^T = 0$. Em seguida, obtemos os elementos das classes laterais restantes, tomando um líder da classe lateral e adicionando a ele cada palavra-código. Observamos que podemos calcular a tabela inteira se conhecemos os líderes das classes laterais e as síndromes.

Para obtermos os líderes das classes laterais, consideramos os vetores, um por um, verificando se a síndrome é nova ou não. Se é nova, armazenamos temporariamente este elemento como o líder da classe lateral. Se já há um líder da classe lateral para aquela síndrome, devemos verificar se o novo elemento tem peso menor que o líder corrente, atualizando a informação sempre que for conveniente. No fim do processo, todos os líderes da classe lateral terão sido calculados. Há várias maneiras de melhorar este algoritmo; por exemplo, considerando os vetores ordenados pelo peso.

Se a palavra recebida é $y = 0101$, então $S(y) = \binom{1}{1}$. Tomamos o líder da classe lateral como sendo o erro e a mensagem transmitida é

$$x = y - e = (1101 + 1000) - 1000 = 1101.$$

A palavra transmitida é 1101 com mensagem original 11.

O seguinte teorema dá uma caracterização das posições dos erros, em termos das colunas da matriz de paridade de códigos binários.

**Teorema 5.1.16.** *Num código binário linear $C(n,k)$ com matriz de paridade $H$, a síndrome é a soma das colunas de $H$ que correspondem às posições onde os erros ocorreram.*

*Demonstração.* Sejam $y \in \mathbb{F}_2^n$ a mensagem recebida, $x$ a palavra-código transmitida e $e$ o erro. Vamos supor que as coordenadas não nulas de $e$ sejam $e_{i_1}, e_{i_2}, \ldots, e_{i_k}$. Temos que

$$\begin{aligned} S(y) &= S(x+e) = S(x) + S(e) = Hx^T + He^T = He^T \\ &= H_{i_1}e_{i_1} + H_{i_2}e_{i_2} + \cdots + H_{i_k}e_{i_k} = H_{i_1} + H_{i_2} + \cdots + H_{i_k}, \end{aligned}$$

onde $H_i$ é a $i$-ésima coluna de $H$. □

Quando o decodificador recebe $y \in \mathbb{F}_2^n$, calcula a síndrome. Se há apenas um erro, ele pode ser corrigido pois, pelo teorema anterior, a síndrome indica a coluna de $H$ em que a posição de $y$ é incorreta. Se há mais que um erro, então o decodificador recebe a soma das colunas e não pode decodificar a mensagem corretamente. Logo temos um código que corrige um erro.

## 5.2 Códigos cíclicos

Um *código cíclico* é um código linear tal que o deslocamento cíclico dos símbolos numa palavra-código produz uma palavra-código.

**Definição 5.2.1.** Um código linear $C(n,k)$ sobre $\mathbb{F}_q$ é *cíclico*, se $(c_0, \ldots, c_{n-1}) \in C$ implica que $(c_{n-1}, c_0, \ldots, c_{n-2}) \in C$.

**Exemplo 5.2.2.** Seja $C = \{000, 110, 101, 011\}$. Por inspeção, $C$ é um código cíclico.

**Exemplo 5.2.3.** Consideramos a seguinte matriz de paridade para o código de Hamming $C_3$ sobre $\mathbb{F}_2$:

$$H = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{pmatrix}.$$

O vetor $c = (0,1,0,1,1,0,0)$ é uma palavra-código, pois

$$Hc^T = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}.$$

O deslocamento cíclico no vetor $c$ produz $\overline{c} = (0,0,1,0,1,1,0)$. Calculamos

$$H\overline{c}^T = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

e concluímos que $\overline{c}$ também é uma palavra-código. De fato, este código é cíclico. Para mostrarmos isso, calculamos a forma geral de uma palavra-código:

$$\begin{pmatrix} r+s+t & s+t+u & r+s+u & r & s & t & u \end{pmatrix},$$

onde $r,s,t,u \in \mathbb{F}_2$. Como

$$H \begin{pmatrix} u & r+s+t & s+t+u & r+s+u & r & s & t \end{pmatrix}^T = 0,$$

temos o resultado desejável.

Consideramos $\mathbb{F}_q^n$ representado por polinômios de grau menor que $n$ em $\mathbb{F}_q[x]/(x^n-1)$, associando o vetor $c = (c_0, \ldots, c_{n-1})$ ao polinômio $c(x) = c_0 + c_1x + \ldots + c_{n-1}x^{n-1}$. O deslocamento cíclico das coordenadas de $c$ corresponde à multiplicação de $c(x)$ por $x$, pois

$$\begin{aligned} xc(x) &= x(c_0 + c_1x + \ldots + c_{n-1}x^{n-1}) \\ &= c_0x + c_1x^2 + \ldots + c_{n-1}x^n \\ &= c_{n-1} + c_0x + c_1x^2 + \ldots + c_{n-2}x^{n-1}, \end{aligned}$$

é o vetor $(c_{n-1}, c_0, c_1, \ldots, c_{n-2})$, já que $x^n = 1$ módulo $x^n - 1$. Usando esta representação polinomial, podemos caracterizar códigos cíclicos algebricamente.

**Teorema 5.2.4.** *Um código linear $C(n,k)$ sobre $\mathbb{F}_q$ é cíclico se e somente se $C$ é um ideal de $\mathbb{F}_q[x]/(x^n-1)$.*

*Demonstração.* Se $C$ é um código cíclico e $c(x) \in C$, temos que $xc(x)$, $x^2c(x)$, $x^3c(x), \ldots$ também pertencem a $C$. Seja $a(x) = \sum_i a_i x^i \in \mathbb{F}_q[x]/(x^n - 1)$. Como $a(x)c(x) = \sum_i a_i(x^i c(x))$ e $C$ é um subespaço vetorial sobre $\mathbb{F}_q$, temos que $C$ é um ideal.

Reciprocamente, se $C$ é um ideal de $\mathbb{F}_q[x]/(x^n - 1)$ e $c(x) = \sum_{i=0}^{n-1} c_i x^i$ é uma palavra-código, então $xc(x)$ também é uma palavra-código. Logo, o código $C$ é cíclico. □

Pelo Teorema 1.3.6, temos que $\mathbb{F}_q[x]$ é um domínio de ideais principais. Assim, todo ideal não nulo em $\mathbb{F}_q[x]/(x^n - 1)$ é gerado por um polinômio mônico $g$ de grau mínimo no ideal. Portanto, pelo teorema anterior, todo código cíclico é gerado por um polinômio.

**Definição 5.2.5.** Seja $C = (g)$ um código cíclico. Dizemos que $g$ é o *polinômio gerador de* $C$ e $h = (x^n - 1)/g$ é o *polinômio verificador de* $C$.

O próximo teorema mostra algumas propriedades do polinômio gerador. Entre elas, está o fato de que $g$ divide $x^n - 1$, mostrando assim que o polinômio verificador está bem definido.

**Teorema 5.2.6.** *Seja $C$ um ideal não nulo em $\mathbb{F}_q[x]/(x^n - 1)$, isto é, $C$ é um código cíclico de comprimento $n$.*

1. *O código $C$ é gerado por um único polinômio mônico $g$ de grau mínimo em $C$.*

2. *O polinômio gerador $g$ de $C$ é um fator de $x^n - 1$.*

3. *Em $\mathbb{F}_q[x]$, qualquer $c \in C$ pode ser escrito unicamente como $c = fg$, onde $\text{grau}(f) < n - r$ e $\text{grau}(g) = r$. Além disso, a dimensão de $C$ é $n - r$.*

4. *Se $g(x) = g_0 + g_1 x + \cdots + g_r x^r$, então $C$ é gerado como um*

*subespaço de $\mathbb{F}_q^n$ pelas linhas da matriz geradora*

$$G = \begin{bmatrix} g_0 & g_1 & \cdot & \cdot & \cdot & g_r & 0 & 0 & 0 \\ 0 & g_0 & g_1 & \cdot & \cdot & \cdot & g_r & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & \cdot & 0 & g_0 & g_1 & \cdot & \cdot & \cdot & g_r \end{bmatrix}$$

$$= \begin{bmatrix} g(x) & 0 & 0 & \cdot & \cdot & \cdot & \cdot & 0 & 0 \\ 0 & xg(x) & 0 & \cdot & \cdot & \cdot & \cdot & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & \cdot & \cdot & \cdot & \cdot & 0 & x^{n-r-1}g(x) \end{bmatrix}.$$

*Demonstração.* A afirmação (1) segue da prova do Teorema 1.3.6. Para mostrar (2), escrevemos $x^n - 1 = hg + r$ onde $r, g \in \mathbb{F}_q[x]$ e $\text{grau}(r) < \text{grau}(g)$. Em $\mathbb{F}_q[x]/(x^n - 1)$, isto implica que $r = -hg$. Como $\text{grau}(r) < \text{grau}(g)$, temos que $r = 0$ e assim $g \mid (x^n - 1)$.

Seja $c \in C$ onde $\text{grau}(c) < n$. Por (2), exite um polinômio $q$ tal que $c = gq$ em $\mathbb{F}_q[x]/(x^n - 1)$. Por (3), suponhamos que $x^n - 1 = gh$. Em $\mathbb{F}_q[x]$, temos que $c = qg + \ell(x^n - 1)$ para algum $\ell \in \mathbb{F}_q[x]$, isto é, $c = (q + \ell h)g$. Seja $f = q + \ell h$. Então $c = fg$ e $\text{grau}(f) \leq n - r - 1$. Assim o código é formado por múltiplos de $g$, que são polinômios de grau no máximo $n - r - 1$ avaliado em $\mathbb{F}_q[x]$ e não em $\mathbb{F}_q[x]/(x^n - 1)$.

Há $n - r$ múltiplos de $g$ linearmente independentes, a saber, $g, xg$, $x^2g, \ldots, x^{(n-r-1)}g$. Os vetores correspondentes são as linhas da matriz geradora $G$. Portanto, o código tem dimensão $n - r$. □

Mostraremos agora o papel do polinômio verificador $h$. Seja $f$ uma mensagem codificada pela multiplicação por $g$:

$$c = fg = \sum_{i=0}^{n-1} c_i x^i.$$

Então $ch = fgh = f(x^n - 1)$, isto é, $ch = 0$ em $\mathbb{F}_q[x]/(x^n - 1)$. Digamos que

$$ch = \left( \sum_{i=0}^{n-1} c_i x^i \right) \left( \sum_{\ell=0}^{k} h_\ell x^\ell \right),$$

onde $h_k \neq 0$. O coeficiente de $x^j$ neste produto é

$$\sum_{i=0}^{n-1} c_i h_{j-i} = 0, \tag{5.3}$$

para $j = 0, 1, \ldots, n-1$, onde os subíndices são tomados módulo $n$. Se $c \in C$, a Equação (5.3) implica que $Hc^T = 0$, onde a matriz de paridade $H$ é dada por

$$H = \begin{bmatrix} 0 & \cdot & \cdot & \cdot & 0 & h_k & h_{k-1} & \cdot & \cdot & \cdot & h_1 & h_0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ h_k & h_{k-1} & \cdot & \cdot & \cdot & h_1 & h_0 & 0 & \cdot & \cdot & \cdot & 0 \end{bmatrix}.$$

Em notação polinomial usada para $G$, temos

$$H = \begin{bmatrix} 0 & 0 & 0 & \cdot & \cdot & \cdot & \cdot & 0 & h(x) \\ 0 & 0 & 0 & \cdot & \cdot & \cdot & 0 & xh(x) & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ x^{n-k-1}h(x) & 0 & 0 & \cdot & \cdot & \cdot & \cdot & 0 & 0 \end{bmatrix}.$$

A dimensão do código é $\dim\ C = n - \text{grau}(g) = \text{grau}(h) = k$.

Lembramos que, pelo Teorema 3.5.8, se $i \in C_s$ então

$$\prod_{j \in C_s} (x - \alpha^j) = M^i(x).$$

A seguir, generalizaremos a construção ciclotômica para $n \neq p^m - 1$ com $\text{mdc}(n, q) = 1$. A classe lateral ciclotômica módulo $n$ sobre $\mathbb{F}_q$ que contém $s$ é

$$C_s = \{s, qs, q^2 s, \ldots, q^{m_s - 1} s\},$$

onde $q^{m_s} s \equiv s \pmod{n}$. Inteiros módulo $n$ são particionados em classes laterais ciclotômicas.

Como antes, digamos que o polinômio minimal de $\alpha^s$ é

$$M^s(x) = \prod_{j \in C_s} (x - \alpha^j),$$

isto é, $M^s$ é um polinômio mônico com coeficientes em $\mathbb{F}_q$ e com o menor grau entre os polinômios tendo $\alpha^s$ como uma raiz. Além disso,

$$x^n - 1 = \prod_s M^{(s)}(x),$$

onde $s$ percorre um conjunto de representantes de classes laterais módulo $n$. Esta também é a fatoração de $x^n - 1$ como um produto de irredutíveis, já que polinômios minimais são irredutíveis sobre $\mathbb{F}_q$. Finalmente, o grau do polinômio minimal $M^{(i)}$ é igual ao número de elementos na classe lateral ciclotômica contendo $i$.

**Exemplo 5.2.7.** Para $n = 7$, $q = 2$ e $m = 3$, temos

$$C_0 = \{0\}, C_1 = \{1, 2, 4\}, C_3 = \{3, 6, 5\},$$

logo

$$x^7 - 1 = M^{(0)}M^{(1)}M^{(3)} = (1 + x)(1 + x^2 + x^3)(1 + x + x^3),$$

onde $(1 + x^2 + x^3)$ e $(1 + x + x^3)$ são polinômios irredutíveis de grau 3 sobre $\mathbb{F}_2$.

A seguir, mostraremos duas aplicações dos polinômios minimais na teoria de códigos: os códigos de Hamming e os códigos BCH que corrigem dois erros.

### 5.2.1 Códigos de Hamming

**Definição 5.2.8.** Seja $m$ um inteiro maior ou igual a 2. Um código binário $C_m$ de comprimento $n = 2^m - 1$ com uma matriz de paridade $H$ de ordem $m \times (2^m - 1)$ é chamado um *código binário de Hamming*, se as colunas de $H$ correspondem às representações binárias dos inteiros $1, 2, \ldots, 2^m - 1$.

**Exemplo 5.2.9.** O código binário de Hamming $C_3$ tem comprimento 7 e matriz de paridade

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}.$$

Observamos que a matriz $H$ tem posto $m$. Portanto a dimensão de $C_m$ é $2^m - 1 - m$, usando o teorema do posto e da nulidade. Quaisquer duas colunas são linearmente independentes, já que nenhuma coluna é

múltipla da outra. Por outro lado, há três colunas que são linearmente dependentes. Por exemplo,

$$\begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 0 \\ \vdots \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 1 \\ 1 \end{pmatrix}.$$

Pelos Teoremas 5.1.9 e 5.1.10, a distância mínima do código é $2+1 = 3$ e o código corrige um erro.

O seguinte teorema resume as observações anteriores e caracteriza a matriz geradora para o código de Hamming.

**Teorema 5.2.10.** *O código de Hamming com parâmetros $n = 2^m - 1$, $k = n - m$ e $d = 3$ é um código cíclico com polinômio gerador $g = M^{(1)}$. Uma matriz geradora para o código é*

$$G = \begin{pmatrix} M^{(1)} & 0 & \cdot & \cdot & \cdot & 0 \\ 0 & xM^{(1)} & \cdot & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & \cdot & \cdot & \cdot & \cdot & x^{n-m-1}M^{(1)} \end{pmatrix}.$$

*Demonstração.* Consideramos o código binário de Hamming de comprimento $n = 2^m - 1$. A matriz de paridade contém $2^m - 1$ colunas que são as $m-$uplas não nulas distintas em $\mathbb{F}_2$. Seja $\alpha$ um elemento primitivo de $\mathbb{F}_{2^m}$. Então $1, \alpha, \alpha^2, \ldots, \alpha^{2^m-1}$ são distintos e podem ser representados por $m-$uplas não nulas distintas. Assim, o código binário de Hamming com parâmetros $n = 2^m - 1$, $k = n - m$, $d = 3$ tem uma matriz de paridade

$$H = (1, \alpha, \alpha^2, \ldots, \alpha^{2^m-2}),$$

onde cada entrada $\alpha^i$ é substituída pelo vetor correspondente de $m$ 0's e 1's.

Um vetor $c = (c_0, c_1, \ldots, c_{n-1})$ pertence ao código de Hamming se e somente se $Hc^T = 0$. Usando a correspondência do vetor $(c_0, c_1, \ldots, c_{n-1})$ com o polinômio $c_0 + \cdots + c_{n-1}x^{n-1}$, temos que

$$\begin{array}{lll} (c_0, c_1, \ldots, c_{n-1}) \in C & \iff & Hc^T = 0 \\ & \iff & \sum_{i=0}^{n-1} c_i \alpha^i = 0 \\ & \iff & c(\alpha) = 0. \end{array}$$

Como $\alpha$ é uma raiz de $c$, temos que o polinômio minimal de $\alpha$ deve dividir $c$, isto é, $M^{(1)} \mid c$. Em outras palavras, um código de Hamming é formado pelos múltiplos de $M^{(1)}$. □

**Exemplo 5.2.11.** Para $n = 2^3 - 1 = 7$ e $m = 3$, a matriz geradora é

$$G = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix},$$

já que $M^{(1)}(x) = 1 + x + x^3$ é o polinômio minimal dos elementos na classe lateral ciclotômica $C_1 = \{1, 2, 4\}$ módulo 7. Cada elemento restante na matriz é zero.

Se não conhecemos a matriz de paridade $H$, podemos usar o polinômio verificador para encontrá-la. Como $h(x) = (x^7 - 1)/(x^3 + x + 1) = x^4 + x^2 + x + 1$, temos que

$$H = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{pmatrix}.$$

### 5.2.2 Códigos BCH que corrigem dois erros

Acabamos de ver que os códigos de Hamming podem corrigir um erro. Introduziremos agora uma variação conhecida como *códigos BCH que corrigem dois erros*. Estes códigos foram desenvolvidos com o objetivo de corrigir dois erros usando uma matriz similar à dos códigos de Hamming.

Como na seção anterior, suponhamos que $\alpha$ é um elemento primitivo de $\mathbb{F}_{2^m}$. Então, os elementos $1, \alpha, \alpha^2, \ldots, \alpha^{2^m-1}$ são distintos e podem ser representados por $m-$uplas não nulas distintas.

A idéia foi considerar uma matriz de paridade com $2m$ linhas (ao invés de $m$ linhas) de tal maneira que as primeiras $m$ linhas fossem iguais às da matriz do código de Hamming, enquanto que as $m$ linhas restantes tivessem o propósito de corrigir dois erros.

**Definição 5.2.12.** Seja $m$ um inteiro maior ou igual a 2. Definimos o *código binário BCH que corrige dois erros* como sendo o código $C$ de comprimento $n = 2^m - 1$ com matriz de paridade de ordem $2m \times (2^m - 1)$ dada por

$$H = \begin{pmatrix} 1 & \alpha & \alpha^2 & \cdots & \alpha^{2^m-2} \\ 1 & \alpha^3 & \alpha^6 & \cdots & \alpha^{3(2^m-2)} \end{pmatrix},$$

onde $\alpha$ é um elemento primitivo de $\mathbb{F}_{2^m}$.

Neste caso, temos

$$\begin{array}{rcl} c \in C & \iff & Hc^T = 0 \\ & \iff & \sum_{i=0}^{n-1} c_i \alpha^i = 0 \text{ e } \sum_{i=0}^{n-1} c_i \alpha^{3i} = 0 \\ & \iff & c(\alpha) = 0 \text{ e } c(\alpha^3) = 0 \\ & \iff & M^{(1)} \mid c \text{ e } M^{(3)} \mid c, \end{array}$$

onde $M^{(3)}$ é o polinômio minimal de $\alpha^3$. Como $M^{(1)}$ e $M^{(3)}$ são irredutíveis e distintos, temos que $c \in C$ se e somente se $M^{(1)}M^{(3)}$ divide $c$. Provamos assim que $M^{(1)}M^{(3)}$ é um gerador do código binário BCH que corrige dois erros. Não é difícil demonstrar que, quando $p = 2$ e $n = 2^m - 1$ com $m \geq 3$, temos $\text{grau}(M^{(1)}) = \text{grau}(M^{(3)}) = m$ (exercício). Assim, $\text{grau}(g) = 2m$ e $k = n - 2m$. A condição $k = n - 2m$ pode ser obtida verificando que o número de linhas de $H$ é $n - k = 2m$ ou que $k = \text{grau}(h) = n - \text{grau}(g) = n - 2m$. Temos então o seguinte resultado, onde a parte da distância será demonstrada mais tarde.

**Teorema 5.2.13.** *O código binário BCH que corrige dois erros com parâmetros $n = 2^m - 1$, $k = n - 2m$, $d \geq 5$ e $m \geq 3$ é um código cíclico com polinômio gerador $g = M^{(1)}M^{(3)}$, onde* $\text{grau}(M^{(1)}) = \text{grau}(M^{(3)}) = m$.

**Exemplo 5.2.14.** Consideremos o código binário BCH que corrige dois erros com comprimento $n = 15$, $m = 4$ e

$$H = \begin{pmatrix} 1 & \alpha & \alpha^2 & \cdots & \alpha^i & \cdots & \alpha^{14} \\ 1 & \alpha^3 & \alpha^6 & \cdots & \alpha^{3i} & \cdots & \alpha^{3(14)} \end{pmatrix}.$$

Sejam $\mathbb{F}_{2^4} \cong \mathbb{F}_2[x]/(x^4+x+1)$ e $\alpha$ um elemento primitivo de $\mathbb{F}_{2^4}$. As classes laterais ciclotômicas módulo 15 são

$$C_0 = \{0\},\ C_1 = \{1,2,4,8\},\ C_3 = \{3,6,12,9\},$$
$$C_5 = \{5,10\},\ C_7 = \{7,14,13,11\}.$$

Como $\alpha$ é primitivo, o polinômio minimal de $\alpha$ é $x^4+x+1$. O polinômio minimal de $\alpha^3$ é $x^4+x^3+x^2+x+1$, então

$$g(x) = (x^4+x+1)(x^4+x^3+x^2+x+1).$$

O polinômio verificador é $h(x) = (x^{15}-1)/g(x)$ e agora é fácil de se obter a matriz geradora e a matriz de paridade do código.

## 5.3 Códigos BCH que corrigem $t$ erros

Em geral, é muito difícil encontrar a distância mínima $d$ de um certo código. A seguir, mostraremos uma cota inferior para $d$, dado que os zeros do polinômio gerador $g$ sejam conhecidos.

**Definição 5.3.1.** Seja $C$ um código cíclico com polinômio gerador $g$. Como $g \mid (x^n-1)$, temos que

$$g(x) = \prod_{j \in K} (x - \alpha^j),$$

onde $K$ é a união de algumas classes laterais ciclotômicas. Assim, $\alpha^i$ é um *elemento nulo do código* se $i \in K$. Caso contrário, $\alpha^i$ é um *elemento não nulo do código.*

Observamos que os elementos não nulos do código são os zeros do polinômio $h$.

No próximo resultado, consideramos polinômios geradores $g$ que tem como raízes uma seqüência de potências consecutivas de uma raiz $n$-ésima primitiva da unidade, digamos $\alpha$.

**Teorema 5.3.2.** [Cota BCH] *Seja $C$ um código cíclico com polinômio gerador $g$ tal que, para certos inteiros $b \geq 0$ e $D \geq 1$, temos que*

$$g(\alpha^b) = g(\alpha^{b+1}) = \cdots = g(\alpha^{b+D-2}) = 0.$$

*Então, a distância mínima $d$ do código satisfaz $d \geq D$.*

*Demonstração.* Como o polinômio gerador é zero em $\alpha^b, \ldots, \alpha^{b+D-2}$, temos que para $c = (c_0, c_1, \ldots, c_{n-1}) \in C$

$$c(\alpha^b) = c(\alpha^{b+1}) = \cdots = c(\alpha^{b+D-2}) = 0.$$

Isto implica que a matriz

$$H' = \begin{pmatrix} 1 & \alpha^b & \alpha^{2b} & \cdots & \alpha^{(n-1)b} \\ 1 & \alpha^{b+1} & \alpha^{2(b+1)} & \cdots & \alpha^{(n-1)(b+1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{b+D-2} & \alpha^{2(b+D-2)} & \cdots & \alpha^{(n-1)(b+D-2)} \end{pmatrix}$$

satisfaz $H'c^T = 0$.

Vamos provar que quaisquer $D-1$ colunas de $H'$ são linearmente independentes e então, pelo Teorema 5.1.10, temos que $d \geq D$. Se pudéssemos provar que existem $D$ colunas linearmente dependentes, então a distância mínima $d$ seria exatamente $D$.

Suponhamos por contradição que $c$ tenha peso no máximo $D-1$, digamos, $c_i = 0$ se e somente se $i \in \{a_1, a_2, \ldots, a_\ell\}$. Então $H'c^T = 0$ implica que a matriz formada pelas colunas de $H'$ que multiplicam $c_i \neq 0$ satisfaz

$$\begin{pmatrix} \alpha^{a_1 b} & \alpha^{a_2 b} & \cdots & \alpha^{a_\ell b} \\ \alpha^{a_1(b+1)} & \alpha^{a_2(b+1)} & \cdots & \alpha^{a_\ell(b+1)} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha^{a_1(b+D-2)} & \alpha^{a_2(b+D-2)} & \cdots & \alpha^{a_\ell(b+D-2)} \end{pmatrix} \begin{pmatrix} c_{a_1} \\ c_{a_2} \\ \vdots \\ c_{a_\ell} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

É importante observar que o número de colunas é no máximo $D-1$.

Temos um sistema homogêneo de equações. Se o determinante da matriz é não nula, a única solução é a solução nula. Se as colunas são linearmente dependentes, então o determinante deve ser igual a zero. Temos

$$\begin{vmatrix} \alpha^{a_1 b} & \alpha^{a_2 b} & \cdots & \alpha^{a_\ell b} \\ \alpha^{a_1(b+1)} & \alpha^{a_2(b+1)} & \cdots & \alpha^{a_\ell(b+1)} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha^{a_1(b+D-2)} & \alpha^{a_2(b+D-2)} & \cdots & \alpha^{a_\ell(b+D-2)} \end{vmatrix}$$

$$= \alpha^{a_1 b + a_2 b + \cdots + a_\ell b} \begin{vmatrix} 1 & 1 & \dots & 1 \\ \alpha^{a_1} & \alpha^{a_2} & \dots & \alpha^{a_\ell} \\ \alpha^{2a_1} & \alpha^{2a_2} & \dots & \alpha^{2a_\ell} \\ \vdots & \vdots & \ddots & \vdots \end{vmatrix} \neq 0.$$

Este é um determinante do tipo *Vandermonde*. Logo o determinante é o produto de $\alpha^{a_i} - \alpha^{a_j}$, para todos $i, j$, e portanto é diferente de zero. Então as colunas são linearmente independentes e $d \geq D$. □

**Exemplo 5.3.3.** O código binário de Hamming tem polinômio gerador $M^{(1)}$, e assim $M^{(1)}(\alpha) = 0$. Os polinômios minimais de $\alpha$ e de $\alpha^p = \alpha^2$ são os mesmos. Então $M^{(1)}(\alpha^2) = 0$ e temos duas potências consecutivas de $\alpha$ que são zero. Pelo teorema anterior, a distância mínima $d$ é pelo menos 3.

**Exemplo 5.3.4.** O código BCH que corrige dois erros tem polinômio gerador $M^{(1)}M^{(3)}$, onde

$$M^{(1)}(\alpha) = M^{(1)}(\alpha^2) = M^{(1)}(\alpha^4) = 0 \quad \text{e}$$

$$M^{(3)}(\alpha^3) = M^{(3)}(\alpha^6) = 0.$$

Temos quatro potências consecutivas de $\alpha$ que são zeros de $g = M^{(1)}M^{(3)}$. Assim, $d \geq 5$ e isto completa a prova do Teorema 5.2.13.

**Definição 5.3.5.** Um código cíclico de comprimento $n$ sobre $\mathbb{F}_q$ é um *código BCH de distância de projeto* $D$ se, para certos inteiros $b \geq 0$ e $D \geq 1$, temos que

$$g(x) = \text{mmc}(M^{(b)}, M^{(b+1)}, \dots, M^{(b+D-2)}).$$

Em outras palavras, $g$ é o polinômio minimal de menor grau sobre $\mathbb{F}_q$ que possui $\alpha^b, \alpha^{b+1}, \dots, \alpha^{b+D-2}$ como zeros, e temos

$$c(\alpha^b) = c(\alpha^{b+1}) = \cdots = c(\alpha^{b+D-2}) = 0$$

se e somente se $c = (c_0, c_1, \dots, c_{n-1}) \in C$.

O teorema anterior implica que $d \geq D$. Se $D \geq 2t+1$, podemos corrigir $t$ erros. Logo, isto nos permite obter códigos com distância de projeto $D$ e capacidade de corrigir $t$ erros.

Vamos estudar a matriz de paridade $H$. Temos que $c \in C$ se e somente se $c(\alpha^b) = c(\alpha^{b+1}) = \cdots = c(\alpha^{b+D-2}) = 0$, ou seja,

$$H = \begin{pmatrix} 1 & \alpha^b & \alpha^{2b} & \dots & \alpha^{(n-1)b} \\ 1 & \alpha^{b+1} & \alpha^{2(b+1)} & \dots & \alpha^{(n-1)(b+1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{b+D-2} & \alpha^{2(b+D-2)} & \dots & \alpha^{(n-1)(b+D-2)} \end{pmatrix},$$

onde cada entrada é substituída pela coluna correspondente de $m$ elementos em $\mathbb{F}_q$. Temos que $\text{grau}(g) = n - \dim(C)$. Como

$$g = \text{mmc}(M^{(b)}, M^{(b+1)}, \dots, M^{(b+D-2)})$$

e $\text{grau}(M^{(i)}) \leq m$, temos que $\text{grau}(g) \leq m(D-1)$ e assim $\dim(C) \geq n - m(D-1)$. Provamos assim o seguinte teorema.

**Teorema 5.3.6.** *Um código BCH sobre $\mathbb{F}_q$ de comprimento $n$ e distância de projeto $D$ tem distância mínima $d \geq D$ e dimensão $\geq n - m(D-1)$.*

Em certos casos particulares, os código BCH tem nomes especiais.

1. Se $b = 1$, o código é chamado *BCH no sentido estrito*.
2. Se $n = q^m - 1$, o código BCH é chamado *primitivo*. Neste caso, $\alpha$ é um elemento primitivo.
3. Se $n = q - 1$, o código BCH é chamado o *código Reed-Solomon* (esses códigos serão analisados com detalhes na Seção 5.3.1).

Damos a seguir exemplos de códigos BCH no sentido estrito e primitivo. Esses exemplos aparecem em [92].

**Exemplo 5.3.7.** Esta é uma lista de todos os códigos BCH (binário, no sentido estrito, primitivo) de comprimento 15:

| distância de projeto | polinômio gerador | expoentes das raízes de $g$ | dimensão $n - \text{grau}(g)$ | distância real |
|---|---|---|---|---|
| 1 | 1 | - | 15 | 1 |
| 3 | $M^{(1)}(x)$ | 1, 2, 4, 8 | 11 | 3 |
| 5 | $M^{(1)}M^{(3)}$ | 1-4, 6, 8, 9, 12 | 7 | 5 |
| 7 | $M^{(1)}M^{(3)}M^{(5)}$ | 1-6, 8-10, 12 | 5 | 7 |
| 9, 11, 13, 15 | $M^{(1)}M^{(3)}M^{(5)}M^{(7)}$ | 1-14 | 1 | 15 |

**Exemplo 5.3.8.** Esta é uma lista de todos os códigos BCH (binário, no sentido estrito, primitivo) de comprimento 31:

| distância de projeto | polinômio gerador | dimensão $n - \text{grau}(g(x))$ | distância real |
|---|---|---|---|
| 1 | 1 | 31 | 1 |
| 3 | $M^{(1)}(x)$ | 26 | 3 |
| 5 | $M^{(1)}M^{(3)}$ | 21 | 5 |
| 7 | $M^{(1)}M^{(3)}M^{(5)}$ | 16 | 7 |
| 9 or 11 | $M^{(1)}M^{(3)}M^{(5)}M^{(7)}$ | 11 | 11 |
| 13 or 15 | $M^{(1)}M^{(3)}M^{(5)}M^{(7)}M^{(11)}$ | 6 | 15 |
| 17, 19, ..., 31 | $M^{(1)}M^{(3)}M^{(5)}M^{(7)}M^{(11)}M^{(15)}$ | 1 | 31 |

**Exemplo 5.3.9.** Seja $q = 2$. Consideramos $n = 23$ (não primitivo). Começamos calculando as classes laterais ciclotômicas:

$$\begin{aligned} C_0 &= \{0\}, \\ C_1 &= \{1, 2, 4, 8, 16, 9, 18, 13, 3, 6, 12\}, \\ C_5 &= \{5, 10, 20, 17, 11, 22, 21, 19, 15, 7, 14\}. \end{aligned}$$

Isto implica que

$$\begin{aligned} |C_1| = 11, \quad \text{grau}(M^{(1)}) = 11 \qquad \text{e} \\ |C_5| = 11, \quad \text{grau}(M^{(5)}) = 11. \end{aligned}$$

A fatoração de $x^{23} - 1$ é

$$x^{23} - 1 = (x - 1)M^{(1)}M^{(5)}.$$

Calculamos os polinômios minimais $M^{(1)}$ e $M^{(5)}$:

$$\begin{aligned} M^{(1)} &= x^{11} + x^9 + x^7 + x^6 + x^5 + x + 1, \\ M^{(5)} &= x^{11} + x^{10} + x^6 + x^5 + x^4 + x^2 + 1. \end{aligned}$$

Finalmente temos a seguinte tabela.

| distância de projeto | polinômio gerador | dimensão $n - \text{grau}(g)$ | distância real |
|---|---|---|---|
| 1 | 1 | 23 | 1 |
| 3 ou 5 | $M^{(1)}$ | 12 | 7 |
| 7, 9, ..., 23 | $M^{(1)}M^{(5)}$ | 1 | 23 |

### 5.3.1 Códigos Reed-Solomon

Códigos Reed-Solomon tem, por definição, comprimento $n$ igual a $q-1$. Não consideramos $q = 2$, porém o corpo pode ser uma extensão de $\mathbb{F}_2$.

Os códigos Reed-Solomon são usados na prática em muitos problemas. Por exemplo, ele são usados na comunicação pela NASA e pela Agência Espacial Européia. Em algumas aplicações, os códigos Reed-Solomon são usados em combinação com outros códigos, tais como os códigos convolucionais; veja [95], por exemplo. São também usados para recuperar erros em CDs.

Como $n = q - 1$, temos que

$$x^n - 1 = x^{q-1} - 1 = \prod_{\beta \in \mathbb{F}_q^*} (x - \beta).$$

Isto implica que o polinômio minimal de $\alpha^i$ seja $M^{(i)}(x) = x - \alpha^i$. Portanto, um código Reed-Solomon de comprimento $n = q - 1$ e distância de projeto $D$ tem polinômio gerador

$$g(x) = (x - \alpha^b)(x - \alpha^{b+1}) \cdots (x - \alpha^{b+D-2}),$$

onde é usual tomar $b = 1$, ou seja, no sentido estrito.

**Exemplo 5.3.10.** Sejam $q = 5$ e $n = q - 1 = 4$. Queremos distância de projeto $D = 3$. Um elemento primitivo em $\mathbb{F}_5$ é $\alpha = 2$. Então $g(x) = (x - \alpha)(x - \alpha^2) = (x - 2)(x - 4) = x^2 + 4x + 3$. Como $k = n - \text{grau}(g) = 4 - 2 = 2$, a dimensão do código é 2. Assim, temos $q^k = 5^2 = 25$ palavras-código. A matriz geradora é

$$G = \begin{pmatrix} 3 & 4 & 1 & 0 \\ 0 & 3 & 4 & 1 \end{pmatrix}.$$

Alguns exemplos de palavras-código são:

$$\begin{aligned}(0\ 0)G &= (0\ 0\ 0\ 0),\\ (1\ 0)G &= (3\ 4\ 1\ 0),\\ (2\ 0)G &= (1\ 3\ 2\ 0),\\ (3\ 0)G &= (4\ 2\ 3\ 0),\\ (4\ 0)G &= (2\ 1\ 4\ 0).\end{aligned}$$

**Exemplo 5.3.11.** Sejam $q = 4$ e $n = q - 1 = 3$. Queremos distância de projeto $D = 2$ e $b = 2$ (não no sentido estrito). Se consideramos $x^2 + x + 1 \in \mathbb{F}_2[x]$ como o polinômio irredutível definindo $\mathbb{F}_{2^2}$, temos $\mathbb{F}_4 = \{0, 1, \alpha, \alpha + 1\}$, onde $\alpha$ é uma raiz de $x^2 + x + 1$ (neste caso, $\beta = \alpha + 1 = \alpha^2$). Então o polinômio gerador é $g(x) = x - \alpha^2 = x - \beta$ e a matriz geradora é $G = \begin{pmatrix} \beta & 1 & 0 \\ 0 & \beta & 1 \end{pmatrix}$.

A dimensão de um código Reed-Solomon é sempre

$$k = n - \text{grau}(g) = n - D + 1.$$

A distância mínima $d$ é, pela cota BCH, pelo menos $D$:

$$d \geq D = n - k + 1.$$

Para códigos Reed-Solomon, tem-se que $d \leq n - k + 1$, logo $d = n - k + 1$. Em geral, códigos com $d = n - k + 1$ são chamados *MDS*[2] (distância máxima separável). Esses códigos conseguem atingir a distância mínima mais alta possível. Assim, os códigos Reed-Solomon são códigos MDS.

[2]Maximum Distance Separable.

# Apêndice A

# Função de Möbius

Ao leitor que deseja aprender mais sobre os assuntos neste e nos próximos apêndices, recomendamos os livros de Coutinho [31], de Moreira e Saldanha [100] e de Santos [116].

**Definição A.0.1.** A *função $\mu$ de Möbius* é a função $\mu\colon \mathbb{N} \to \mathbb{N}$ definida por

$$\mu(n) = \begin{cases} 1 & \text{se } n = 1, \\ (-1)^k & \text{se } n \text{ é um produto de } k \text{ primos distintos}, \\ 0 & \text{caso contrário.} \end{cases}$$

**Exemplo A.0.2.** $\mu(7) = -1$, $\mu(6) = (-1)^2 = 1$, $\mu(12) = 0$

A seguir, apresentamos duas propriedades importantes da função de Möbius.

**Lema A.0.3.** *Para todo $n \in \mathbb{N}$, temos que*

$$\sum_{d|n} \mu(d) = \begin{cases} 1 & \textit{se } n = 1, \\ 0 & \textit{se } n > 1. \end{cases}$$

*Demonstração.* O caso $n = 1$ é trivial, uma vez que $\sum_{d|n} \mu(d) = \mu(1) = 1$. Se $n > 1$, é suficiente considerar a soma $\sum_{d|n} \mu(d)$ para

os valores $d = 1$ e os divisores $d$ de $n$ que têm uma fatoração em números primos distintos, dado que $\mu(d) = 0$ para todos os outros valores de $d$. Suponhamos que $p_1, p_2, \ldots p_k$ sejam os divisores primos distintos de $n$. Temos que

$$\begin{aligned}\sum_{d|n}\mu(d) &= \mu(1) + \sum_{i=1}^{k}\mu(p_i) + \sum_{1\le i_1<i_2\le k}\mu(p_{i_1}p_{i_2}) \\ &\quad + \sum_{1\le i_1<i_2<i_3\le k}\mu(p_{i_1}p_{i_2}p_{i_3}) + \cdots + \mu(p_1p_2\cdots p_k) \\ &= 1 + \binom{k}{1}(-1)^1 + \binom{k}{2}(-1)^2 + \cdots + \binom{k}{k}(-1)^k \\ &= \big(1 + (-1)\big)^k = 0.\end{aligned}$$

□

**Teorema A.0.4.** [Fórmula de inversão de Möbius] *Sejam $f$ e $g$ funções de $\mathbb{N}$ num grupo abeliano aditivo. Então, para todo $n \in \mathbb{N}$, temos que $f(n) = \sum_{d|n} g(d)$ se e somente se $g(n) = \sum_{d|n}\mu(d)f\left(\frac{n}{d}\right)$.*

*Demonstração.* Suponhamos que $f(n) = \sum_{d|n} g(d)$. Se $c$ e $d$ são divisores de $n$, então $c$ divide $n/d$ se e somente se $d$ divide $n/c$. Portanto,

$$\begin{aligned}\sum_{d|n}\mu(d)f\left(\frac{n}{d}\right) &= \sum_{d|n}\mu(d)\sum_{c|\frac{n}{d}}g(c) \\ &= \sum_{c|n}\sum_{d|\frac{n}{c}}\mu(d)g(c) \\ &= \sum_{c|n}g(c)\sum_{d|\frac{n}{c}}\mu(d) \\ &= g(n),\end{aligned}$$

onde o último passo se deve ao fato de que, pelo Lema A.0.3, é suficiente considerar o caso $c = n$. A recíproca é provada de maneira similar. □

# Apêndice B

# Teorema chinês dos restos

Sejam $R$ um domínio de ideais principais, $r_1, \ldots, r_m$ elementos em $R$ relativamente primos dois a dois e $r = r_1 r_2 \cdots r_m$. Para cada $i$, $1 \leq i \leq m$, consideramos o homomorfismo canônico de anéis

$$\begin{array}{rccc} \Pi_i : & R & \longrightarrow & R/(r_i) \\ & a & \longmapsto & a + (r_i). \end{array}$$

Combinando as aplicações acima, construímos um outro homomorfismo

$$\begin{array}{rccc} \Pi = \Pi_1 \times \cdots \times \Pi_m : & R & \longrightarrow & R/(r_1) \times \cdots \times R/(r_m) \\ & a & \longmapsto & (a + (r_1), \ldots, a + (r_m)). \end{array}$$

O *teorema chinês dos restos* diz que $\Pi$ induz um isomorfismo.

**Teorema B.0.5.** [Teorema chinês dos restos] *Com a notação anterior, temos que*

$$R/(r) \cong R/(r_1) \times \cdots \times R/(r_m).$$

*Demonstração.* Por uma indução simples, basta considerarmos o caso $m = 2$. É evidente que $a \in \ker \Pi$ se e somente se $r_1$ e $r_2$ dividem $a$. Como $\mathrm{mdc}(r_1, r_2) = 1$, temos que $r = r_1 r_2$ divide $a$, ou seja, $\ker \Pi = (r)$. Vamos provar agora que o homormofismo $\Pi$ é sobrejetor.

Existem $a$ e $b$ em $R$ tais que $ar_1 + br_2 = 1$, já que $\mathrm{mdc}(r_1, r_2) = 1$. Portanto, $br_2 + (r_1) = 1 + (r_1)$ e $ar_1 + (r_2) = 1 + (r_2)$. Assim, dado

$$(c + (r_1), d + (r_2)) \in R/(r_1) \times R/(r_2),$$

temos que

$$\begin{aligned} \Pi(dar_1 + cbr_2) &= (dar_1 + cbr_2 + (r_1), dar_1 + cbr_2 + (r_2)) \\ &= (cbr_2 + (r_1), dar_1 + (r_2)) \\ &= (c + (r_1), d + (r_2)). \end{aligned}$$

Pelo Teorema 1.1.24, obtemos o isomorfismo desejado. □

# Apêndice C

# Função de Euler

**Definição C.0.6.** A *função de Euler* é uma função $\phi\colon \mathbb{N} \to \mathbb{N}$, onde $\phi(n)$ é definido como sendo o número de inteiros $m$, $1 \leq m \leq n$, com a propriedade de que $m$ e $n$ são relativamente primos.

**Exemplo C.0.7.**

1. $\phi(5) = 4$, $\phi(6) = 2$, $\phi(12) = 4$
2. Seja $\mathbb{Z}_n^*$ o conjunto dos elementos invertíveis em $\mathbb{Z}_n$. Pelo Exemplo 1.1.9 (3), o número de elementos em $\mathbb{Z}_n^*$ é $\phi(n)$.

A função de Euler também pode ser dada em termos de uma fórmula.

**Teorema C.0.8.**
1. *Se $p$ é um primo, então $\phi(p^j) = p^j\left(1 - \frac{1}{p}\right)$ para todo $j \in \mathbb{N}$.*
2. *Se $m$ e $n$ são inteiros relativamente primos, então $\phi(mn) = \phi(m)\phi(n)$.*
3. *Seja $m = p_1^{e_1} \dots p_r^{e_r}$ onde $p_1, \dots, p_r$ são primos distintos. Então $\phi(m) = m\left(1 - \frac{1}{p_1}\right) \cdots \left(1 - \frac{1}{p_r}\right)$.*

*Demonstração.*

Para mostrarmos (1), observamos que, dentre os $p^j$ números menores ou iguais a $p^j$, há exatamente $p^{j-1}$ múltiplos de $p$. Portanto, $\phi(p^j) = p^j - p^{j-1}$.

A segunda afirmação segue do teorema chinês dos restos, de onde obtemos o seguinte isomorfismo de anéis

$$\mathbb{Z}_{mn} \cong \mathbb{Z}_m \times \mathbb{Z}_n.$$

Assim, temos o seguinte isomofismo de grupos de elementos invertíveis

$$\mathbb{Z}_{mn}^* \cong \mathbb{Z}_m^* \times \mathbb{Z}_n^*.$$

Parte (3) resulta imediatamente de (1) e (2). □

# Bibliografia

[1] L. Adleman, "A subexponential algorithm for the discrete logarithm problem with applications", Procedings of 20th IEEE Symp. Foundations of Computer Science, 55-60, 1979.

[2] A. V. Aho, J. E. Hopcroft e J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading MA, 1974.

[3] A. Arwin, "Über Kongruenzen von dem fünften und höheren Graden nach einem Primzahlmodulus", Arkiv för matematik, astronomi och fysik, 14, 1-46, 1918.

[4] D. W. Ash, I. F. Blake e S. A. Vanstone, "Low complexity normal bases", Discrete Applied Mathematics, 25, 191-210, 1989.

[5] J.-P. Aumasson, M. Finiasz, W. Meier e S. Vaudenay, TCHo: a hardware-oriented trapdoor cipher, a aparecer em 12th Australasian Conference on Information Security and Privacy, Townsville, Australia, 2007.

[6] M. Ben-Or, "Probabilistic algorithms in finite fields", Proceedings of the 22nd Annual IEEE Symp. on Foundations of Computer Science, 394-398, 1981.

[7] E. R. Berlekamp, "Factoring polynomials over finite fields", Bell System Technical Journal, 46, 1853-1859, 1967.

[8] E. R. Berlekamp, *Algebraic Coding Theory*, McGraw-Hill, New York, 1968.

[9] E. R. Berlekamp, "Factoring polynomials over large finite fields", Mathematics of Computation, 24, 713-735, 1970.

[10] I. F. Blake, R. Fuji-Hara, R. C. Mullin e S. A. Vanstone, "Computing discrete logarithms in finite fields of characteristic two", SIAM J. Alg. Disc. Meth., 5, 276-285, 1984.

[11] I. F. Blake, S. Gao e R. C. Mullin, "Normal and self-dual normal bases from factorization of $cx^{q+1} + dx^q - ax - b$", SIAM Journal on Discrete Mathematics, 7, 499-512, 1994.

[12] I. F. Blake, R. C. Mullin e S. A. Vanstone, "Computing logarithms in GF($2^n$)", Advances in Cryptology, Lecture Notes in Computer Science 196, Springer-Verlag, 73-82, 1985.

[13] I. F. Blake, G. Seroussi e N. Smart, *Elliptic Curves in Cryptography*, London Mathematical Society, Lecture Notes Series 265, Cambridge University Press, 1999

[14] A. W. Bluher, "A Swan-like theorem", Finite Fields and Their Applications, 12, 128-138, 2006.

[15] R. P. Brent, S. Larvala e P. Zimmermann, "A fast algorithm for testing reducibility of trinomials mod 2 and some new primitive trinomials of degree 3021377", Mathematics of Computation, 72, 1443-1452, 2003.

[16] R. P. Brent, S. Larvala e P. Zimmermann, "A primitive trinomial of degree 6972593", Mathematics of Computation, 74, 1001-1002, 2005.

[17] R. P. Brent e P. Zimmermann, Algorithms for finding almost irreducible and almost primitive trinomials, in "Primes and Misdemeanours: Lectures in Honour of the Sixtieth Birthday of Hugh Cowie Williams", Fields Institute Communication FIC/41, The Fields Institute, Toronto, 91-102, 2004.

[18] D. G. Cantor, "On arithmetical algorithms over finite fields", Journal of Combinatorial Theory Series A, 50, 285-300, 1989.

[19] D. G. Cantor e E. Kaltofen, "On fast multiplication of polynomials over arbitrary algebras", Acta Informatica, 28, 693-701, 1991.

[20] D. G. Cantor e Z. Zassenhaus, "A new algorithm for factoring polynomials over finite fields", Mathematics of Computation, 36, 587-592, 1981.

[21] B. Chor e R. Rivest, "A knapsack-type public key cryptosystem based on arithmetic in finite fields", IEEE Transactions on Information Theory, 34, 901-909, 1988.

[22] M. Christopoulou, T. Garefalakis, D. Panario e D. Thomson, "The trace of an optimal normal element and low complexity normal elements bases", Proceedings of the Workshop on Coding and Cryptography 2007, 79-88, 2007.

[23] S. D. Cohen, "Primitive elements and polynomials with arbitrary traces", Discrete Mathematics, 83, 1-7, 1990.

[24] S. D. Cohen, "Primitive elements and polynomials: existence results", Lecture Notes in Pure and Applied Mathematics, vol. 141, editado por G. L. Mullen e P. J. Shiue, Dekker, New York, 43-55, 1993.

[25] S. D. Cohen e D. Mills, "Primitive polynomials with first and second coefficients prescribed", Finite Fields and Their Applications, 9, 334-350, 2003.

[26] S. D. Cohen e M. Presern, "Primitive polynomials with prescribed second coefficient", Glasgow Mathematical Journal, 48, 281-307, 2006.

[27] H. Cohen, G. Frey e R. Avanzi, *Handbook of Elliptic and Hyperelliptic Curve Cryptography*, Chapman and Hall/CRC, 2006.

[28] D. Coppersmith, "Fast evaluation of logarithms in fields of characteristic two", IEEE Transactions on Information Theory, 30, 587-594, 1984.

[29] D. Coppersmith, "Solving homogeneous linear equations over GF(2) via block Wiedemann algorithm", Mathematics of Computation, 62, 333-350, 1994.

[30] S. Costa, C. C Lavor, M. Muniz e R. M. Siqueira, *Uma Introdução à Teoria de Códigos*, Notas em Matemática Aplicada 21, XXIX CNMAC, 2006.

[31] S. C. Coutinho, *Números Inteiros e Criptografia RSA*, Série Computação e Matemática, IMPA, 2003.

[32] J. Daemen e V. Rijmen, *The Design of Rijndael*, Springer, 2002. Veja também NIST (National Institute of Standards and Technology) disponível em `http://csrc.nist.gov/CryptoToolkit/aes/`.

[33] R. Dahab, D. Hankerson, F. Hu, M. Long, J. Lopez e A. Menezes, "Software multiplication using Gaussian normal bases", IEEE Transactions on Computers, 55, 974-984, 2006.

[34] R. Dahab e J. Lopez, Técnicas criptográficas modernas: algoritmos e protocolos, capítulo 3 do livro das Jornadas de Atualização em Informática (JAI), Rio de Janeiro, SBC, 2007.

[35] L. E. Dickson, *Linear Groups with an Exposition of Galois Field Theory*, Teubner, Leipzig, 1901; Dover, New York, 1958.

[36] W. Diffie e M. E. Hellman, "New directions in cryptography", IEEE Transactions on Information Theory, 22, 644-654, 1976.

[37] M. Drmota e D. Panario, "A rigorous proof of the Waterloo algorithm for the discrete logarithm problem", Designs, Codes and Cryptography, 26, 229-241, 2002.

[38] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms", Advances in Cryptology - Proceedings of CRYPTO 84 (LNCS 196), 10-18, 1985.

[39] S. Fan e W-B. Han, "Primitive polynomials over finite fields of characteristic two", Applicable Algebra in Engineering, Communication and Computing, 14, 381-395, 2004.

[40] P. Flajolet, X. Gourdon e D. Panario, "The complete analysis of a polynomial factorization algorithm over finite fields", Journal of Algorithms, 40, 37-81, 2001.

[41] P. Flajolet e A. Odlyzko, "Singularity analysis of generating functions", SIAM Journal on Discrete Mathematics, 3, 216-240, 1990.

[42] P. Flajolet e R. Sedgewick, *Analytic Combinatorics*, livro em preparação.

[43] G. S. Frandsen, "On the density of normal basis in finite fields", Finite Fields and Their Applications, 6, 23-38, 2000.

[44] E. Galois, "Sur la théorie des nombres", Bulletin des sciences mathématiques Férussac, 13, 428-435, 1830. Veja também Journal de Mathématiques Pures et Appliquées, 11, 398-407, 1846, e Écrits et mémoires d'Évariste Galois, eds. ROBERT BOURGNE e J.-P. AZRA, Gauthier-Villars, Paris, 112–128, 1962.

[45] S. Gao, *Normal Bases over Finite Fields*, Ph.D Thesis, University of Waterloo, Canada, 1993.

[46] S. Gao, "Elements of provable high orders in finite fields", Proceedings of the American Mathatematical Society, 127, 1615-1623, 1999.

[47] S. Gao, J. von zur Gathen e D. Panario, "Gauss periods: orders and cryptographical applications", Mathematics of Computation, 67, 343-352, 1998.

[48] S. Gao, J. von zur Gathen, D. Panario e V. Shoup, "Algorithms for exponentiation in finite fields", Journal of Symbolic Computation, 29, 879-889, 2000.

[49] S. Gao, J. Howell e D. Panario, "Irreducible polynomials of given forms", *Finite fields: theory, applications and algorithms*, R. C. Mullin e G. L. Mullen, (eds), volume 225 of Contemporary Mathematics Series, Amer. Math. Soc., 43-54, 1999.

[50] S. Gao e H. W. Lenstra, "Optimal normal bases", Designs, Codes and Cryptography, 2, 315-323, 1992.

[51] S. Gao e D. Panario, "Density of normal elements", Finite Fields and Their Applications, 3, 141-150, 1997.

[52] T. Garefalakis e D. Panario, "The index calculus method using non-smooth polynomials", Mathematics of Computation, 70, 1253-1264, 2001.

[53] T. Garefalakis e D. Panario, "Polynomials over finite fields free from large and small degree irreducible factors", Journal of Algorithms, 44, 98-120, 2002.

[54] J. von zur Gathen, "Efficient and optimal exponentiation in finite fields", Computational Complexity, 1, 360-394, 1991.

[55] J. von zur Gathen e J. Gerhard, Polynomial factorization over $\mathbb{F}_2$, Mathematics of Computation, 71, 1677-1698, 2002.

[56] J. von zur Gathen e J. Gerhard, *Modern Computer Algebra*, Cambridge University Press, 2nd edition, 2003.

[57] J. von zur Gathen e M. Giesbrecht, "Constructing normal bases in finite fields", Journal of Symbolic Computation, 10, 547-570, 1990.

[58] J. von zur Gathen e D. Panario, "A survey on factoring polynomials over finite fields", Journal of Symbolic Computation, 31, 3-17, 2001.

[59] J. von zur Gathen e V. Shoup, "Computing Frobenius maps and factoring polynomials", Computational Complexity, 2, 187-224, 1992.

[60] E. N. Gilbert, "A comparison of signalling alphabets", Bell System Technical Journal, 31, 504-522, 1952.

[61] S. Golomb, *Shift-Register Sequences*, Aegean Park Press, 1982.

[62] S. Golomb e G. Gong, *Signal Design for Good Correlation: for Wireless Communication, Cryptography, and Radar*, Cambridge University Press, 2005.

[63] G. Gong e A. Youssef, "Cryptographic properties of the Welch-Gong transformation sequence generators", IEEE Transactions on Information Theory, 48, 2837-2846, 2002.

[64] A. W. Hales e D. W. Newhart, "Swan's theorem for binary tetranomials", Finite Fields and Their Applications, 12, 301-311, 2006.

[65] W-B. Han, "The coefficients of primitive polynomials over finite fields", Mathematics of Computation, 65, 331-340, 1996.

[66] D. R. Hankerson, D. G. Hoffman, D. A. Leonard, C. C. Linder, K. T. Phelps, C. A. Rodger e J. R. Wall, *Coding Theory and Cryptography*, Marcel Dekker, 1991.

[67] D. R. Hankerson, A. J. Menezes e S. A. Vanstone, *Guide to Elliptic curve Cryptography*, Springer-Verlag, 2004.

[68] T. Hansen e G. L. Mullen, "Primitive polynomials over finite fields", Mathematics of Computation, 59, 639-643, 1992.

[69] A. Hefez e M. L. T. Villela, *Códigos Corretores de Erros*, Série Computação e Matemática, IMPA, 2002.

[70] K. Hensel, "Über die Darstellung der Zahlen eines Gattungsbereiches für einen beliebigen Primdivisor", Journal für die reine und angewandte Mathematik, 103, 230-237, 1888.

[71] L. H. Hernandez Encinas, J. M. Munoz Masque e A. Q. Queiruga Dios, "Maple implementation of the Chor-Rivest cryptosystem", Proceedings of ICCS'06, LNCS 3992, 438-445, 2006.

[72] C. -H. Hsu, "The distribution of irreducibles in $\mathbb{F}_q[t]$", Journal of Number Theory, 61, 85-96, 1996.

[73] IEEE Standard specifications for public-key cryptography, Technical Report IEEE Std 1361-2000; IEEE Inc., 3 Park Ave., NY 10016-5997, USA.

[74] IEEE, Diffie-Hellman groups for internet key exchange, IEEE 1363, PKCS; ver `http://www.rfc-archive.org/getrfc.php?rfc=3526`

[75] D. Jungnickel, *Finite Fields: Structure and Arithmetics*, B.I. Wissenschaftsverlag, Mannheim, Germany, 1993.

[76] D. Jungnickel e S. A. Vanstone, "On primitive polynomials over finite fields", Journal of Algebra, 124, 337-353, 1989.

[77] E. Kaltofen, "Polynomial factorization 1982–1986", *Computers in Mathematics*, D. V. Chudnovsky e R. D. Jenks (eds), New York. Marcel Dekker, 285-309, 1990.

[78] E. Kaltofen, "Polynomial factorization 1987–1991", Proceedings of LATIN '92, LNCS 583, Springer-Verlag, 294-313, 1992.

[79] E. Kaltofen, "Analysis of Coppersmith's block Wiedemann algorithm for the parallel solution of sparse linear systems", Proceedings of AAECC-10, LNCS 673, Springer-Verlag, 195-212, 1993.

[80] E. Kaltofen e A. Lobo, "Factoring high-degree polynomials by the black box Berlekamp algorithm", Proceedings of ISSAC'94, ACM Press, 90-98, 1994.

[81] E. Kaltofen e V. Shoup, "Subquadratic-time factoring of polynomials over finite fields", Mathematics of Compution, 67, 1179-1197, 1998.

[82] A. Karatsuba e Y. Ofman, "Multiplication of multidigit numbers on automata" (in Russian), Soviet Physics–Doklady, 7, 595-596, 1963.

[83] D. Knuth, *The Art of Computer Programming, Vol.2: Seminumerical Algorithms*, 3 ed., Addison-Wesley, Reading MA, 1997.

[84] A. M. Legendre, "Recherches d'analyse indéterminée", Mémoires de l'Académie Royale des Sciences, 465-559, 1785.

[85] H. W. Lenstra Jr., "On the Chor-Rivest knapsack cryptosystem", Journal of Cryptology, 3, 149-155, 1991.

[86] J. Levine e J. V. Brawley, "Some cryptographic applications of permutation polynomials", Cryptologia, 1, 76-92, 1977.

[87] R. Lidl e W. B. Müller, "A note on polynomials and functions in algebraic cryptography", Ars Combin., 17A, 223-229, 1984.

[88] R. Lidl e W. B. Müller, "Permutation polynomials in RSA-cryptosystems", Proceedings of CRYPTO 83, 293-301, 1984.

[89] R. Lidl e H. Niederreiter, *Introduction to Finite fields and their Applications*. Cambridge University Press, 2nd edition, 1994.

[90] R. Lidl e H. Niederreiter, *Finite Fields*, Cambridge University Press, 1997.

[91] J. H. van Lint, *Introduction to Coding Theory*, Springer-Verlag, 1982.

[92] F. J. MacWilliams e N. J. A. Sloane, *The Theory of Error-correcting Codes*, North-Holland, 1977.

[93] J. L. Massey e J. K. Omura, "Computational Method and Apparatus for Finite Fields Arithmetic", U. S. Patent #4,587,627, May 1986.

[94] A. Masuda, L. Moura, D. Panario e D. Thomson, "Low complexity normal elements over finite fields of characteristic two", preprint.

[95] R. J. McEliece, *The Theory of Information and Coding: a Mathematical Framework for Communications*, Addison-Wesley Publishing, Reading, MA, 1977.

[96] R. J. McEliece, "A public key system based on algebraic coding theory", JPL DSN Progress Report 42-44, 114-116, 1978.

[97] R. J. McEliece, *The Theory of Information and Coding*, Encyclopedia of Mathematics and its Applications, vol. 86, 2002.

[98] A. J. Menezes, P. van Oorschot e S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996.

[99] D. Mills, "Existence of primitive polynomials with three coefficients prescribed", JP Journal of Algebra, Number Theory and Applications, vol. 4, 1 - 22, 2004.

[100] C. G. T. A. Moreira e N. C. Saldanha, *Primos de Mersenne (e Outros Primos Muito Grandes)*, 22° Colóquio Brasileiro de Matemática, IMPA, 1999.

[101] R. C. Mullin, I. M. Onyszchuk, S. A. Vanstone e R. M. Wilson, "Optimal normal bases in $GF(p^n)$", Discrete Applied Mathematics, 22, 149-161, 1988/1989.

[102] Y. Nawaz e G. Gong, "The WG stream cipher", submetido ao projeto ECRYPT Stream Cipher, 2005.

[103] Y. Nawaz e G. Gong, "Preventing chosen IV attack on WG cipher by increasing the length of key/IV setup"; disponível em `http://cr.yp.to/streamciphers/wg/047.pdf`

[104] H. Niederreiter, "A new efficient factorization algorithm for polynomials over small finite fields", Applied Algebra in Engineering, Communication and Computing, 4, 81-87, 1993.

[105] National Institute of Standards and Technology, Computing Security Research Center; ver `http://csrc.nist.gov/csrc/fedstandards.html`

[106] A. Odlyzko, "Discrete logarithms and their cryptographic significance", Advances in Cryptology, Proceedings of Eurocrypt 1984, Lecture Notes in Computer Science 209, Springer-Verlag, 224-314, 1985.

[107] A. Odlyzko, "Asymptotic enumeration methods", *Handbook of Combinatorics*, R. Graham, M. Grötschel, and L. Lovász, Eds., vol. 2. Elsevier, 1063-1229, 1995.

[108] D. Panario, B. Pittel, B. Richmond e A. Viola, "Analysis of Rabin's polynomial irreducibility test", Random Structures and Algorithms, 19, 525-551, 2001.

[109] D. Panario e B. Richmond, "Analysis of Ben-Or's polynomial irreducibility test", Random Structures and Algorithms, 13, 439-456, 1998.

[110] V. Pless, *Introduction to the Theory of Error-Correcting Codes*, Wiley, 1982.

[111] M. Plotkin, "Binary codes with specified minimum distance", IRE Transactions on Information Theory, 6, 445-450, 1960.

[112] M. O. Rabin, "Probabilistic algorithms in finite fields", SIAM Journal on Computing, 9, 273-280, 1980.

[113] A. Reyhani-Masoleh, Efficient algorithms and architectures for field multiplication using Gaussian normal bases, IEEE Transactions on Computers, 55, 34-47, 2006.

[114] R. L. Rivest, "Permutation polynomials modulo $2^w$", Finite Fields and Their Applications, 7, 287-292, 2001.

[115] R. L. Rivest, M. J. B. Robshaw, R. Sidney e Y. L. Yin, "The RC6 block cipher", disponível em `http://theory.lcs.mit.edu/~rivest/rc6.pdf`

[116] J. P. O. Santos, *Introdução à Teoria dos Números* , Coleção Matemática Universitária, IMPA, 2005.

[117] A. Schönhage, "Schnelle Multiplikation von Polynomen über Körpern der Charakteristik 2", Acta Informatica, 7, 395-398, 1977.

[118] A. Schönhage e V. Strassen, "Schnelle Multiplikation großer Zahlen", Computing, 7, 281-292, 1971.

[119] R. Sedgewick e P. Flajolet, *An Introduction to the Analysis of Algorithms*, Addison-Wesley, Reading MA, 1996.

[120] I. A. Semaev, "An algorithm for the evaluation of discrete logarithms in some nonprime finite fields", Mathematics of Computation, 224, 1679-1689, 1998.

[121] G. Seroussi, "Table of low-weight binary irreducible polynomials", HP Labs Technical Report HPL-98–135, 1998, 15 pages.

[122] J.-A. Serret, *Cours d'Algèbre Supérieure*, Gauthier-Villars, Paris, 3rd edition, 1866.

[123] V. Shoup, "Factoring polynomials over finite fields: asymptotic complexity vs. reality", Proceedings of IMACS Symposium, 124-129, 1993

[124] V. Shoup, "A new polynomial factorization algorithm and its implementation", Journal of Symbolic Computation, 20, 363-397, 1995.

[125] V. Shoup, Number Theory Library (NTL), Version 5.4, http://www.shoup.net, 2006.

[126] L. Stickelberger, "Über eine neue Eigenschaft der Diskriminanten algebraischer Zahlkörper", Verh. 1 Internat. Math. Kongresses, Zurich 1897, 182-193.

[127] D. Stinson, "Some observations on parallel algorithms for fast exponentiation in $\mathbb{F}_{2^n}$", SIAM Journal on Computing, 19, 711-717, 1990.

[128] V. Strassen, "Gaussian elimination is not optimal", Numer. Math., 13, 354-356, 1969.

[129] R. G. Swan, "Factorization of polynomials over finite fields", Pacific Journal of Mathematics, 12, 1099-1106, 1962.

[130] R. R. Varshamov, "Estimate of the number of signals in error correcting codes" (in Russian), Doklady Akademii Nauk, 117, 739-741, 1957.

[131] S. Vaudenay, "Cryptanalysis of the Chor-Rivest cryptosystem", Proceedings of CRYPTO 98, LNCS 1462, Springer-Verlag, 243-256, 1998.

[132] Z. Wan e K. Zhou, "On the complexity of the dual basis of a type I optimal normal basis", Finite Fields and Their Applications, 13, 411-417, 2007.

[133] A. E. Western e J. C. P. Miller, "Tables of indices and primitive roots", Royal Society Mathematical Tables, 9, Cambridge University Press, 1968.

[134] D. H. Wiedemann, "Solving sparse linear equations over finite fields", IEEE Transactions on Information Theory, 32, 54-62, 1986.

[135] H. Wu e B. Preneel, "Chosen IV attack on stream cipher WG", 2005; disponível em http://cr.yp.to/streamciphers/wg/045.pdf.

[136] B. Young e D. Panario, "Low complexity normal bases", Finite Fields and Their Applications, 10, 53-64, 2004.

[137] D. Y. Y. Yun, "On square-free decomposition algorithms", Proceedings of ISSAC'76, ACM Press, 26-35, 1976.

# Índice