UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA

Gustavo Souza Banegas

PENTANÔMIOS IRREDUTÍVEIS SOBRE $GF(2^M)$ PARA REDUÇÃO MODULAR EFICIENTE

Florianópolis

2015

Gustavo Souza Banegas

PENTANÔMIOS IRREDUTÍVEIS SOBRE $GF(2^M)$ PARA REDUÇÃO MODULAR EFICIENTE

Dissertação submetida ao Programa de Pós-Graduação em Ciências da Computação para a obtenção do Grau de Mestre.
Orientador
Universidade Federal de Santa Catarina: Prof. Dr. Ricardo Custodio
Coorientador
Carleton University: Prof. Dr. Daniel Panario

Florianópolis

2015

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Banegas, Gustavo Souza
Pentanômios irredutíveis sobre $GF(2^m)$ para redução modular eficiente / Gustavo Souza Banegas ; orientador, Ricardo Custodio ; coorientador, Daniel Panario. - Florianópolis, SC, 2015.
111 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico. Programa de Pós-Graduação em Ciência da Computação.

Inclui referências

1. Ciência da Computação. 2. Polinomios Irredutiveis. 3. Reducao Modular. 4. Aritmetica Modular. I. Custodio, Ricardo. II. Panario, Daniel. III. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Ciência da Computação. IV. Título.

Gustavo Souza Banegas

PENTANÔMIOS IRREDUTÍVEIS SOBRE $GF(2^M)$ PARA REDUÇÃO MODULAR EFICIENTE

Esta Dissertação foi julgada aprovada para a obtenção do Título de "Mestre", e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciências da Computação.

Florianópolis, 01 de outubro 2015.

Prof. Dr. Carina F. Dorneles
Coordenador
Universidade Federal de Santa Catarina

Banca Examinadora:

Prof. Dr. Ricardo Custodio
Orientador
Universidade Federal de Santa Catarina

Prof. Dr. Daniel Panario
Coorientador
Carleton University

Prof. Ricardo Dahab, Dr.
Universidade Estadual de Campinas

Prof. Lau Cheuk Lung, Dr. Eng.
Universidade Federal de Santa Catarina

Prof. Jean Matina, Dr.
Universidade Federal de Santa Catarina

Quarto membro
Universidade ...

Dedico este trabalho a todos que acreditaram em mim. Dedico este a todos que buscam conhecimento e a todos aqueles que são verdadeiros cientistas.

AGRADECIMENTOS

Primeiramente, gostaria de agradecer a minha mãe e meu irmão pelo suporte e dedicação durante a minha jornada para finalizar o mestrado. Gostaria de agradecer a minha namorada Carolina Haddad pelo carinho, dedicação, por tornar a minha vida mais bonita e por todas as revisões e comentários nos artigos e na dissertação.

Aos professores Ricardo Custódio e Daniel Panario por me apresentarem a criptografia e todo o mundo que a envolve, pelos conselhos acadêmicos e não acadêmicos. E no mundo da criptografia gostaria de agradecer ao Peter Schwabe pelos conselhos e pelos convites ao Chaos Communication Congress. E por fim, no mundo da criptografia, gostaria de agradecer a Tanja Lange e ao Daniel Bernstein pela oportunidade de doutorado na *Eindhoven University of Technology*.

Gostaria de agradecer a todas as pessoas que fizeram parte dessa jornada como a Lucila Alosilla, a professora Lucia Moura e todos os integrantes do LabSEC. Da BRy, empresa em que atualmente estou empregado, gostaria de agradecer em especial Cristian, Jeandré e Caio e a todos que fizeram parte neste último ano da minha jornada pelo título de mestre.

Por último, mas não menos importante, gostaria de agradecer a comissão de defesa desta dissertação de mestrado.

"I am the master of my fate: I am the captain of my soul." (William Ernest Henley,1888)

RESUMO

Este trabalho teve como objetivo propor uma análise de complexidade de pentanômios na aritmética modular polinômial em $GF(2^m)$. Para isto, foi realizado um estudo das técnicas existentes e implementado um algoritmo para determinar o número de operações base em *bits*.
O algoritmo teve uma heurística de algoritmos gulosos para otimizar estas operações. O resultado da computação do algoritmo para determinados pentanômios de grau de interesse foi a constituição de duas novas famílias de pentanômios irredutíveis. Com isso, é apresentada uma nova classe de pentanômio irredutível sobre $\mathbb{F}_2$ com o seguinte formato $f(x) = x^{2b+c} + x^{b+c} + x^b + x^c + 1$ onde $b > c$. Seja $m = 2b + c$ e o uso de f para definir a extensão de um corpo finito $\mathbb{F}_{2^m}$. É demonstrado que a complexidade da aritmética modular pode ser efetuada em $3m-2 = 6b+3c-2$ *XORs*. Entretanto, são apresentados casos particulares para quando $b = 2c$. Neste caso, o número de operações cai para $\frac{12}{5}m - 1$. Consequentemente, o número total de operações *XOR* para multiplicar $\mathbb{F}_{2^m}$ utilizando a família proposta é $m^2 + m - 1$; e quando $b = 2c$ o número total é $m^2 + \frac{2}{5}m$. O atraso das portas lógicas é tão bom quanto os pentanômios encontrados na literatura. A família proposta neste trabalho apresenta uma excelente performance na redução modular para alguns graus de m, incluindo os recomendados pelo NIST, isto é, para 163, 283 e 571.

Palavras-chave: Pentanômios irredutíveis. Aritmética de Corpos Finitos. Corpos Binários.

ABSTRACT

This study is aimed at proposing an analysis of pentanomials for modular reduction in $GF(2^m)$. To achieve this goal, an evaluation of the techniques for reduction was implemented that is capable to determine the number of ground operations in bits. The algorithm uses a greedy heuristic to optimize these operations. The result of the computation of the algorithm for some polynomials was the basis for the detection of two new families of irreducible pentanomials. We introduce a new class of irreducible pentanomials over $\mathbb{F}_2$ of the form $f(x) = x^{2b+c} + x^{b+c} + x^b + x^c + 1$ where $b > c$. Let $m = 2b + c$ and use f to define the finite field extension $\mathbb{F}_{2^m}$. We show that the bit complexity of reducing modulo f is, in general, $3m - 2 = 6b + 3c - 2$ XORs. In the particular case when $b = 2c$, we further reduce these number of operations to $\frac{12}{5}m - 1$. Consequently, the total number of XOR operations to multiply in $\mathbb{F}_{2^m}$ using our pentanomials is $m^2 + m - 1$; when $b = 2c$ this number is $m^2 + \frac{2}{5}m$. Our gate delay is as good as the best pentanomials found in the literature. Hence, our new class of pentanomials has excellent performance, and it is the best possible for some degree extensions m including the NIST degrees 163, 283 and 571.

Keywords: Irreducible Pentanomials. Finite Fields Arithmetic. Binary Fields.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

GF	*Galois Field*
XOR	Operação Lógica de *Ou Exclusivo*
AND	Operação Lógica *e*
nr	Número de Passos de Redução
NIST	*National Institute of Standards and Technology*
MR	Matriz de Redução
MER	Matriz de Elementos Repetidos

LISTA DE SÍMBOLOS

$\oplus$	Operação *XOR*
$N_{\oplus}$	Número de Operações *XOR*
T_X	Atraso de uma porta *XOR* com apenas 2 entradas

SUMÁRIO

1 INTRODUÇÃO

Extensões de corpos finitos desempenham um importante papel em diferentes aplicações na computação e engenharias, como por exemplo:

- álgebra (Extensões de Corpos e *Galois Theory*) (LIDL; NIEDERREITER, 1986; MULLEN; PANARIO, 2013)
- Matemática discreta e combinatória (COLBOURN; DINITZ, 2006)
- Teoria de números (LIDL; NIEDERREITER, 1986)
- Criptografia (GOLOMB; GONG, 2004; MULLEN; PANARIO, 2013)
- Geradores de números pseudoaleátorios (*Linear Feedback Shift Register*) (GOLOMB, 1981; LIDL; NIEDERREITER, 1986)

Os elementos em extensões de corpos finitos binários, também denotados por $GF(2^m)$ ou $\mathbb{F}_{2^m}$, podem ser representados de várias formas, sendo as mais comuns as bases polinomiais e as bases normais. Neste trabalho, o interesse maior é na base polinomial e suas operações aritméticas.

As operações aritméticas em bases polinomiais são amplamente estudadas. Neste âmbito, muitos autores apresentam a construção de multiplicadores e outras funções sobre $GF(2^m)$, tais como Fan e Hasan (2015), Xiong e Fan (2014), Reyhani-Masoleh e Hasan (2004), Rodríguez-Henríquez e Koç (2003) e Wu (2002).

Entre as operações em $\mathbb{F}_{2^m}$, duas operações se destacam: a multiplicação e a redução modular. Ambas possuem um custo computacional não trivial para serem executadas. Entretanto, estas operações são essenciais para diversas aplicações, assim como, por exemplo, em criptografia (HANKERSON; MENEZES; VANSTONE, 2003; COHEN et al., 2012).

A multiplicação em $GF(2^m)$ consiste em multiplicar dois polinômios $a, b \in GF(2^m)$, onde a e b têm grau máximo $m-1$, o que resulta em um terceiro polinômio $c = ab$, que pode possuir grau até $2m-1$. Desta forma, caso o grau de c seja maior que m, é necessário reduzi-lo para que ele volte a ter grau menor que m. Para isso, é necessário efetuar a redução modular, ou seja, efetuar a divisão de c por f, onde f é um polinômio irredutível de grau m, e tomar o resto. É importante destacar que um corpo finito é definido se e somente se

o polinômio f é irredutível. Sabe-se que, para cada $m \geq 1$, existe pelo menos um polinômio irredutível de grau m (LIDL; NIEDERREITER, 1986).

Matematicamente, pode-se escolher qualquer polinômio irredutível f para a geração do corpo. Entretanto, nas implementações de hardware, a escolha do polinômio é crucial para uma boa performance, conforme a literatura apresenta (FAN; HASAN, 2015).

Para as operações consideradas nesta dissertação, uma forma de mensurar a performance nas implementações é por meio da verificação da quantidade de portas lógicas de *ou exclusivo* e de portas lógicas *e*. Neste trabalho, estas serão designadas pelas siglas em inglês: *XOR* e *AND*, respectivamente. Além disso, deve-se considerar o atraso dessas portas, quando estas são utilizadas.

1.1 MOTIVAÇÃO

Dada a complexidade que envolve a multiplicação e a redução, comentada anteriormente, e devido á necessidade de multiplicação em corpos finitos, encontra-se o seguinte problema em aberto:

- Encontrar o melhor polinômio irredutível para efetuar a operação modular em corpos finitos da forma $GF(2^m)$.

Para efetuar a aritmética modular, isto é, a redução modular, são normalmente utilizados polinômios irredutíveis com poucos elementos não nulos. Como, por exemplo, trinômios irredutíveis, que apresentam três coeficientes não nulos. O trabalho de (WU, 2002) apresenta uma forma de efetuar a redução modular de forma eficiente utilizando estes trinômios.

Infelizmente, não existem trinômios para todo m e, para estes casos, normalmente é utilizado um pentanômio irredutível. Um pentanômio é um polinômio com cinco coeficientes não nulos. No caso de pentanômios, alguns autores apresentam as seguintes famílias irredutíveis:

- $x^m + x^a + x^b + x^c + 1$, onde $1 \leq c < b < a \leq m/2$ (FAN; HASAN, 2015);
- $x^m + x^{b+1} + x^b + x + 1$, onde $1 < b < m - 1$ (RODRÍGUEZ-HENRÍQUEZ; KOÇ, 2003; REYHANI-MASOLEH; HASAN, 2004);
- $x^m + x^{b+1} + x^b + x^{b-1} + 1$, onde $1 \leq b \leq \frac{m}{2} - 1$ (RODRÍGUEZ-HENRÍQUEZ; KOÇ, 2003; REYHANI-MASOLEH; HASAN, 2004);

- $x^m + x^{m-c} + x^b + x^c + 1$, onde $1 \leq c < b < m - c$ (CILARDO, 2013);
- $x^m + x^{m-s} + x^{m-2s} + x^{m-3s} + 1$, onde $(m-1)/8 \leq s \leq (m-1)/3$ (REYHANI-MASOLEH; HASAN, 2004);
- $x^{4c} + x^{3c} + x^{2c} + x^c + 1$, onde $c = 5^i$ e $i \geq 0$ (HASAN; WANG; BHARGAVA, 1992).

Estas famílias apresentam boas propriedades em relação a multiplicação e redução modular polinomial em corpos finitos. Isto é, a redução modular usando polinômios irredutíveis dessas famílias requerem um número de operações *XOR* reduzido. Isso implica que, quando implementados em hardware, este seja o melhor possível.

Nesse sentido, esta pesquisa destina-se a encontrar uma família de pentanômios que utilize uma quantidade menor de operações *XORs* para realizar a redução modular para diferentes valores de m que aquelas das famílias acima citadas. Além disto, adicionalmente, outras contribuições deste trabalho podem ser citadas: Primeiramente, têm-se o estudo detalhado das operações de módulo de polinômios sobre $GF(2^m)$. Em segundo lugar, há o desenvolvimento de um algoritmo para analisar o número de operações *XORs* do processo de redução de um polinômio dado o polinômio irredutível.

Para a realização deste trabalho, foi realizada uma pesquisa teórica sobre os métodos existentes em corpos finitos, projetos e análises de algoritmos. A análise dos dados obtidos desta revisão da literatura permitiu o desenvolvimento do algoritmo apresentado neste trabalho e possibilitou a descoberta de mais de uma família de pentanômios irredutíveis com uma boa performance, no que tange ao número de operações *XORs* necessárias.

1.2 OBJETIVOS

O objetivo geral deste trabalho consiste em melhorar a aritmética de redução modular sobre $\mathbb{F}_{2^m}$.

1.2.1 Objetivos específicos

Os objetivos específicos para realização desta pesquisa são:

- estudar as famílias existentes disponíveis na literatura;

- estudar os metódos de redução polinômial existentes;
- avaliar a utilização dos pentanômios existentes;
- propor um algoritmo para a contagem de operações *XORs* da redução modular polinômial em $GF(2^m)$;
- procurar famílias eficientes para redução modular;
- provar que essas famílias são de fato eficientes.

1.3 LIMITAÇÕES

Este trabalho tem o foco no desenvolvimento do algoritmo para contar as operações *XOR* e na definição de uma família de pentanômios irredutíveis eficiente. Isto é, será definida uma nova família que diminui a quantidade de operações *XORs* da aritmética modular polinomial, dadas as famílias de polinômios presentes na literatura. O trabalho não irá apresentar nenhuma relação com portas lógicas *AND*, uma vez que estas são apenas utilizadas na multiplicação de polinômios antes da redução pelo polinômio irredutível, e o foco dessa dissertação é a redução modular. Entretanto, para uma comparação com as referências, será utilizado o valor de $(m-1)^2$ operações *XORs* devido ao multiplicador.

Os resultados da execução do algoritmo desta dissertação indicam que existem outras possibilidades de famílias de pentanômios eficientes. Esses resultados serão propostos como trabalho futuro. Eles poderão vir a ter uma uma performance melhor do que a família detalhada neste trabalho.

1.4 METODOLOGIA

Para a realização deste trabalho, buscou-se utilizar uma combinação de procedimentos metodológicos, conforme apresentado no trabalho de Vieira (2012). O uso de mais de uma técnica de pesquisa previne que o estudo seja limitado, aumentando a "flexibilidade na correspondência do fenômeno sob investigação" e a robustez dos resultados (KAPLAN; DUCHON, 1988; WAZLAWICK, 2009).

Tendo isso em vista, este trabalho foi estruturado em três etapas: revisão de literatura; modelagem teórica; e resultados finais. A Figura

1 apresenta um esquema da operacionalização da pesquisa, discutida nos tópicos seguintes.

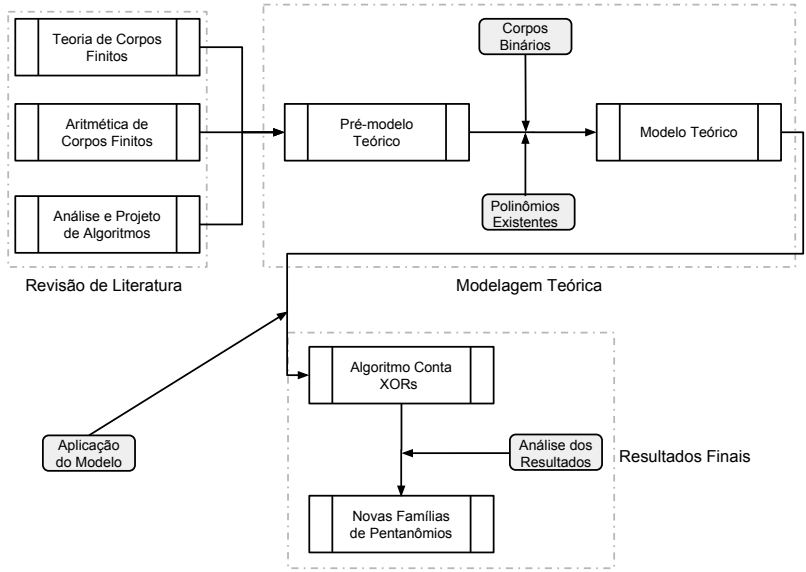

Figura 1 – Esquema de realização do trabalho.

1.4.1 Revisão da literatura

Para a construção da revisão da literatura foram consultadas bases de dados e os orientadores. Esta consulta foi realizada com três objetivos principais: identificar a teoria de corpos finitos disponíveis na literatura, identificar os polinômios utilizados na redução polinomial disponíveis na literatura e identificar a forma como é analisado o custo de implementação destes polinômios.

Para o primeiro caso, foram identificadas três bases de dados: Scopus, IEEEXplore e Springer Link. A escolha das bases se deve o fato de que estas englobam os periódicos que mais publicam sobre Ciência da Computação e Álgebra Computacional, bem como sobre o tema de análise de complexidade de operações sobre $GF(2^m)$. O apoio dos orientadores foi necessário para estabelecer um guia dos principais artigos e autores sobre o tema de pesquisa.

Para os outros temas que figuram no referencial teórico deste trabalho, como a definição de corpo finito e polinômio irredutível, que são definições teóricas, não foi necessário um estudo detalhado. Assim, buscou-se apenas verificar os principais autores e conceitos destes temas em livros, periódicos das áreas e conversas com os orientadores do trabalho.

1.4.2 Modelagem teórica

Primeiramente, foram estudadas a redução polinomial em $GF(2^m)$ e as famílias de polinômios irredutíveis existentes. Em seguida, foi efetuada uma análise para estabelecer as etapas do algoritmo que conta o número de operações *XOR*. Ao final, estabeleceu-se um modelo teórico da família de pentanômios irredutíveis de performance melhor que as encontradas na literatura.

1.4.3 Modelagem do algoritmo de redução

Tendo em mãos o resultado da pesquisa bibliográfica, realizou-se a leitura dos artigos selecionados em busca de implementações e descrições de operações aritméticas em corpos binários, especialmente a redução modular. Baseando-se nas soluções existentes e experimentações práticas, estabeleceu-se uma forma de construção de matrizes de redução. Estas matrizes são fundamentais ao trabalho, uma vez que fornecem todos os dados para calcular o número de operações *XORS* necessárias. Além disso, a otimização de operações é estabelecida devido a modelagem dessas matrizes de redução.

1.4.4 Resultado final

Por fim, fez-se a avaliação dos resultados do algoritmo de contagem de *XORs*. Tomaram-se os resultados do algoritmo e suas avaliações para verificar qual apresentava uma quantidade menor de operações e o seu padrão na construção do pentanômio. Desta forma, encontraram-se as seguintes famílias de pentanômios:

$$x^{2b+c} + x^{b+c} + x^b + x^c + 1 \text{ onde } b > c,$$

e

$$x^{b+2c} + x^{b+c} + x^b + x^c + 1 \text{ onde } b > c.$$

Após a identificação destas famílias, deu-se então o ínicio do estabelecimento de um modelo matemático de redução modular polinômial para $GF(2^m)$. Foi então analisada a família $x^{2b+c} + x^{b+c} + x^b + x^c + 1$ e desta análise foi estabelecido dois resultados importantes: o número exato de *XORS* dado m e um algoritmo para a redução que utiliza um pentanômio irredutível do formato $x^{2b+c} + x^{b+c} + x^b + x^c + 1$. Com estes resultados, foi efetuada a comparação desta família com as demais famílias existentes na literatura. Desta forma, concluiu-se que a família faz uso de menos operações que as existentes na literatura para efetuar a redução modular.

1.5 ESTRUTURA DA DISSERTAÇÃO

Com relação á estrutura, este trabalho está organizado conforme descrito a seguir. O Capítulo 2 apresenta uma revisão da literatura sobre a definição de corpo finito e traz outros conceitos necessários para o entendimento deste trabalho. O Capítulo 3 contém a explicação sobre aritmética modular em corpos finitos. Esses conceitos embasaram a construção do algoritmo para a contagem do número de operações *XOR*, denominado Conta-XOR.

O Capítulo 4 detalha o algoritmo Conta-XOR. Para o entendimento do funcionamento deste algoritmo, é apresentado um exemplo numérico com o passo a passo da execução do mesmo. Em seguida, é efetuada uma análise de complexidade do algoritmo, onde são considerados os números de operações e as alocações de espaço utilizados neste. Para comparar os resultados do algoritmo proposto com os encontrados na literatura, calculou-se o número de *XORs* nas famílias existentes com os resultados gerados pelo Conta-XOR. Por fim, é apresentado os resultados da execução do Conta-XOR para os graus que utilizam pentanômios recomendados pelo *National Institute of Standards and Technology* (NIST, 2012), e as duas famílias de pentanômios que apresentaram o menor número de *XORs*. Uma das famílias achadas através do Conta-XOR é detalhada no Capítulo 5. A outra família encontrada é proposta como um trabalho futuro, devido a complexidade requerida para estabelecer o modelo teórico.

O detalhamento da família se inicia no estabelecimento do número de reduções, apresentado no Capítulo 5. Em seguida, são estabelecidas as equações para redução modular para o caso em que o número de

passos de redução é 2. Para este caso, é apresentado o algoritmo para redução modular e um exemplo numérico. Além disso, são demonstradas as equações para a redução e o algoritmo de redução modular para o caso em que o número de passos de redução é 3. Ao final do capítulo, é apresentada uma comparação do número de operações *XORs* utilizadas pelas famílias conhecidas na literatura e pela família descoberta neste trabalho.

Por fim, o Capítulo 6 apresenta as considerações finais deste trabalho e indica as oportunidades para pesquisas futuras.

2 REFERENCIAL TEÓRICO

Para um melhor entendimento deste trabalho, faz-se necessário explicar pontos da teoria de corpos finitos e suas propriedades.

Neste capítulo, são apresentados brevemente definições, teoremas e conceitos sobre corpos finitos e suas operações aritméticas. Primeiramente, será definido o que é um corpo finito, corpo binário e quais são as operações. Em seguida, são apresentados polinômios em corpos finitos, e por fim são apresentados trinômios e pentanômios irredutíveis.

Com o intuito de clarificar o texto, todas as definições, teoremas e exemplos neste capítulo são encontradas nos trabalhos de Deschamps, Imana e Sutter (2009), Masuda e Panario (2007) e Lidl e Niederreiter (1986).

2.1 CORPOS

Os corpos, originalmente, foram concebidos como abstrações de conjuntos numéricos ($\mathbb{Q}, \mathbb{R}, \mathbb{C}$) e suas propriedades essenciais. Os corpos consistem de um conjunto $\mathbb{F}$ com duas operações sobre ele, que satisfazem o seguinte:

a) $(\mathbb{F}, +)$ é um grupo abeliano aditivo com identidade denotada por 0;

b) $(\mathbb{F} \setminus \{0\}, \cdot)$ é um grupo abeliano multiplicativo com identidade denotada por 1 ; e

c) $a \cdot (b + c) = a \cdot b + a \cdot c$, $(b + c) \cdot a = b \cdot a + c \cdot a$ para quaisquer $a, b, c \in \mathbb{F}$.

Se o conjunto $\mathbb{F}$ é finito, então $\mathbb{F}$ é um corpo finito.

Definição 1 *A característica de um corpo* $\mathbb{F}$ *é o menor número inteiro positivo* m *tal que* $ma = \underbrace{a + a + \ldots + a}_{m \text{ vezes}} = 0$*, sendo* $a \in \mathbb{F}$*. Se tal* m *não existir, a característica do corpo é definida como zero.*

A seguir irá apresentar a definição de corpos de característica 2, e a aritmética que a envolve.

2.1.1 Corpos finitos de característica 2

Um corpo finito de característica 2, usualmente denotado por $\mathbb{F}_{2^m}$ ou $GF(2^m)$, também são conhecidos como corpos binários.

Os elementos do corpo finito $GF(2^m)$ são polinômios, onde α é a raiz do polinômio irredutível f sobre $GF(2)$, e os coeficientes do polinômio são $\{0, 1\}$.

Este corpo é de grande interesse na computação, uma vez que pode ser representado como um vetor de *bits*, e sua a soma e produto correspondem a *XOR* e *AND*. Outro ponto de interesse são as operações aritméticas no corpo, que são detalhados a seguir.

2.1.1.1 Aritmética em $\mathbb{F}_{2^m}$

Todas as operações neste corpo são efetuadas módulo f, onde f é um polinômio irredutível de grau m. A seguir, tem-se um pequeno resumo das operações, que são:

- *Adição*: são somas de polinômios com coeficientes resultantes reduzidos módulo 2. A adição representa uma operação *XOR*.

- *Multiplicação*: Pode ser efetuada através de *shifts* e operações de *XOR* reduzindo os coeficientes módulo 2.

As demais operações que envolvem corpos binários podem ser encontradas na literatura especificada no inicio deste trabalho. Para determinar as complexidades destas operações, elas podem ser mensuradas de acordo com as definições dadas a seguir.

2.1.1.2 Complexidade em $\mathbb{F}_{2^m}$

A implementação de multiplicadores e demais operações aritméticas em hardware de corpos binários são tipicamente avaliados pelas complexidades de tempo e espaço. Como são utilizadas portas *XORs* (e *AND*) de duas entradas para realizar as operações, uma medida muito utilizada para a complexidade espacial é o número total de portas. O atraso do sinal, na combinação destas portas, é a complexidade temporal. Os símbolos T_A e T_X são utilizados para representar o atraso em portas *AND* e portas *XOR*, respectivamente (FAN; HASAN, 2015).

2.2 POLINÔMIOS E CORPOS FINITOS

Esta seção estrutura a relação de corpos finitos e polinômios e apresenta as propriedades matemáticas que serão necessárias para o entendimento dos próximos capítulos deste trabalho.

2.2.1 Divisão de polinômios

O Teorema 2 demonstra que é possível dividir polinômios em corpos finitos, ou seja, existe a operação de divisão em corpos finitos, quando é utilizado polinômios. A seguir, apresenta-se um algoritmo para efetuar a divisão.

Teorema 2 *Sejam F um corpo e $f, g \in F[x]$ com $g \neq 0$. Existem polinômios únicos $h, r \in F[x]$, tais que $f = gh + r$ e $grau(r) < grau(g)$.*

O algoritmo de divisão de polinômios muito utilizado é, em inglês, chamado de *polynomial long division*. O Algoritmo 1 representa o algoritmo *polynomial long division*. A função $deg()$ representa o grau do polinômio e a função $lead()$ é o termo de maior grau do polinômio (BARNARD, 2008).

Algoritmo 1: Algoritmo de divisão *polynomial long division*.

Entrada: a e b, onde $b \neq 0$ e a, b são dois polinômios
Saída: quociente q e resto r

```
enquanto r ≠ 0 e deg(r) ≥ deg(b) faça
    t ← lead(r)/lead(b);
    (q, r) ← (q + t, r − (t ∗ b));
retorna (q, r);
```

Definição 3 *Seja F um corpo. Dados $f, g \in F[x]$, existe um único polinômio mônico $d \in F[x]$, tal que:*

a) d divide f e g;

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

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

2.2.2 Congruência de polinômios

Definição 4 *Dado três polinômios g, h e f em $\mathbb{F}[x]$, g é congruente a h módulo f se f divide $g - h$. A notação é dada por:*

$$g \equiv h \pmod{f}$$

A congruência apresenta as seguintes propriedades:

1. $g \equiv h \pmod{f}$ se e somente se os restos da divisão de g por f e h por f são iguais.

2. Congruência modulo f é uma relação de equivalência, isto é, é reflexiva, simétrica e transitiva.

3. Se $g_1 \equiv h_1 \pmod{f}$ e $g_2 \equiv h_2 \pmod{f}$, então $g_1 + h_1 \equiv g_2 + h_2 \pmod{f}$, $g_1 - h_1 \equiv g_2 - h_2 \pmod{f}$ e $g_1 h_1 \equiv g_2 h_2 \pmod{f}$.

2.3 POLINÔMIOS IRREDUTÍVEIS

Definição 5 *Seja $f \in F[x]$ um polinômio de grau pelo menos 1. Então, f é irredutível sobre F se ele não pode ser escrito como o produto de outros dois polinômios em $F[x]$.*

Teorema 6 *Seja $\mathbb{F}$ um corpo e f um polinômio mônico com coeficientes sobre $\mathbb{F}$ de grau positivo. Então $\mathbb{F}[x]/(f)$ é um corpo se e somente se f é irredutível sobre $\mathbb{F}$.*

Teorema 7 *Seja $q = p^n$, onde p é primo. Se f um polinômio irredutível sobre $\mathbb{F}_p$ de grau n então $\mathbb{F}_q[x] \equiv \mathbb{F}_p[x]/(f)$.*

Em geral, para $q = p^n$, $\mathbb{F}_{q^m} \equiv \mathbb{F}_q[x]/(f)$, onde f é um polinômio irredutível de grau m sobre $\mathbb{F}_q$. Assim, um elemento em $\mathbb{F}_{q^m}$ pode ser representado por um polinômio em $\mathbb{F}_q[x]$ de grau menor que m.

Exemplo 8 *Como $f(x) = x^2 + x + 1$ tem grau 2 e não possui raízes em $\mathbb{F}_2$, f é irredutível sobre $\mathbb{F}_2$. Pelos Teoremas 6 e 7, tem-se que $\mathbb{F}_4[x]$ e $\mathbb{F}_2[x]/(f)$ são isomorfos. Os elementos de $\mathbb{F}_4[x]$, representados em termos de polinômios, são $0, 1, x, x + 1$.*

2.3.1 Trinômios e pentanômios irredutíveis

Seja $f(x) = x^m + x^a + 1$, com $a > 0$ um trinômio irredutível usado para gerar $GF(2^m)$. O trinômio f tem apenas 3 coeficientes diferentes de zero e nenhum outro polinômio irredutível tem um número menor de termos. Os trinômios se destacam porque são polinômios com baixa complexidade na aritmética de corpos finitos. Entretanto, sabe-se que trinômios irredutíveis onde m é múltiplo de 8 não existem (SWAN, 1962).

Para os graus que não apresentam nenhum trinômio irredutível, geralmente existe um pentanômio irredutível. Um pentanômio é um polinômio com 5 coeficientes diferentes de zero, isto é, $f(x) = x^m + x^a + x^b + x^c + 1$, onde $1 \leq c < b < a < m$. Os pentanômios muitas vezes continuam apresentando boas propriedades na aritmética de $GF(2^m)$, como será apresentado nessa dissertação.

Trinômios e pentanômios são muito utilizados em aplicações de corpos finitos, especialmente em criptografia (COHEN et al., 2012). Além do mais, foi provado que existem trinômios ou pentanômios para todo m no intervalo $[2, 120.000]$, ver em (MULLEN; PANARIO, 2013).

3 REDUÇÃO MODULAR POLINOMIAL

A multiplicação de polinômios que representam elementos em um corpo finito pode produzir um polinômio de grau maior ou igual a m. Se este for o caso, é necessário tomar o módulo deste polinômio pelo polinômio irredutível f usado para definir o corpo. Denomina-se esta operação de redução modular. Como este trabalho se dedica a bases polinomiais, para esta base é utilizada a aritmética modular polinomial.

Sabe-se que eficiência para realizar essa redução modular depende do polinômio irredutível f. Assim, é importante escolher f de tal forma que o processo de redução seja o menos custoso possível.

Neste capítulo será demonstrado o processo genérico da redução modular polinomial. O intuito é explicar o processo e introduzir a noção da utilização de polinômios irredutíveis, em especial, trinômios e pentanômios. Também será apresentado o conceito de número de passos de redução e a forma para o seu cálculo.

A organização do capítulo segue o seguinte formato: na Seção 3.1 existe a explicação do algoritmo de redução modular polinomial. Em seguida, na Seção 3.2, será apresentado o cálculo para determinar o número de passos de redução, que é uma extensão do trabalho de Sunar e Koç (1999). Por fim, na Seção 3.3, será ilustrado a redução modular polinomial utilizando trinômios.

3.1 ALGORITMO DE ARITMÉTICA MODULAR

A redução modular polinomial utilizando um polinômio irredutível do formato $x^m + x^{m-1} + \cdots + x + 1$ é ilustrado na Figura 2. Por convenção, a parte mais baixa do elemento, isto é, de 0 a $m-1$, fica á direta, e a parte mais alta á esquerda.

Percebe-se que a redução modular é efetuada pegando o conjunto de elementos entre m a $2m-2$ e substituindo-os por seus equivalentes definitos por f. Em seguida, é efetuado o *XOR* dos elementos abaixo de $m-1$ até 0. Todavia, percebe-se que existe uma parte que ainda continua acima de $m-1$. Na Figura 2, a sobressaliência é representada pela parte escura. Então, faz-se necessário repetir o processo com cada parte que passa de $m-1$.

O Algoritmo 2 apresenta, em alto nível, a redução modular polinomial. No algoritmo, tem-se a atribuição de uma variável temporária da parte que é necessário reduzir, isto é, para elementos com grau m

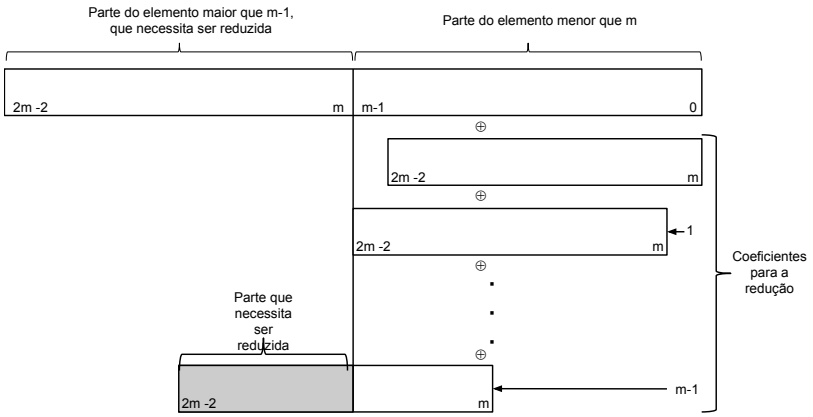

Figura 2 – Aritmética modular polinomial genérica.

ou maior, no algoritmo esta variável é representada por v_2. A seguir, no *loop* da linha 3 a 5, tem-se uma função acumuladora, onde serão definidos os elementos que irão ser utilizados na redução, deslocando o tamanho do expoente do polinômio irredutível. O símbolo $\ll$ significa deslocamento de elementos à esquerda.

Algoritmo 2: Algoritmo de alto nível para efetuar a aritmética modular polinomial.

Entrada: V vetor de bits, m grau do polinômio irredutível, $expoentes[]$ expoentes do polinômio irredutível f
Saída: O polinômio reduzido V como um vetor de bits de tamanho $m-1$

1 $v_1 \leftarrow V[0:(m-1)]$;
2 $v_r \leftarrow 0$;
3 **para** $i \leftarrow 0$ **até** $tamanho(expoentes)$ **faça**
4 $\quad v_2 \leftarrow V[m:(2*d-2)]$;
5 $\quad v_r \leftarrow v_r \oplus (v_2 \ll expoentes[i])$;
6 $V \leftarrow v_1 \oplus v_r$;
7 **retorna** V;

O algoritmo descrito anteriormente não está totalmente correto. Pode haver a necessidade de mais uma redução. Por isso, a Figura 3 ilustra a aritmética modular polinomial com diversos passos de redução.

Estes passos são possíveis de determinar dado um polinômio irredutível, como será visto na seção a seguir.

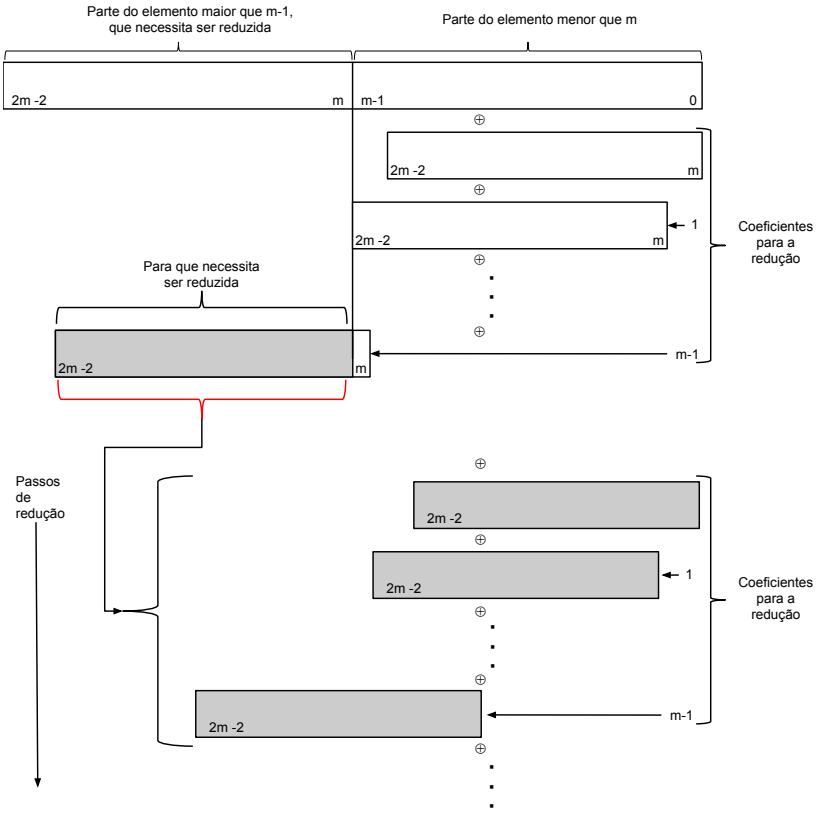

Figura 3 – Aritmética modular polinomial genérica com representação ilustrativa do número de passos de redução.

3.2 PASSOS DE REDUÇÃO

Para determinar o número de passos de redução necessários, representado neste trabalho de nr, Sunar e Koç (1999) desenvolveram a Equação 3.1, onde m é o grau do polinômio irredutível f e a o segundo maior expoente de f. Nesta dissertação, foi efetuado um estudo das equações de Sunar e Koç (1999), e descobriu que esses passos traba-

lham dentro de intervalos. A seguir, será demonstrado a dedução destes intervalos.

Seja k o número de passos requeridos para reduzir um elemento em $GF(2^m)$. Este inteiro k satisfaz $2m - k(m-a) - 2 < m$, no qual implica em $k > \frac{m-2}{m-a}$. Logo, tem-se

$$k = \left\lfloor \frac{m-2}{m-a} \right\rfloor + 1. \tag{3.1}$$

Seja

$$L = \left\lfloor \frac{m-2}{m-a} \right\rfloor \tag{3.2}$$

então $k = L + 1$ e, da Equação 3.1, é possível obter

$$L \leq \frac{m-2}{m-a} < L + 1. \tag{3.3}$$

É possível isolar a da Equação 3.3:

$$\frac{L-1}{L}m + \frac{2}{L} \leq a < \frac{L}{L+1}m + \frac{2}{L+1}. \tag{3.4}$$

A Equação 3.4 apresenta o exato alcance de a em função de m e L. Para $k = 1$ ou $L = 0$, isto implica em apenas 1 iteração. Utilizando então a Eq. (3.2), tem-se:

$$0 = \left\lfloor \frac{m-2}{m-a} \right\rfloor,$$

isso é apenas possível se $m - a > m - 2$. Então,

$$a < 2.$$

Portanto, para $k = 1$, tem-se $a = 1$. Só existe um polinômio para este caso: $x^m + x + 1$.

Outro exemplo a ser demonstrado é quando o $k = 2$ ou $L = 1$. Então,

$$2 \leq a \leq \frac{1}{2}m + 1. \tag{3.5}$$

A Tabela 1 apresenta o número de passos de redução necessários, baseando-se nas variações de a entre 1 e $m - 1$.

Tabela 1 – Número de passos de redução necessários em função de a.

k	L	Variação de a
1	0	$a = 1$
2	1	$2 \leq a \leq \frac{1}{2}m + 1$
3	2	$\frac{1}{2}m + 1 < a \leq \frac{2m+2}{3}$
$\vdots$	$\vdots$	$\vdots$
$L + 1$	L	$\frac{(L-1)m+2}{L} < a \leq \frac{Lm+2}{L+1}$
$\vdots$	$\vdots$	$\vdots$
$m - 1$	$m - 2$	$a = m - 1$

3.3 REDUÇÃO UTILIZANDO TRINÔMIOS

Os trinômios apresentam o formato $x^m + x^a + 1$. O expoente a sempre determina o número de passos de redução, não importa se é trinômio ou outro polinômio. Conforme será demonstrado a seguir, o número de operações XOR para trinômios é $2m - 2$.

3.3.1 Caso para $x^m + x + 1$

A Figura 4 representa graficamente o que acontece quando é utilizado um trinômio no formato $f(x) = x^m + x + 1$. Na figura, o elemento $C0$ representa um polinômio de tamanho $x^{2m-2} + \cdots + x^0$. A figura se destaca por representar graficamente a congruência de polinômios, onde T_a representa a congruência dos elementos de x até x^{m-1} e T_b presenta a congruência dos elementos de 1 até x^{m-2}. Assim, tem-se:

$$x^m \equiv x + 1 \bmod f,$$

o grau máximo que um elemento pode resultar após uma multiplicação é $2m - 2$. Desta forma, tem-se:

$$
\begin{aligned}
x^m &\equiv x + 1 \bmod f, \\
x^{m+1} &\equiv x^2 + x \bmod f, \\
x^{m+2} &\equiv x^3 + x^2 \bmod f, \\
x^{m+3} &\equiv x^4 + x^3 \bmod f, \\
&\vdots \equiv \vdots \\
x^{2m-2} &\equiv x^{m-1} + x^{m-2} \bmod f.
\end{aligned}
$$

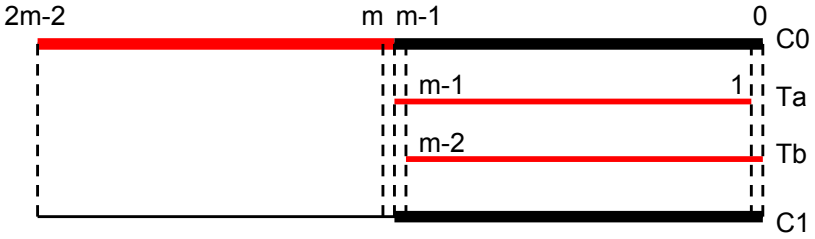

Figura 4 – Redução utilizando $f(x) = x^m + x + 1$.

A Figura 4 pode ser resumida na seguinte equação:

$$C1 = \sum_{i=0}^{2m-2} x^i a_i \pmod{x^m + x + 1}. \tag{3.6}$$

Logo, $C1$ é o resultado da redução modular. Desta forma, a Equação 3.6 pode ser transformada no seguinte:

$$
\begin{aligned}
&a_{2m-2}x^{m-1} + \sum_{1}^{m-2} a_{m+i-1}x^i + \\
&\sum_{1}^{m-2} a_{m+i}x^i + a_m + a_{m-1}x^{m-1} + \sum_{1}^{m-2} a_i x^i + a_0.
\end{aligned} \tag{3.7}
$$

Reescrevendo a equação, tem-se:

$$(a_{2m-2} + a_{m-1})x^{m-1} + \sum_{i=1}^{m-2} (a_{m+i} + a_{m+i-1} + a_i)x^i + (a_m + a_0). \tag{3.8}$$

A Equação 3.8 é a equação da redução, dado qualquer elemento em $\mathbb{F}_{2^m}$, definida a extensão usando um trinômio no formato de f.

3.3.2 Algoritmo para $f(x) = x^m + x + 1$

O Algoritmo 3 implementa a Equação 3.8. No algoritmo tem-se m como o grau do polinômio e $C_0[2m-2, 0]$ como um vetor de bits de tamanho $2m-1$.

Algoritmo 3: Algoritmo de redução para $x^m + x + 1$.

Entrada: m grau do trinômio irredutível, vetor de bits $C_0[2m-2, 0]$ de tamanho $2m-1$
Saída: vetor de bits $C_1[m-1, 0]$ de tamanho $m-1$

```
1 C1[m − 1] = C0[m − 1] ⊕ C0[2m − 2];
2 para i ← 1 até m − 2 faça
3     C1[i] = C0[i] ⊕ C0[m + i − 1] ⊕ C0[m + i];
4 C1[0] = C0[0] ⊕ C0[m];
5 retorna C1;
```

Com base na Equação 3.8 e no Algoritmo 3, é possível verificar que o número de operações XOR é $2m-2$.

3.3.3 Caso para $x^m + x^a + 1$ com $2 \leq a \leq \lfloor \frac{m+1}{2} \rfloor$

Para esses valores de a, o número de passos de redução é 2. A Figura 5 ilustra o processo de redução para este caso. Utilizando a congruência de polinômios, conforme demonstrado anteriormente, tem-se:

$$
\begin{aligned}
x^m &\equiv x^a + 1 \bmod f, \\
x^{m+1} &\equiv x^{a+2} + x \bmod f, \\
x^{m+2} &\equiv x^{a+3} + x^2 \bmod f, \\
x^{m+3} &\equiv x^{a+4} + x^3 \bmod f, \\
&\vdots \equiv \vdots \\
x^{2m-2} &\equiv x^{a+m-1} + x^{m-2} \bmod f.
\end{aligned}
$$

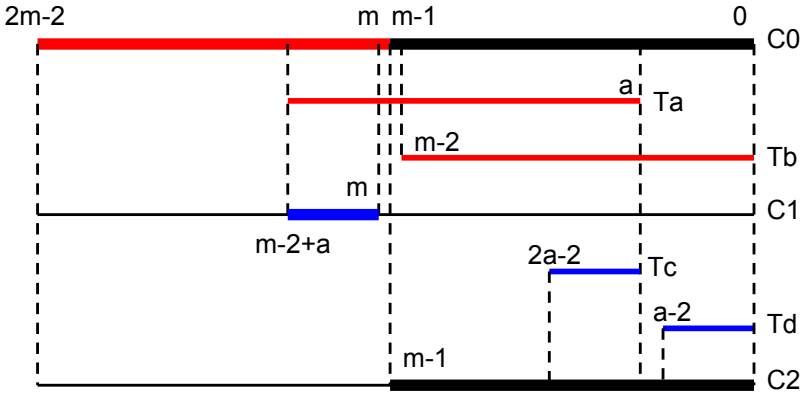

Figura 5 – Redução utilizando $x^m + x^a + 1$ com $2 \leq a \leq \lfloor \frac{m+1}{2} \rfloor$.

Da mesma forma que foi feito anteriormente, contudo, agora com mais um passo de redução, pode-se dizer que a computação da redução para este formato de polinômio será:

$$
\begin{aligned}
C_2[m-1, m-1] &\equiv C_0[m-1, m-1] \oplus C_0[2m-a-1, 2m-a-1] \\
C_2[m-2, 2a-1] &\equiv C_0[m-2, 2a-1] \oplus C_0[2m-a-2, a+m-1] \\
&\quad \oplus C_0[2m-2, m+2a-1] \\
C_2[2a-2, a] &\equiv C_0[2a-2, a] \oplus C_0[m+2a-2, a+m] \\
&\quad \oplus C_0[a+m-2, m] \oplus C_0[2m-2, 2m-a] \\
C_2[a-1, a-1] &\equiv C_0[a-1, a-1] \oplus C_0[a+m-1, a+m-1] \\
C_2[a-2, 0] &\equiv C_0[a-2, 0] \oplus C_0[a+m-2, m] \\
&\quad \oplus C_0[2m-2, 2m-a].
\end{aligned}
$$

Entretanto, percebe-se que a operação $C_0[a+m-2, m] \oplus C_0[2m-2, 2m-a]$ acontece mais de uma vez. Desta forma, é possível extrair estes bits para uma variável temporária, tendo assim:

$$T \equiv C_0[a+m-2, m] \oplus C_0[2m-2, 2m-a] \tag{3.9}$$

Reescrevendo as equações, agora com a variável T, tem-se:

$$
\begin{aligned}
C_2[m-1, m-1] &\equiv C_0[m-1, m-1] \oplus C_0[2m-a-1, 2m-a-1] \\
C_2[m-2, 2a-1] &\equiv C_0[m-2, 2a-1] \oplus C_0[2m-a-2, a+m-1] \\
&\quad \oplus C_0[2m-2, m+2a-1] \\
C_2[2a-2, a] &\equiv C_0[2a-2, a] \oplus C_0[m+2a-2, a+m] \oplus T \\
C_2[a-1, a-1] &\equiv C_0[a-1, a-1] \oplus C_0[a+m-1, a+m-1] \\
C_2[a-2, 0] &\equiv C_0[a-2, 0] \oplus T.
\end{aligned}
$$

Com base nas equações anteriores, pode-se apresentar o Algoritmo 4.

Algoritmo 4: Algoritmo de redução modular que utiliza o trinômios irredutíveis com $2 \leq a \leq \lfloor \frac{m+1}{2} \rfloor$.

Entrada: m grau do trinômio, a segundo expoente do trinômio, vetor de bits $C_0[2m-2, 0]$ de tamanho $2m-1$

Saída: vetor de bits $C_2[m-1, 0]$ de tamanho $m-1$

```
1  C_2[m-1] = C_0[m-1] ⊕ C_0[2m-a-1];
2  C_2[a-1] = C_0[a-1] ⊕ C_0[a+m-1];
3  para i ← m-2 até 2a-1 faça
4      C_2[i] = C_0[i] ⊕ C_0[m-a+i] ⊕ C_0[m+i];
5  para i ← a+m-2 até m faça
6      T = C_0[m-1+i] ⊕ C_0[2m-1+i];
7  j = 0;
8  para i ← 2a-2 até a faça
9      C_2[i] = C_0[i] ⊕ C_0[m+i] ⊕ T[j];
10     j++;
11 j = 0;
12 para i ← a-2 até 0 faça
13     C_2[i] = C_0[i] ⊕ T[j];
14     j++;
15 return C_2;
```

Se não fosse pelo artifício da variável temporária, o número de *XORs* seria maior que $2m-2$. Contudo, como é possível perceber através da contagem de operações no algoritmo e nas equações acima, tem-se $2m-2$ operações *XOR*.

3.3.4 Caso para $x^m + x^a + 1$ com $\frac{m+1}{2} < a \leq m$

Uma vez que o recíproco de um trinômio irredutível é irredutível, então, não há razões, de ordem prática, para se utilizar irredutíveis com valores de $a > \frac{m}{2}$.

4 ALGORITMO CONTA-XOR

Em muitas aplicações que utilizam corpos finitos, a redução de elementos é uma operação muito frequente. Deseja-se, portanto, que o custo de redução seja o menor possível. É sabido que este custo depende do polinômio irredutível f. Uma boa estimativa deste custo é avaliar o número de operações elementares e o tempo necessário para que o processo de redução seja feito. Quando se lida com extensões de corpos finitos de característica binária, utiliza-se, basicamente, duas operações elementares: a) ou exclusivo (XOR) e b) deslocamento de bits (Shift). Em hardware, o maior custo de redução é devido ao número de portas $XORs$ e ao tempo de propagação dos sinais para o uso dessas portas. A análise de complexidade da redução é feita em duas frentes: complexidade espacial e complexidade temporal. A complexidade espacial é a quantidade de portas $XORs$ de duas entradas necessárias para realizar a redução. A complexidade temporal é o tempo necessário para que essas portas sejam utilizadas para realizar a redução.

Como dito, a complexidade da redução depende do polinômio irredutível f. Assim, procura-se selecionar o polinômio f que leva á menor complexidade de redução e, consequentemente, á otimização das outras operações que dependem da redução. É reconhecido que, sempre que possível, deve-se escolher um trinômio para definir um corpo binário, uma vez que trinômios levam a operações aritméticas muito eficientes.

Contudo, não existem trinômios irredutíveis para todo m, sendo frequentemente necessário recorrer a pentanômios. No Capítulo 3 foi demonstrado que para trinômios na forma $x^m + x^a + 1$, onde $1 < a \leq \frac{m+1}{2}$, a complexidade espacial de redução de um elemento é $2m - 2$. A complexidade temporal depende do expoente a. No entanto, a estimativa da complexidade espacial e temporal de pentanômios não é trivial.

A inexistência do conhecimento de um pentanômio ótimo, que tenha o menor número de operações possíveis, faz com que a escolha de f seja feita considerando os expoentes a, b e c. Em geral, é recomendado para um dado grau m, o pentanômio que tem os menores expoentes, este é o caso, por exemplo, dos polinômios indicados pelo NIST.

Sabe-se, no entanto, que há polinômios irredutíveis que tem baixa complexidade, mesmo que os expoentes não utilizam a ordem lexicográfica anterior. Na literatura são encontradas poucas famílias de pentanômios, onde é possível estimar sua complexidade. As famílias

são um conjunto de pentanômios onde os expoentes dos polinômios irredutíveis possuem uma determinada relação em comum (FAN; HASAN, 2015). É possível, em muitos casos, avaliar a complexidade dessas famílias em termos dos expoentes do polinômio. E, em muitos casos, tais famílias tem menor complexidade que os pentanômios lexicográficos.

Neste trabalho, depara-se com vários polinômios irredutíveis que possuem baixa complexidade, mas que é difícil classifica-los dentro de uma das famílias conhecidas da literatura. Como o processo de redução é bem conhecido, propõem-se um algoritmo genérico, o qual se deu o nome de Conta-XOR, que permite avaliar a complexidade espacial e temporal de um dado polinômio irredutível. A sistematização e automação da avaliação da complexidade de polinômios irredutíveis permite avaliar para, por exemplo, um determinado grau baixo, todos os polinômios irredutíveis. A partir da classificação, em termos de complexidade espacial do resultado da aplicação do Conta-XOR, foi possível identificar as famílias de polinômios irredutíveis encontradas na literatura. Adicionalmente, esta ferramenta tem sido muito útil para encontrar novas famílias, que possuem complexidade similar áquelas conhecidas e, em alguns casos, até famílias que são melhores que estas. Como resultado da execução do Conta-XOR, descobriu-se duas novas família as quais são apresentadas neste trabalho.

A Seção 4.1 descreve o algoritmo Conta-XOR. O algoritmo consiste de 4 etapas bem definidas. Na primeira etapa, é gerada a matriz de redução (MR). A segunda etapa consiste em retirar as operações de ou exclusivo duplicadas, uma vez que $a \oplus a = 0$, para $a \in \{0, 1\}$. A terceira etapa otimiza as operações. Esta otimização consiste em procurar todas as operações que são executadas mais de uma vez e substituí-las por uma operação intermediária. A última etapa consiste em avaliar a complexidade espacial e temporal. Para uma melhor compreensão do Conta-XOR, o algoritmo será descrito juntamente com uma execução detalhada do algoritmo. A Seção 4.2 contém uma análise de complexidade do algoritmo Conta-XOR. Esta análise de complexidade mostra que, dependendo do polinômio que se está avaliando, o consumo de memória e de processamento é muito elevado. Em alguns casos, foi utilizado um computador de alto desempenho durante semanas para uma avaliação de um único irredutível. Na Seção 4.4 é apresentada uma avaliação e alguns resultados obtidos com a utilização do algoritmo. A Seção 4.5 conclui este capítulo.

4.1 DESCRIÇÃO DO ALGORITMO CONTA-XOR

O objetivo do Conta-XOR é contar eficientemente o número de operações de ou-exclusivo necessárias para realizar a redução deste elemento. Para isso, são executadas as quatro seguintes etapas:

Etapa 1 Geração de uma matriz de redução MR;

Etapa 2 Eliminação dos termos repetidos dois a dois em cada coluna da matriz;

Etapa 3 Substituição de operações repetidas por variáveis temporárias, as quais são armazenadas na Matriz de Elementos Repetidos (MER);

Etapa 4 Cálculo do número de *XORs*.

A Etapa 1 é responsável por proceder a divisão do elemento a reduzir, a partir das equações de equivalência emergidas do polinômio irredutível. O uso sistemático dessas equações permite reduzir bit a bit os termos do elemento a reduzir. O resultado da aplicação dessas equações é adicionado em uma matriz denominada Matriz de Redução (MR). Essa matriz conterá a redução individual de cada termo do polinômio que se deseja reduzir, sem se preocupar com qualquer otimização de operações. Uma vez obtido a MR, o elemento reduzido pode ser computado através da soma de todas a linhas da MR.

A matriz MR terá, provavelmente, muitos elementos repetidos, que estarão na mesma coluna. Tais elementos podem ser eliminados dois a dois. Isso ocorre pois a operação de XOR de dois bits de mesmo valor se anula. A Etapa 2 consiste em retirar da matriz MR essas operações.

A Etapa 3 consiste em buscar na matriz MR as operações de ou exclusivo que se repetem mais de uma vez e substituir cada uma dessas operações por uma variável temporária, que conterá o resultado da operação propriamente dita. Assim, após a terceira etapa, não haverá na matriz MR operações de ou exclusivo repetidas. As operações repetidas são registradas em uma segunda matriz, denominada Matriz de Elementos Repetidos (MER), que contém três colunas. A primeira coluna conterá o identificador da operação, ou seja, da variável temporária. As duas colunas seguintes conterão os bits da operação repetida.

O número total de ou exclusivos será determinado na quarta e última etapa. Esse número será a soma do número de operações

de ou exclusivo restantes na matriz MR com o número de operações registradas na matriz MER. Seja $c^{(i)}$ o número de elementos da coluna i da matriz MR e v_{temp} o número de variáveis temporárias da matriz MER. O número de *XORs* é calculado da seguinte forma:

$$N_{\oplus} = \sum_{i=0}^{m-1} (c^{(i)} - 1) + v_{temp} \tag{4.1}$$

Cada uma destas etapas são apresentadas nas subseções a seguir. Para ajudar o entendimento do algoritmo Conta-XOR, as etapas serão apresentadas por meio de um exemplo numérico. Para isto, seja um corpo binário definido pelo polinômio irredutível $f(x) = x^m + x^a + x^b + x^c + 1 = x^{10} + x^4 + x^3 + x + 1$. Assim, tem-se que $m = 10$, $a = 4$, $b = 3$ e $c = 1$. Os elementos deste corpo podem ser representados na forma de um polinômio de grau até 9. A multiplicação de dois elementos deste corpo, no entanto, pode gerar um polinômio de grau até 18. Seja este polinômio definido por:

$$D(x) = \sum_{i=0}^{18} d_i x^i \text{ onde } d_i \in \{0, 1\}.$$

De forma expandida, tem-se:

$$\begin{aligned} D(x) = & d_{18}x^{18} + d_{17}x^{17} + d_{16}x^{16} + \\ & d_{15}x^{15} + d_{14}x^{14} + d_{13}x^{13} + \\ & d_{12}x^{12} + d_{11}x^{11} + d_{10}x^{10} + \\ & d_9x^9 + d_8x^8 + d_7x^7 + d_6x^6 + \\ & d_5x^5 + d_4x^4 + d_3x^3 + d_2x^2 + \\ & d_1x + d_0. \end{aligned} \tag{4.2}$$

4.1.1 Etapa 1: Montagem da matriz MR

Sejam $f(x) = \sum_{i=0}^{m} f_i x^i, f_i \in \{0, 1\}$ um polinômio irredutível e $D(x) = \sum_{i=0}^{2m-2} d_i x^i, d_i \in \{0, 1\}$ um elemento que se deseja reduzir. O objetivo é determinar

$$D_{red} \equiv D(x) \bmod f(x). \tag{4.3}$$

Pode-se escrever o termo de mais alto grau de $D_{red}(x)$ como

$x^{2m-2} \equiv x^m x^{m-2} \bmod f(x)$. Como $x^m \equiv \sum_{i=0}^{m-1} f_i x^i \bmod f(x)$, tem-se

$$x^{2m-2} \equiv x^{m-2} \sum_{i=0}^{m-1} f_i x^i \bmod f(x).$$

De forma mais geral,

$$x^{2m-2-j} \equiv x^{m-2-j} \sum_{i=0}^{m-1} f_i x^i \bmod f(x). \tag{4.4}$$

Seja MR uma matriz de inteiros com n linhas e $2m-1$ colunas. Seja $G(i) = 1$, se $i > 0$. Se $i \leq 0$ então $G(i) = 0$. Sejam A_i e B_i, $i \geq 0$ inteiros. Seja ainda

$$k_i = \begin{cases} \left\lfloor \frac{m-2}{m-i} \right\rfloor + 1, & \text{se } f_i = 1, \\ 0, & \text{se } f_i = 0. \end{cases}$$

Seja W o peso de $f(x)$, ou seja, o número de termos não nulos de $f(x)$.

Tem-se:

1. $A_0 = 1$ e $B_0 = 1$.

2. Para $i > 0$

   (a) $A_i = \sum_{j=0}^{m} G(k_j - i)$

   (b) $B_i = (W-1)A_{i-1}$.

Seja a o segundo maior grau de $f(x)$, ou seja

$$a = \text{grau}(f(x) + x^m).$$

O número n de linhas da matriz MR será

$$n = \sum_{i=0}^{k_a} B_i. \tag{4.5}$$

A primeira linha da matriz conterá os termos do elemento a reduzir, ou seja,

$$MR[0, i] = d_i, \text{ para } 0 \leq i \leq 2m-2.$$

As próximas linhas serão preenchidas com os termos de $D(x)$ que precisam ser reduzidos, de acordo com a Equação 4.4. Esse preenchimento será feito k_a vezes. À posição da matriz que não contém nenhum elemento ou quando este é removido, atribui-se o valor -1. Valores positivos são interpretados como termos dos polinômios.

Para o exemplo, a matriz MR terá $2m - 1 = 19$ colunas. Por convenção, a coluna mais á esquerda conterá o termo de mais alto grau do polinômio a ser reduzido. A coluna mais á direita conterá o termo independente, ou seja, de grau zero. Também, por convenção, os elementos do polinômio a serem reduzidos serão identificados na matriz MR por sua posição. Por exemplo, o elemento d_{15} será designado simplesmente por 15.

Utilizando-se as operações de aritmética modular, tem-se que

$$x^{18} \equiv x^{10}x^8 \bmod f(x),$$

contudo, $x^{10} \equiv x^4 + x^3 + x + 1 \bmod f(x)$, logo

$$x^{18} \equiv x^8(x^4 + x^3 + x + 1) \bmod f(x),$$

Da mesma forma, é fácil verificar que

$$x^{18} \equiv x^{12} + x^{11} + x^9 + x^8 \mod f(x) \tag{4.6}$$
$$x^{17} \equiv x^{11} + x^{10} + x^8 + x^7 \mod f(x) \tag{4.7}$$
$$x^{16} \equiv x^{10} + x^9 + x^7 + x^6 \mod f(x) \tag{4.8}$$
$$x^{15} \equiv x^9 + x^8 + x^6 + x^5 \mod f(x) \tag{4.9}$$
$$x^{14} \equiv x^8 + x^7 + x^5 + x^4 \mod f(x) \tag{4.10}$$
$$x^{13} \equiv x^7 + x^6 + x^4 + x^3 \mod f(x) \tag{4.11}$$
$$x^{12} \equiv x^6 + x^5 + x^3 + x^2 \mod f(x) \tag{4.12}$$
$$x^{11} \equiv x^5 + x^4 + x^2 + x \mod f(x) \tag{4.13}$$
$$x^{10} \equiv x^4 + x^3 + x + 1 \mod f(x) \tag{4.14}$$

A primeira linha da matriz MR, denotada por L_0, conterá os termos do polinômio $D(x)$, conforme ilustra a Figura 6. As linhas seguintes serão utilizadas para incluir os termos que substituirão cada termo a ser reduzido, conforme as equações acima.

O primeiro termo a ser reduzido será o 18. Substitui-se este termo copiando o seu valor para os bits 12, 11, 9 e 8, conforme a Equação 4.6. A Figura 7 mostra o resultado desta operação.

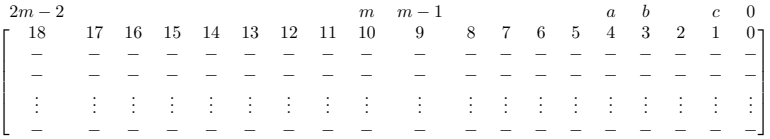

Figura 6 – Matriz de Redução MR com os termos do polinômio a ser reduzido.

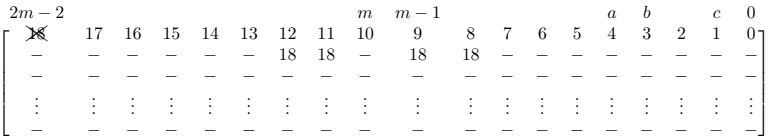

Figura 7 – MR com um passo de redução do termo d_{18}.

Como pode ser visto na Figura 7, o polinômio resultante desta operação é

$$\begin{aligned} D(x) \equiv & d_{17}x^{17} + d_{16}x^{16} + d_{15}x^{15} + \\ & d_{14}x^{14} + d_{13}x^{13} + (d_{12} + d_{18})x^{12} + \\ & (d_{11} + d_{18})x^{11} + d_{10}x^{10} + (d_9 + d_{18})x^9 + \\ & (d_8 + d_{18})x^8 + d_7x^7 + d_6x^6 + d_5x^5 + \\ & d_4x^4 + d_3x^3 + d_2x^2 + d_1x + d_0 \mod f(x) \end{aligned} \tag{4.15}$$

Esse processo de redução é repetido para os termos 17, 16, 15, 14, 13, 12, 11 e 10, obtendo-se a matriz mostrada na figura 8.

Pode-se rearranjar os elementos dessa matriz de forma a facilitar a sua visualização e interpretação, conforme ilustra a Figura 9.

Esse rearranjo de MR permite uma interpretação mais elucidativa do que ocorre em cada passo do processo de redução. As linhas L_{1a}, L_{1b}, L_{1c} e L_{10} correspondem ao primeiro passo de redução dos termos de 18 a 10. Note que esses termos do polinômio $D(x)$ são copiados da linha L_0 para a Linha L_{1a} a partir da posição 12. Esses mesmos termos são copiados para as linhas L_{1b}, L_{1c} e L_{10} para as posições 11, 9 e 8, respectivamente. Pode-se notar que o bit 10 é assentado na posição do expoente a na Linha L_{1a}, na posição b na Linha L_{1b}, na posição c

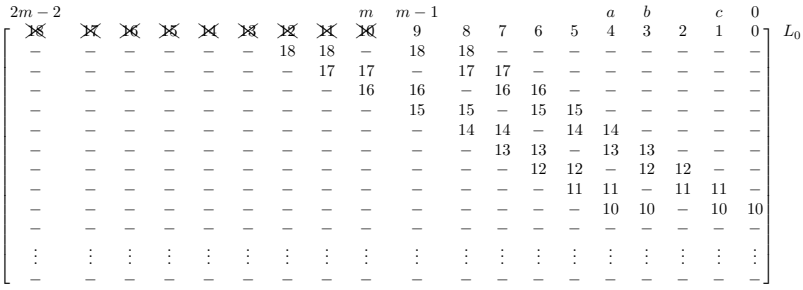

Figura 8 – MR com um passo de redução para os bits de 10 a 18.

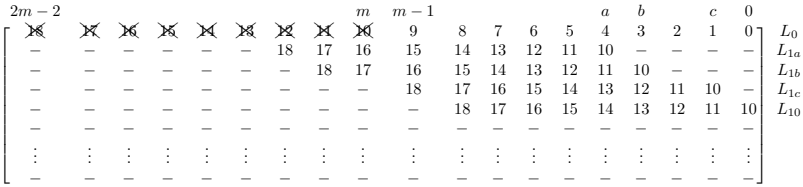

Figura 9 – MR com um passo de redução.

$2m-2$								m	$m-1$					a	b		c	0	
~~16~~	~~16~~	~~16~~	~~16~~	~~16~~	~~16~~	~~16~~	~~16~~	~~16~~	9	8	7	6	5	4	3	2	1	0	L_0
–	–	–	–	–	–	~~16~~	~~17~~	~~16~~	15	14	13	12	11	10	–	–	–	–	L_{1a}
–	–	–	–	–	–	–	~~16~~	~~17~~	16	15	14	13	12	11	10	–	–	–	L_{1b}
–	–	–	–	–	–	–	–	–	18	17	16	15	14	13	12	11	10	–	L_{1c}
–	–	–	–	–	–	–	–	–	–	18	17	16	15	14	13	12	11	10	L_{10}
–	–	–	–	–	–	–	–	–	–	–	–	18	17	16	–	–	–	–	L_{2a}
–	–	–	–	–	–	–	–	–	–	–	–	–	18	17	16	–	–	–	L_{2b}
–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	18	17	16	–	L_{2c}
–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	18	17	16	L_{20}
–	–	–	–	–	–	–	–	–	–	–	–	–	18	17	–	–	–	–	L_{3a}
–	–	–	–	–	–	–	–	–	–	–	–	–	–	18	17	–	–	–	L_{3b}
–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	18	17	–	L_{3c}
–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	18	17	L_{30}

Figura 10 – MR com o segundo passo de redução.

na Linha L_{1c} e na posição zero na Linha L_{10}.

No entanto, a redução ainda não terminou, uma vez que o polinômio equivalente apresentado na matriz MR da Figura 9 ainda possui termos a reduzir. São eles: o bit 18 na coluna 12, os bits 17 e 18 na coluna 11 e os bits 16 e 17 na coluna $m = 10$. É necessário, portanto, um novo passo de redução. Para a redução desses termos, procede-se da mesma forma que o primeiro passo de redução. A Figura 10 mostra o resultado da redução desses elementos.

As linhas L_{2a}, L_{2b}, L_{2c}, L_{20}, L_{3a}, L_{3b}, L_{3c} e L_{30} contém os bits resultantes do segundo passo de redução. Como não há mais termos a reduzir, o polinômio descrito na matriz MR da Figura 10 é o elemento reduzido de $D(x)$. Esse polinômio é o seguinte:

$$\begin{aligned} D(x) \equiv & (d_9 + d_{15} + d_{16} + d_{18})x^9 + \\ & (d_8 + d_{14} + d_{15} + d_{17} + d_{18})x^8 + \\ & (d_7 + d_{13} + d_{14} + d_{16} + d_{17})x^7 + \\ & (d_6 + d_{12} + d_{13} + d_{15} + d_{16} + d_{18})x^6 + \\ & (d_5 + d_{11} + d_{12} + d_{14} + d_{15} + d_{17} + d_{18} + d_{18})x^5 + \\ & (d_4 + d_{10} + d_{11} + d_{13} + d_{14} + d_{16} + d_{17} + d_{17} + d_{18})x^4 + \\ & (d_3 + d_{10} + d_{12} + d_{13} + d_{16} + d_{18} + d_{17})x^3 + \\ & (d_2 + d_{11} + d_{12} + d_{17} + d_{18} + d_{18})x^2 + \\ & (d_1 + d_{10} + d_{11} + d_{16} + d_{17} + d_{17} + d_{18})x + \\ & (d_0 + d_{10} + d_{16} + d_{17}) \mod f(x). \end{aligned} \tag{4.16}$$

$m-1$					a	b		c	0	
9	8	7	6	5	4	3	2	1	0	L_0
15	14	13	12	11	10	–	–	–	–	L_{1a}
16	15	14	13	12	11	10	–	–	–	L_{1b}
18	17	16	15	14	13	12	11	10	–	L_{1c}
–	18	17	16	15	14	13	12	11	10	L_{10}
–	–	–	18	17	16	–	–	–	–	L_{2a}
–	–	–	–	18	17	16	–	–	–	L_{2b}
–	–	–	–	–	–	18	17	16	–	L_{2c}
–	–	–	–	–	–	–	18	17	16	L_{20}
–	–	–	–	18	17	–	–	–	–	L_{3a}
–	–	–	–	–	18	17	–	–	–	L_{3b}
–	–	–	–	–	–	–	18	17	–	L_{3c}
–	–	–	–	–	–	–	–	18	17	L_{30}

Figura 11 – MR representando o polinômio reduzido.

4.1.2 Etapa 2: Eliminação dos elementos repetidos

A partir desta etapa, serão trabalhadas as colunas de 0 até $m-1$ da matriz MR, uma vez que essas colunas já representam o elemento reduzido. É provável que haja em cada coluna termos de $D(x)$ repetidos. Nesta etapa, os termos repetidos desnecessários serão eliminados, o que ocorre da seguinte forma: Se a quantidade de termos repetidos for par, todos os esses termos serão eliminados. Se a quantidade de termos for ímpar, será mantido um único representante do termo.

Em outras palavras, seja $\#d_i$ sendo $i > 0$, o número de termos d_i em uma determinada coluna j, para $0 \leq j \leq (m-1)$. Se $\#d_i$ for par, então elimina-se desta coluna todos os elementos d_i. Caso contrário, mantem-se apenas um representante deste termo.

A matriz montada na etapa anterior já contém o polinômio reduzido. Entretanto, ela não está otimizada. Há várias operações de ou exclusivo desnecessárias. Nesta etapa, essas operações desnecessárias serão retiradas. A Figura 11 ilustra MR das colunas 9 até 0.

Conforme mencionado anteriormente, o ou exclusivo de um termo com ele mesmo resulta no valor 0. Desta forma, a operação pode ser eliminada das colunas da matriz MR. Por exemplo, na Figura 12, o termo 17 da coluna 1 efetua um XOR com ele mesmo. Esta operação

$m-1$					a	b		c	0	
9	8	7	6	5	4	3	2	1	0	L_0
15	14	13	12	11	10	–	–	–	–	L_{1a}
16	15	14	13	12	11	10	–	–	–	L_{1b}
18	17	16	15	14	13	12	11	10	–	L_{2c}
–	18	17	16	15	14	13	12	11	10	L_{20}
–	–	–	18	17	16	–	–	–	–	L_{2a}
–	–	–	–	(18)	(17)	16	–	–	–	L_{2b}
–	–	–	–	–	–	18	17	16	–	L_{2c}
–	–	–	–	–	–	–	(18)	(17)	16	L_{20}
–	–	–	–	(18)	(17)	–	–	–	–	L_{3a}
–	–	–	–	–	18	17	–	–	–	L_{3b}
–	–	–	–	–	–	–	(18)	(17)	–	L_{3c}
–	–	–	–	–	–	–	–	18	17	L_{30}

Figura 12 – Termos repetidos.

não precisa ser executada e os dois elementos 17 podem ser removidos de MR. O mesmo ocorre com os termos 18, na coluna 2, 17, na coluna 4, e 18, na coluna 5.

A Figura 13 mostra a matriz MR sem os pares repetidos.

O polinômio resultante representado na matriz MR, da Figura 13, é ilustrado na Equação 4.17.

$$\begin{aligned} D(x) \equiv & (d_9 + d_{15} + d_{16} + d_{18})x^9 + \\ & (d_8 + d_{14} + d_{15} + d_{17} + d_{18})x^8 + \\ & (d_7 + d_{13} + d_{14} + d_{16} + d_{17})x^7 + \\ & (d_6 + d_{12} + d_{13} + d_{15} + d_{16} + d_{18})x^6 + \\ & (d_5 + d_{11} + d_{12} + d_{14} + d_{15} + d_{17})x^5 + \\ & (d_4 + d_{10} + d_{11} + d_{13} + d_{14} + d_{16} + d_{18})x^4 + \\ & (d_3 + d_{10} + d_{12} + d_{13} + d_{16} + d_{18} + d_{17})x^3 + \\ & (d_2 + d_{11} + d_{12} + d_{17})x^2 + \\ & (d_1 + d_{10} + d_{11} + d_{16} + d_{18})x + \\ & (d_0 + d_{10} + d_{16} + d_{17}) \mod f(x). \end{aligned} \tag{4.17}$$

Neste ponto, não há mais pares de um mesmo termo repetidos

$m-1$					a	b		c	0	
9	8	7	6	5	4	3	2	1	0	L_0
15	14	13	12	11	10	–	–	–	–	L_{1a}
16	15	14	13	12	11	10	–	–	–	L_{1b}
18	17	16	15	14	13	12	11	10	–	L_{1c}
–	18	17	16	15	14	13	12	11	10	L_{10}
–	–	–	18	17	16	–	–	–	–	L_{2a}
–	–	–	–	–	–	16	–	–	–	L_{2b}
–	–	–	–	–	–	18	17	16	–	L_{2c}
–	–	–	–	–	–	–	–	–	16	L_{20}
–	–	–	–	–	18	17	–	–	–	L_{3b}
–	–	–	–	–	–	–	–	18	17	L_{30}

Figura 13 – MR reduzida sem termos repetidos.

nas colunas da matriz MR. Entretanto, constata-se facilmente que existem operações de ou exclusivo repetidas em diferentes colunas de MR. Por exemplo, o ou exclusivo entre os termos d_{10} e d_{16} é repetido nas colunas 0, 1, 3 e 4. Na próxima estap,estas operações redundantes serão retiradas.

4.1.3 Etapa 3: Otimização de operações

Nesta etapa, serão eliminadas as operações repetidas de XOR entre os elementos da matriz MR. Para isso, propõe-se a seguinte heurística: para cada coluna de 0 a $m-1$ da matriz, gera-se todas as combinações 2 a 2 dos seus elementos. Em seguida, conta-se a quantidade de pares iguais em todas as colunas e os classifica-se na ordem dos mais repetidos até os menos repetidos. A repetição de um par é, na verdade, uma operação ou exclusivo redundante. A ideia é realizar a operação uma única vez. Para isso, substitui-se as operações repetidas por uma variável temporária. A variável temporária deve conter o resultado da operação. A substituição será feita em ordem decrescente do número de repetições. Primeiramente, substitui-se o par mais repetido e, em seguida, o segundo par mais repetido e assim por diante, iterativamente.

Acontece que, após a primeira substituição, a matriz MR conterá um novo elemento. Esse elemento é a variável temporária que

Variável	Termo 1	Termo 2
$2m-1$	*termo*	*termo*
$2m$	*termo*	*termo*
$\vdots$	$\vdots$	$\vdots$

Figura 14 – Matriz de Elementos Repetidos (MER).

representa a operação recém substituída. Portanto, é muito provável que o número de operações repetidas seja alterado. Sendo assim, a determinação das operações repetidas deve ser refeita sempre que houver a substituição de uma operação repetida por sua variável temporária.

Da mesma forma que se representa os termos do elemento que está sendo reduzido com números inteiros entre 0 e $2m-2$, utiliza-se um número inteiro para representar as variáveis temporárias. Para designar tais variáveis, utiliza-se números maiores que $2m-1$. A primeira variável temporária será designada por $2m-1$. A segunda por $2m$ e assim por diante.

Ao se classificar e contar as operações repetidas, poderá haver duas ou mais operações com o mesmo número de ocorrências repetidas. Neste caso, escolhe-se o par que possui o menor termo, em termos numéricos, em sua composição. Caso o menor termo seja igual em dois ou mais conjuntos, escolhe-se aquele que tem o segundo menor termo. Certamente só haverá, neste caso, um único conjunto.

A medida que as operações repetidas são substituídas por variáveis temporárias, faz-se necessário o registro dessas operações. O registro é feito em uma segunda matriz. Seja esta a Matriz de Elementos Repetidos (MER). MER terá 3 colunas e um número indefinido de linhas. A primeira coluna da matriz irá conter o identificador da variável temporária, e as colunas seguintes, o identificador dos termos que fazem parte da operação de ou exclusivo. A Figura 14 ilustra a matriz MER.

Esta etapa é finalizada quando não houver mais operações de ou exclusivo repetidas na matriz MR.

A seguir tem-se o exemplo numérico. Conforme mencionado, para efetuar a otimização, gera-se, para cada coluna, a combinação de termos 2 a 2, sendo d_i com $i \geq m$. Após, efetua-se a contagem do número de ocorrências desses pares em todas as colunas. Ao final do processo, toma-se o par com a maior ocorrência e o substitui por uma variável temporária. As variáveis temporárias são designadas por números maiores que $2m-2$. Caso existam dois ou mais pares com o mesmo número de ocorrências, toma-se aquele que possui o termo de menor grau. Isso é possível pois os termos são representados por

Tabela 2 – Lista de todos os pares por coluna.

9	8	7	6	5	4	3	2	1	0
(15, 16)	(14, 15)	(13, 14)	(12, 13)	(11, 12)	(10, 11)	(10, 12)	(11, 12)	(10, 11)	(10, 16)
(15, 18)	(14, 17)	(13, 16)	(12, 15)	(11, 14)	(10, 13)	(10, 13)	(11, 17)	(10, 16)	(10, 17)
(16, 18)	(14, 18)	(13, 17)	(12, 16)	(11, 15)	(10, 14)	(10, 16)	(12, 17)	(10, 18)	(16, 17)
	(15, 17)	(14, 16)	(12, 18)	(11, 17)	(10, 16)	(10, 17)		(11, 16)	
	(15, 18)	(14, 17)	(13, 15)	(12, 14)	(10, 18)	(10, 18)		(11, 18)	
	(17, 18)	(16, 17)	(13, 16)	(12, 15)	(11, 13)	(12, 13)		(16, 18)	
			(13, 18)	(12, 17)	(11, 14)	(12, 16)			
			(15, 16)	(14, 15)	(11, 16)	(12, 17)			
			(15, 18)	(14, 17)	(11, 18)	(12, 18)			
			(16, 18)	(15, 17)	(13, 14)	(13, 16)			
					(13, 16)	(13, 17)			
					(13, 18)	(13, 18)			
					(14, 16)	(16, 17)			
					(14, 18)	(16, 18)			
					(16, 18)	(17, 18)			

Tabela 3 – Número de repetições de pares de termos da matriz MR.

Par	Repetições	Par	Repetições	Par	Repetições	Par	Repetições
(16, 18)	5	(10, 16)	4	(13, 16)	4	(10, 18)	3
(12, 17)	3	(13, 18)	3	(14, 17)	3	(15, 18)	3
(16, 17)	3	(10, 11)	2	(10, 13)	2	(10, 17)	2
(11, 12)	2	(11, 14)	2	(11, 16)	2	(11, 17)	2
(11, 18)	2	(12, 13)	2	(12, 15)	2	(12, 16)	2
(12, 18)	2	(13, 14)	2	(13, 17)	2	(14, 15)	2
(14, 16)	2	(14, 18)	2	(15, 16)	2	(15, 17)	2
(17, 18)	2						

números onde o termo de grau zero é representado pelo número zero, o termo de grau um pelo número um e assim por diante. Caso existam pares que possuam menores termos iguais, toma-se aquele que possui o segundo termo menor.

A Tabela 2 lista todos os pares de termos de cada coluna da matriz MR.

E a Tabela 3 lista os pares repetidos considerando todas as colunas da matriz MR.

O primeiro levantamento mostra que o par que tem mais ocorrências é o $(16, 18)$, que apresenta 5 ocorrências. A Figura 15 mostra essas ocorrências na matriz MR. O passo seguinte é substituir esse par pela variável temporária 19. Essa operação de substituição é então registrada na matriz MER, ilustrada pela Figura 16.

Como a matriz MR foi alterada, é preciso determinar novamente os pares de termos de cada coluna. A Tabela 4 mostra esses novos pares. Comparando esta tabela com a Tabela 2, nota-se que o número de pares diminuiu sensivelmente. A Tabela 5 lista os pares que se repetem mais de uma vez na nova matriz MR, junto com o número de ocorrências.

Pode-se constatar que alguns pares, que antes tinham uma frequência maior, tiveram seu número de repetições diminuído. Isso se deu devido a retirada do par $(16, 18)$ e a inclusão da variável temporária 19.

A Figura 17 mostra a matriz MR após a remoção do par $(16, 18)$ e a inserção da variável temporária 19 em seu lugar.

$m-1$					a	b		c	0	
9	8	7	6	5	4	3	2	1	0	L_0
15	14	13	12	11	10	–	–	–	–	L_{1a}
(16)	15	14	13	12	11	10	–	–	–	L_{1b}
(18)	17	16	15	14	13	12	11	10	–	L_{1c}
–	18	17	(16)	15	14	13	12	11	10	L_{10}
–	–	–	(18)	17	(16)	–	–	–	–	L_{2a}
–	–	–	–	–	–	(16)	–	–	–	L_{2b}
–	–	–	–	–	–	(18)	17	(16)	–	L_{2c}
–	–	–	–	–	–	–	–	–	16	L_{20}
–	–	–	–	–	(18)	17	–	–	–	L_{3b}
–	–	–	–	–	–	–	–	(18)	17	L_{30}

Figura 15 – Ocorrências do par $(16, 18)$ na matriz MR.

Variável	Elemento 1	Elemento 2
[19	16	18]

Figura 16 – Matriz de Elementos Repetidos (MER).

$m-1$					a	b		c	0	
9	8	7	6	5	4	3	2	1	0	L_0
15	14	13	12	11	10	–	–	–	–	L_{1a}
19	15	14	13	12	11	10	–	–	–	L_{1b}
–	17	16	15	14	13	12	11	10	–	L_{1c}
–	18	17	19	15	14	13	12	11	10	L_{10}
–	–	–	–	17	19	–	–	–	–	L_{2a}
–	–	–	–	–	–	19	–	–	–	L_{2b}
–	–	–	–	–	–	–	17	19	–	L_{2c}
–	–	–	–	–	–	–	–	–	16	L_{20}
–	–	–	–	–	–	17	–	–	–	L_{3b}
–	–	–	–	–	–	–	–	–	17	L_{30}

Figura 17 – MR com a variável 19 no lugar dos pares $(16, 18)$.

Tabela 4 – Lista de todos os pares por coluna após a primeira remoção.

9	8	7	6	5	4	3	2	1	0
$(15, 19)$	$(14, 15)$	$(13, 14)$	$(12, 13)$	$(11, 12)$	$(10, 11)$	$(10, 12)$	$(11, 12)$	$(10, 11)$	$(10, 16)$
	$(14, 17)$	$(13, 16)$	$(12, 15)$	$(11, 14)$	$(10, 13)$	$(10, 13)$	$(11, 17)$	$(10, 19)$	$(10, 17)$
	$(14, 18)$	$(13, 17)$	$(12, 19)$	$(11, 15)$	$(10, 14)$	$(10, 17)$	$(12, 17)$	$(11, 19)$	$(16, 17)$
	$(15, 17)$	$(14, 16)$	$(13, 15)$	$(11, 17)$	$(10, 19)$	$(10, 19)$			
	$(15, 18)$	$(14, 17)$	$(13, 19)$	$(12, 14)$	$(11, 13)$	$(12, 13)$			
	$(17, 18)$	$(16, 17)$	$(15, 19)$	$(12, 15)$	$(11, 14)$	$(12, 17)$			
				$(12, 17)$	$(11, 19)$	$(12, 19)$			
				$(14, 15)$	$(13, 14)$	$(13, 17)$			
				$(14, 17)$	$(13, 19)$	$(13, 19)$			
				$(15, 17)$	$(14, 19)$	$(17, 19)$			

Após a inserção da variável 19, faz-se necessário gerar todos os pares. Efetuando este processo novamente, a Tabela 4 lista todos os pares de termos. Contudo, a variável 19 já entra na geração destes pares.

A Tabela 5 lista os pares repetidos considerando todas as colunas, agora com a variável 19.

O segundo levantamento mostra que existem pares com o mesmo número de ocorrências, que neste caso são os pares $(10, 19)$, $(12, 17)$, $(13, 19)$ e $(14, 17)$. Então, escolhe-se o par com o menor termo, que é o

Tabela 5 – Número de repetições de pares de termos da matriz MR.

Par	Repetições	Par	Repetições	Par	Repetições	Par	Repetições
$(10, 19)$	3	$(12, 17)$	3	$(13, 19)$	3	$(14, 17)$	3
$(10, 11)$	2	$(10, 13)$	2	$(10, 17)$	2	$(11, 12)$	2
$(11, 14)$	2	$(11, 17)$	2	$(11, 19)$	2	$(12, 13)$	2
$(12, 15)$	2	$(12, 19)$	2	$(13, 14)$	2	$(13, 17)$	2
$(14, 15)$	2	$(15, 17)$	2	$(15, 19)$	2	$(16, 17)$	2

Tabela 6 – Lista de todos os pares por coluna após a segunda remoção.

9	8	7	6	5	4	3	2	1	0
(15, 19)	(14, 15)	(13, 14)	(12, 13)	(11, 12)	(11, 13)	(12, 13)	(11, 12)	(11, 20)	(10, 16)
	(14, 17)	(13, 16)	(12, 15)	(11, 14)	(11, 14)	(12, 17)	(11, 17)		(10, 17)
	(14, 18)	(13, 17)	(12, 19)	(11, 15)	(11, 20)	(12, 20)	(12, 17)		(16, 17)
	(15, 17)	(14, 16)	(13, 15)	(11, 17)	(13, 14)	(13, 17)			
	(15, 18)	(14, 17)	(13, 19)	(12, 14)	(13, 20)	(13, 20)			
	(17, 18)	(16, 17)	(15, 19)	(12, 15)	(14, 20)	(17, 20)			
				(12, 17)					
				(14, 15)					
				(14, 17)					
				(15, 17)					

Tabela 7 – Número de repetições de pares de termos da matriz MR.

Par	Repetições	Par	Repetições	Par	Repetições	Par	Repetições
(12, 17)	3	(14, 17)	3	(11, 12)	2	(11, 14)	2
(11, 17)	2	(11, 20)	2	(12, 13)	2	(12, 15)	2
(13, 14)	2	(13, 17)	2	(13, 20)	2	(14, 15)	2
(15, 17)	2	(15, 19)	2	(16, 17)	2		

par $(10, 19)$.

Efetuando novamente o mesmo processo, gera-se as Tabelas 6, 7, 8 e 9.

Nesta iteração do algoritmo, o par $(12, 17)$ é a terceira remoção.

É importante destacar que nas Tabelas 8 e 9 acontece uma tomada de decisão importante do algoritmo. Percebe-se que os pares $(11, 14)$, $(11, 20)$ e $(11, 21)$ possuem o mesmo número de ocorrências, e todos iniciam por 11. Para este caso, o algoritmo classifica o par pelo termo seguinte, optando pelo menor, que para o exemplo é o par $(11, 14)$.

Repete-se este processo, de geração e de troca de pares, até que não existam mais pares iguais. Para o exemplo, a Figura 18 ilustra todas as variáveis temporárias utilizadas.

A Figura 19 ilustra a matriz otimizada. No caso, a versão final da matriz de redução.

Ao final de todas as etapas do algoritmo, o polinômio resultante é o da Equação 4.19. Contudo, antes de calcular o polinômio final é necessário calcular as variáveis temporárias, que estão representadas na

Tabela 8 – Lista de todos os pares por coluna após a terceira remoção.

9	8	7	6	5	4	3	2	1	0
(15, 19)	(14, 15)	(13, 14)	(12, 13)	(11, 14)	(11, 13)	(13, 20)	(11, 21)	(11, 20)	(10, 16)
	(14, 17)	(13, 16)	(12, 15)	(11, 15)	(11, 14)	(13, 21)			(10, 17)
	(14, 18)	(13, 17)	(12, 19)	(11, 21)	(11, 20)	(20, 21)			(16, 17)
	(15, 17)	(14, 16)	(13, 15)	(14, 15)	(13, 14)				
	(15, 18)	(14, 17)	(13, 19)	(14, 21)	(13, 20)				
	(17, 18)	(16, 17)	(15, 19)	(15, 21)	(14, 20)				

Tabela 9 – Número de repetições de pares de termos da matriz MR.

Par	Repetições	Par	Repetições	Par	Repetições	Par	Repetições
$(11, 14)$	2	$(11, 20)$	2	$(11, 21)$	2	$(13, 14)$	2
$(13, 20)$	2	$(14, 15)$	2	$(14, 17)$	2	$(15, 19)$	2
$(16, 17)$	2						

$$
\begin{array}{c}
\begin{array}{ccc} \text{Variável} & \text{Elemento 1} & \text{Elemento 2} \end{array} \\
\left[\begin{array}{ccc}
19 & 16 & 18 \\
20 & 10 & 19 \\
21 & 12 & 17 \\
22 & 11 & 14 \\
23 & 13 & 20 \\
24 & 14 & 17 \\
25 & 15 & 19
\end{array}\right]
\end{array}
$$

Figura 18 – Matriz de Elementos Repetidos (MER).

$$
\begin{array}{c}
\begin{array}{cccccccccc} m-1 & & & & & a & b & & c & 0 \end{array} \\
\left[\begin{array}{cccccccccc}
9 & 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\
25 & 15 & 13 & 12 & 15 & 22 & 21 & 11 & 11 & 10 \\
- & 18 & 16 & 13 & 21 & 23 & 23 & 21 & 20 & 16 \\
- & 24 & 24 & 25 & 22 & - & - & - & - & 17
\end{array}\right]
\end{array}
$$

Figura 19 – MR otimizada.

Equação 4.18.

$$\begin{aligned} d_{19} &= d_{16} + d_{18} \\ d_{20} &= d_{10} + d_{19} \\ d_{21} &= d_{12} + d_{17} \\ d_{22} &= d_{11} + d_{14} \\ d_{23} &= d_{13} + d_{20} \\ d_{24} &= d_{14} + d_{17} \\ d_{25} &= d_{15} + d_{19} \end{aligned} \tag{4.18}$$

$$\begin{aligned} D(x) \equiv &(d_9 + d_{25})x^9 + (d_8 + d_{15} + d_{18} + d_{24})x^8 + \\ &(d_7 + d_{13} + d_{16} + d_{24})x^7 + (d_6 + d_{12} + d_{13} + d_{25})x^6 + \\ &(d_5 + d_{15} + d_{21} + d_{22})x^5 + (d_4 + d_{22} + d_{23})^4 + \\ &(d_3 + d_{21} + d_{23})x^3 + (d_2 + d_{11} + d_{21})x^2 + \\ &(d_1 + d_{11} + d_{20})x + (d_0 + d_{10} + d_{16} + d_{17}) \mod f(x). \end{aligned} \tag{4.19}$$

4.1.4 Etapa 4: Contagem de portas *XOR*

Nesta última etapa, acontece a contagem do número de portas *XORs* necessárias. O número de operações $N_{\oplus}$ de ou exclusivo final será dado por o número de operações $N_{\oplus_{MR}}$ que restaram na matriz MR adicionados do número de variáveis temporárias $N_{\oplus_{MER}}$, registradas na matriz MER.

Seja $c^{(i)}$ o número de elementos da coluna i de MR. $N_{\oplus_{MR}}$ é dado por

$$N_{\oplus_{MR}} = \sum_{i=0}^{m-1} c^{(i)} - m.$$

$N_{\oplus_{MER}}$ será o número de linhas da matriz MER. Assim,

$$N_{\oplus} = N_{\oplus_{MR}} + N_{\oplus_{MER}} = \sum_{i=0}^{m-1} c^{(i)} - m + N_{\oplus_{MER}}.$$

Para o exemplo, tem-se os seguintes resultados: Então, para este exemplo, tem-se:

$$v_{temp} = 7$$

e

$$\sum_{i=0}^{9}(c^{(i)} - 1) + 7 = 24 + 7 = 31.$$

Logo, o número total de operações para reduzir qualquer elemento utilizando $x^{10} + x^4 + x^3 + x + 1$ é 31 operações *XOR*.

4.2 ANÁLISE DE COMPLEXIDADE DO CONTA-XOR

Esta seção apresenta uma análise de complexidade do Conta-XOR. A análise de complexidade é feita, separadamente, para cada etapa do algoritmo, considerando o consumo de memória e a quantidade de operações elementares necessárias á sua execução.

Denomina-se complexidade temporal o número de operações realizadas durante a execução do algoritmo e complexidade espacial o consumo de memória. Nessas análises, quando não especificado o contrário, será levada em consideração o pior caso. Considera-se operação elementar toda e qualquer operação simples que o algoritmo executa, tal como a soma ou multiplicação de inteiros ou a cópia de um termo de um vetor para outro vetor.

4.2.1 Análise temporal

A seguir será apresentada a análise temporal. Esta análise será demonstrada para cada etapa do algoritmo.

4.2.1.1 Etapa 1: Montagem da matriz MR

Esta etapa corresponde a montagem da matriz de redução MR. Como visto na Seção 4.1.1, esta matriz tem n linhas e $2m - 1$ colunas. Nesta etapa, inicialmente são alocados os termos do polinômio a ser reduzido na primeira linha de MR. Isso envolve um total de $2m - 1$ operações.

Em seguida, inicia-se o processo de redução propriamente dito. Este processo consiste em copiar os termos de m até $2m - 2$ nas linhas seguintes. Neste processo, há dois casos a considerar: os elementos que já estão reduzidos e os elementos a reduzir.

Para o primeiro, o processo será efetuado n vezes, onde n é o número de linhas da matriz, conforme definido na Subseção 4.1.1. Este

processo irá gerar o total de nm alocações, uma vez que o algoritmo insere o valor de -1 na matriz MR, quando não existe elemento entre 0 e $m-1$.

Para o segundo caso, isto é, para os elementos que precisam ser reduzidos, o número de elementos a serem copiados é dado na Equação 4.20. Esta equação depende de cada um dos expoentes do polinômio irredutível. Assim, seja C o conjunto que contém os expoentes do polinômio irredutível menores que m. Então

$$N_e = \sum_{j \in C} \left[\sum_{i=0}^{k_j} ((m-1) - i(m-j)) \right]. \tag{4.20}$$

Ao final da montagem, o número total de operações é dado por $2m - 1 + nm + N_e$.

4.2.1.2 Etapa 2: Eliminação de elementos repetidos

Nesta etapa, conforme descrito na Subseção 4.1.2, a matriz MR já contém o polinômio reduzido nas colunas entre 0 e $m-1$. Então a eliminação de elementos repetidos será feita somente nessas colunas.

Para todas as colunas entre 0 e $m-1$ são executados os seguintes passos:

Passo 1 - Iniciando na primeira linha, procura-se o primeiro elemento e_1, diferente de -1.

Passo 2 - A partir da linha seguinte a do elemento e_1, procura-se um elemento igual e_1 até o último elemento da coluna.

Passo 3 - Caso seja encontrado um elemento e_2 igual a e_1, atribui-se -1 para as posições de e_1 e e_2.

Passo 4 - A partir da posição seguinte de e_1, busca-se um novo elemento e_1 diferente de -1. Repete-se o passo 3 até que e_1 seja o último elemento da coluna.

Para o cálculo da complexidade da etapa de eliminação de elementos repetidos, considera-se o percorrimento das linhas de cada coluna da matriz MR. O pior caso é quando todos os elementos da coluna são diferentes de -1 e não existem repetidos. A busca então é dado pela grandeza de $\mathcal{O}(n^2)$. Entretanto, é necessário considerar o custo do percorrimento das m colunas. Logo, o número total de operações neste processo é da grandeza de $\mathcal{O}(n^2 m)$.

4.2.1.3 Etapa 3: Otimização de operações

A etapa de otimização de operações é a etapa com a complexidade mais alta do algoritmo. Isto ocorre pois são necessárias muitas iterações para a geração dos pares e busca de elementos na matriz.

No processo de geração de pares, geram-se pares para todas as m colunas da matriz. A quantidade de operações neste processo é dado pelo percorrimento das m colunas da matriz, efetuando a geração de pares dois a dois, como descrito na Seção 4.1.3.

Um limite superior para o número de pares gerados ocorre quando a matriz está completa e não apresenta nenhum elemento -1. Neste caso, a geração dos pares é a combinação de n elemento dois a dois, que resulta em $\frac{n(n-1)}{2}$. Isso será feito para todas as m colunas. Logo, tem-se a complexidade de $\mathcal{O}(n^2m)$.

A contagem de pares repetidos consiste em um percorrimento nos pares gerados, e será somado os pares que estiverem repetidos. Este processo é efetuado em $\mathcal{O}(n)$. Considerando esta complexidade a anterior, o total é de $\mathcal{O}(n^3m)$.

No processo de busca de pares na matriz MR, tem-se a busca nas m colunas pelas n linhas da matriz. Para esta busca, tem-se a complexidade de $\mathcal{O}(nm)$.

Por fim, existe o número de inserções na matriz MER, detalhada na Seção 4.1.3. Em um caso médio, serão necessárias $\frac{n}{2}$ inserções, sendo n o número de linhas da matriz MR. Entretanto, cada inserção é de três elementos, pois serão inseridas as variáveis temporárias e os dois termos que as compõem. Logo, o número final de inserções será de $\frac{3n}{2}$.

Ao final da etapa, somando as complexidades, tem-se a complexidade de $\mathcal{O}(n^3m + nm + \frac{3n}{2})$.

4.2.1.4 Etapa 4: Contagem de portas *XOR*

Na última etapa, o algoritmo irá percorrer as $m - 1$ colunas da matriz MR, contanto todos os elementos diferentes de -1 nas linhas de cada coluna. Seja n o número de linhas de cada coluna. Logo, em todos os casos, o algoritmo irá executar $n(m - 1)$ operações. A este cálculo é necessário adicionar a contagem do número de linhas da matriz MER. Portanto, o número total de operações será de $\mathcal{O}(n(m - 1))$.

4.2.2 Análise espacial

A seguir será apresentado a análise espacial. Esta análise, seguindo o mesmo modelo que a anterior, será demonstrada para cada etapa do algoritmo.

4.2.2.1 Etapa 1: Montagem da matriz MR

Conforme descrito na análise temporal, foi dado o número de alocações para o processo de montagem da matriz. Para a análise espacial, utiliza-se a mesma linha de raciocínio.

Para algumas arquiteturas de computadores, um inteiro é representado utilizando 4 bytes. Portanto, para a matriz MR serão necessários $4(2m - 1 + nm + N_e))$ bytes.

4.2.2.2 Etapa 2: Eliminação de elementos repetidos

No processo de eliminação de repetidos, a complexidade temporal diminui, uma vez que serão trabalhados apenas os termos das colunas 0 até $m-1$. Além disso, se cada coluna possuir muitos elementos repetidos, o número de linhas da matriz irá diminuir.

Contudo, no pior caso, nenhum elemento será eliminado. Desta forma, o número de bytes utilizados será de $4(2m - 1 + nm + N_e))$.

4.2.2.3 Etapa 3: Otimização de operações

Nesta etapa, a matriz MR irá diminuir, uma vez que há a remoção dos pares repetidos, gerando a matriz MER. Entretanto, no pior caso, não existe remoção dos pares e, portanto, não é necessário gerar a matriz MER. Desta forma, a complexidade espacial não será modificada.

4.2.2.4 Etapa 4: Contagem de portas *XOR*

O tamanho da matriz nesta etapa não irá modificar, uma vez que é efetuada apenas a contagem dos termos nas matrizes MR e MER.

Tabela 10 – Complexidade espacial e temporal de alguns polinômios.

Polinômio irredutível	Complexidade temporal	Complexidade espacial
$x^5 + x^2 + 1$	$\approx$ 48 operações	$\approx$ 68 bytes
$x^{10} + x^4 + x^3 + x + 1$	$\approx$ 10598 operações	$\approx$ 452 bytes
$x^{155} + x^{93} + x^{62} + x^{31} + 1$	$\approx$ 31384 operações	$\approx$ 9188 bytes
$x^{212} + x^{211} + x^{210} + x + 1$	$\approx$ 9528443 operações	$\approx$ 178928 bytes
$x^{975} + x^{974} + x^{973} + x + 1$	$\approx$ 926386497 operações	$\approx$ 3798600 bytes

4.2.3 Exemplos

Com o intuito de demonstrar a complexidade do algoritmo Conta-XOR, foram selecionados alguns polinômios irredutíveis como exemplo. Para cada polinômio, foi calculado o número de operações e a quantidade de *bytes* para a execução do algoritmo. A Tabela 10 apresenta os resultados obtidos.

Por exemplo, se um processador executa 24512 instruções por segundo, o polinômio de grau 975 da Tabela 10 levaria cerca de 10 horas para processar e alocaria cerca de $3,7986$ megabytes. Por esta razão, é difícil efetuar a computação do algoritmo para polinômios de grau muito elevado e/ou com muitos termos.

4.3 COMPARAÇÕES DO RESULTADO DO CONTA-XOR

A fim de testar o algoritmo Conta-XOR, utilizou-se as famílias de trinômios e pentanômios irredutíveis, apresentados pela literatura. Por exemplo, no trabalho de Wu (2002) é apresentado a complexidade da aritmética modular para trinômios irredutíveis, conforme descrito no Capítulo 3. No Capítulo 1 deste trabalho, também foram apresentadas diversas famílias de pentanômios irredutíveis de interesse. O resultado das comparações estão descritos a seguir.

4.3.1 Complexidade em trinômios

De acordo com Wu (2002), dado um trinômio do formato $f(x) = x^m + x^a + 1$, a complexidade da aritmética modular polinomial pode ser efetuada em $2m - 2$ operações *XORs*. Isto, para quando $a \leq \frac{m}{2}$. O valor da complexidade se torna $\frac{3m}{2} - 1$, quando $a = \frac{m}{2}$.

Quando utilizado um trinômio irredutível como entrada do algoritmo Conta-XOR, a complexidade apresentada é de $2m - 2$ operações *XOR*, quando o valor de $a \neq \frac{m}{2}$. Quando o trinômio apresenta $a = \frac{m}{2}$,

Tabela 11 – Tabela de comparação de resultados para trinômios.

Trinômio irredutível	No. de XORs Algoritmo Conta-XOR	No. de XORs esperado
$x^5 + x^2 + 1$	8	8
$x^{15} + x + 1$	28	28
$x^{15} + x^{14} + 1$	28	28
$x^{18} + x^9 + 1$	26	26
$x^{879} + x^{868} + 1$	1756	1756
$x^{882} + x^{539} + 1$	1762	1762
$x^{882} + x^{639} + 1$	1762	1762

Tabela 12 – Tabela de comparação de resultados para pentanômios *equally spaced*.

Pentanômio *equally spaced* irredutível	No. de XORs Algoritmo Conta-XOR	No. de XORs esperado
$x^{20} + x^{15} + x^{10} + x^5 + 1$	34	34
$x^{100} + x^{75} + x^{50} + x^{25} + 1$	174	174

o valor das operações cai para $\frac{3m}{2} - 1$. Os resultados computados pelo algoritmo podem ser visualizados na Tabela 11. Na primeira coluna da tabela é apresentado o trinômio de entrada, na segunda coluna, o número de operações *XOR* que o algoritmo resultou e, na última coluna, o número de operações esperado de acordo com Wu (2002).

4.3.2 Complexidade em pentanômios

Diferentemente da complexidade de trinômios, a de pentanômios não é definida facilmente. Por isto, a literatura apresenta famílias de pentanômios irredutíveis. Estas famílias apresentam boas propriedades na redução modular. Por exemplo, a família conhecida como *equally spaced polynomials* (ESP) do trabalho de Hasan, Wang e Bhargava (1992), é uma das melhores conhecidas para efetuar a redução modular polinomial. O pentanômio *equally spaced* apresenta o seguinte formato:

$$x^{4c} + x^{3c} + x^{2c} + x^c + 1, \text{ onde } c = 5^i \text{ e } i \geq 0.$$

Para esta família a complexidade resultante é $\frac{7m}{4} - 1$ XORs. A Tabela 12 apresenta a comparação entre o número de *XORs* resultante do Conta-XOR e o valor apresentado no trabalho de Hasan, Wang e Bhargava (1992).

Outra família apresentada na literatura se encontra no trabalho de Rodríguez-Henríquez e Koç (2003), que tem o seguinte formato:

$$x^m + x^{b+1} + x^b + x + 1, \text{ onde } 1 < b < m - 1.$$

Tabela 13 – Tabela de comparação de resultados para pentanômios $x^m + x^{b+1} + x^b + x + 1$.

Pentanômio irredutível	No. de XORs Algoritmo Conta-XOR	No. de XORs esperado
$x^{53} + x^{16} + x^{15} + x + 1$	171	188
$x^{35} + x^8 + x^7 + x + 1$	109	118

A complexidade dessa família é $4m + 2b - 1$ *XORs*. Conforme efetuado anteriormente, a Tabela 13 apresenta os resultados do Conta-XOR e o resultado esperado, dado um polinômio irredutível no formato da família apresentada.

Verifica-se que existe uma divergência entre o resultado algoritmo e o resultado proposto na literatura. Isto ocorre devido a algumas otimizações que o Conta-Xor executa e que não eram previstas no trabalho de Rodríguez-Henríquez e Koç (2003).

Para as demais famílias que são apresentadas, como por exemplo, as que estão no trabalho de Cilardo (2013) e no trabalho de Reyhani-Masoleh e Hasan (2004), não é possível efetuar uma comparação. O motivo é que os autores reutilizam portas lógicas do processo de multiplicação no processo de redução, e não apresentam o número de operações *XORs* para a redução modular polinomial. Eles apresentam o número de *XORs* da multiplicação e da redução. No Capítulo 5 é apresentado uma comparação mais justa com essas famílias.

4.3.3 Pentanômios do NIST

O *National Institute of Standards and Technology* (NIST) é um instituto que recomenda diversos parâmetros em aplicações tecnológicas. Um desses parâmetros são polinômios irredutíveis que são utilizados principalmente em criptografia de curvas elípticas. Os polinômios recomendados pelo NIST tem os graus 163, 233, 283, 409 e 571. Para dois graus destes polinômios, existem trinômios irredutíveis, que são os graus, 233 e 409. Para os graus restantes, não existe trinômio. Neste caso, utiliza-se um pentanômio. Os pentanômios recomendados pelo NIST são:

$$x^{163} + x^7 + x^6 + x^3 + 1,$$
$$x^{283} + x^{12} + x^7 + x^5 + 1,$$
$$x^{571} + x^{10} + x^5 + x^2 + 1.$$

A Tabela 14 apresenta o resultado da computação do Algoritmo Conta-XOR para os pentanômios recomendados pelo NIST.

Tabela 14 – Tabela de número de *XORs* para pentanômios do NIST.

Pentanômio irredutível	No. de XORs Algoritmo Conta-XOR
$x^{163} + x^7 + x^6 + x^3 + 1$	571
$x^{283} + x^{12} + x^7 + x^5 + 1$	862
$x^{571} + x^{10} + x^5 + x^2 + 1$	2003

Tabela 15 – Tabela de número de *XORs* para pentanômios de grau 163.

Pentanômio irredutível	No. de XORs Algoritmo Conta-XOR
$x^{163} + x^{89} + x^{74} + x^{15} + 1$	487
$x^{163} + x^{100} + x^{63} + x^{37} + 1$	487
$x^{163} + x^{26} + x^{23} + x^3 + 1$	515
$x^{163} + x^{60} + x^{59} + x + 1$	545
$x^{163} + x^{82} + x^6 + x + 1$	567
$x^{163} + x^{97} + x^{75} + x^{22} + 1$	567
$x^{163} + x^7 + x^6 + x^3 + 1$	571

Com base no número de operações apresentado anteriormente, deseja-se encontrar um pentanômio que utilize uma quantidade menor de operações. Para descobrir se existe um pentanômio melhor, gerou-se todos os pentanômios irredutíveis para os graus 163, 283 e 571. Em seguida, rodou-se os pentanômios através do algoritmo Conta-Xor. Infelizmente, para os graus 283 e 571 o número de pentanômios é muito grande e não foi possível a determinação do número de operações para todos eles.

A Tabela 15 apresenta alguns pentanômios irredutíveis de grau 163, que são melhores que o recomendado pelo NIST. Com isso, verificou-se que existe 18 pentanômios melhores para fazer a aritmética modular.

Para o grau 283, rodou-se alguns pentanômios e não o número total deles. O resultado, ainda que parcial, demonstrou pentanômios melhores que o recomendado. Como, por exemplo, o polinômio $x^{283} + x^{160} + x^{123} + x^{37} + 1$, que utiliza 847 operações *XORs*. A Tabela 16 apresenta mais alguns resultados.

O grau 571, o qual também se recomendado um pentanômio, não foi possível a execução do Conta-Xor para todos os pentanômios deste grau. Entretanto, os resultados obtidos indicam que existem pen-

Tabela 16 – Tabela de número de *XORs* para pentanômios de grau 283.

Pentanômio irredutível	No. de XORs Algoritmo Conta-XOR
$x^{283} + x^{172} + x^{111} + x^{61} + 1$	847
$x^{283} + x^{12} + x^7 + x^5 + 1$	862

tanômios melhores. A Tabela 17 apresenta alguns dos polinômios que são melhores que o recomendado.

Tabela 17 – Tabela de número de *XORs* para pentanômios de grau 571.

Pentanômio irredutível	No. de XORs Algoritmo Conta-XOR
$x^{571} + x^{465} + x^{359} + x^{106} + 1$	1604
$x^{571} + x^{389} + x^{207} + x^{182} + 1$	1685
$x^{571} + x^{549} + x^{527} + x^{22} + 1$	1688
$x^{571} + x^{353} + x^{218} + x^{135} + 1$	1711
$x^{571} + x^{10} + x^5 + x^2 + 1$	2003

4.4 FAMÍLIAS DE PENTANÔMIOS DESCOBERTAS

A literatura apresenta diversas famílias de pentanômios irredutíveis. O algoritmo descrito anteriormente teve o intuito de descobrir novas famílias que utilizam um número menor de operações *XORs*.

Para determinar se existiam novas famílias, foi necessário definir parâmetros para execução do algoritmo. Para isto, foi definido graus de pentanômios utilizados na prática. Um exemplo são os graus recomendados para curvas elípticas: 163, 283 e 571 (BROWN et al., 2001; HANKERSON; HERNANDEZ; MENEZES, 2000; HANKERSON; MENEZES; VANSTONE, 2003). Foi utilizado graus menores para que o processamento fosse mais rápido. O grau selecionado foi o 19, por apresentar 159 pentanômios.

Após a definição dos graus, utilizou-se o software SAGE (MATH, 2013) para a geração dos pentanômios irredutíveis em $\mathbb{F}_2$. Os graus 571 e 283 apresentaram um número muito grande de pentanômios e, por isso, foram avaliados parcialmente. Para os outros graus, isto é 19 e 163, foi executado o algoritmo para todos os pentanômios.

As entradas do algoritmo eram os expoentes do polinômio na forma de uma lista, e os resultados também eram os expoentes, em forma de lista, concatenados com um separador e com o número de operações *XOR*. A seguir temos um exemplo de saída, para o polinômio $x^{10} + x^4 + x^3 + x + 1$, do algoritmo: [10,4,3,1,0]:31.

Munido dos resultados de todos os polinômios para os graus 19 e 163, foi iniciada a análise destes pentanômios. Gerou-se uma tabela para 19 e uma para 163, contendo o número de *XOR* e expoentes. Em seguida, pegou-se os polinômios que tinham o menor número de operações. Com estes, foram efetuados testes nas propriedades dos

expoentes, como por exemplo, a divisibilidade entre eles, se existia uma distância fixa ou se o m era composição dos outros expoentes. Com os resultados desta análise, foi possível a descoberta de duas novas famílias:

$$x^{2b+c} + x^{b+c} + x^b + x^c + 1$$

e

$$x^{b+2c} + x^{b+c} + x^b + x^c + 1.$$

4.5 CONCLUSÃO

O Algoritmo Conta-XOR, descrito acima, foi desenvolvido com o intuito de contar o número de operações *XORs* que compõem a aritmética modular polinomial, otimizando estas operações com o auxílio de uma heurística. Ainda existe muito o que ser trabalhado no algoritmo, para que este apresente sempre o melhor resultado. Entretanto, o algoritmo já apresentou resultados interessantes, ou seja, melhores que os existentes na literatura.

A análise dos resultados do algoritmo possibilitou a descoberta de duas novas famílias de pentanômios irredutíveis. Contudo, faz-se necessário um estudo mais detalhado da melhoria que estas famílias proporcionaram.

Para a primeira família, isto é $x^{2b+c} + x^{b+c} + x^b + x^c + 1$, este estudo foi realizado e apresentado no Capítulo 5. O estudo da segunda família, isto é $x^{b+2c} + x^{b+c} + x^b + x^c + 1$, é proposto como um trabalho futuro, uma vez que esta família demonstrou que é necessário uma compreensão maior do processo de redução que a envolve.

A razão pela qual é necessário um estudo mais detalhado da segunda família é que a mesma indica um algoritmo para redução modular mais complexo. Isto acontece porque o número de passos desta família pode ser alto, diferentemente da família onde $m = 2b + c$. Para exemplificar, considere o polinômio $f(x) = x^{691} + x^{669} + x^{647} + x^{22} + 1$. O número de passos de redução deste é dado por

$$k_a = \left\lfloor \frac{m-2}{m-a} \right\rfloor + 1 = \left\lfloor \frac{691-2}{691-669} \right\rfloor + 1 = 32,$$

em contra partida, um pentanômio de mesmo grau para a família $m = 2b + c$ possui um número menor de nr. Seja o pentanômio $f(x) = x^{691} + x^{448} + x^{243} + x^{205} + 1$. O número de passos de redução deste é

dado por

$$k_a = \left\lfloor \frac{691 - 2}{691 - 448} \right\rfloor + 1 = 3.$$

Conforme será demonstrado no Capítulo 5, o número de passos de redução influencia a determinação do número de *XORs*, e a forma como é gerado o algoritmo de redução. De fato, no Capítulo 5 dase o número exato de *XORs* necessários para pentanômios da forma $x^{2b+c} + x^{b+c} + x^b + x^c + 1$.

5 FAMÍLIA $X^{2B+C} + X^{B+C} + X^B + X^C + 1$

Conforme apresentado anteriormente neste trabalho, os pentanômios sobre $\mathbb{F}_2$ são de grande interesse. Neste capítulo, será apresentado o resultado obtido para uma nova família, que apresenta boas propriedades em termos de complexidade espaço e tempo. Para isto, mostra-se quantas operações *XORs* são necessárias para a redução de um elemento de um corpo finito binário. É importante ressaltar que este capítulo foi submetido como um artigo para a revista *IEEE Transactions on Computer* (BANEGAS; CUSTÓDIO; PANARIO, 2015).

O formato de pentanômios desta nova família é:

$$f(x) = x^m + x^a + x^b + x^c + 1$$

onde $m = 2b + c$, $a = b + c$, ou seja

$$f(x) = x^{2b+c} + x^{b+c} + x^b + x^c + 1.$$

Dado $a(x), b(x) \in \mathbb{F}_{2^m}$, deseja-se efetuar a multiplicação destes dois elementos, gerando assim um terceiro polinômio, dado por $D_0 = a(x)b(x)$. O polinômio D_0 pode ter grau maior que m, podendo ser no máximo de $2m - 2$. Para que este polinômio pertença a $\mathbb{F}_{2^m}$, é necessário reduzi-lo utilizando a aritmética modular. Isto é, divide-se o mesmo uma vez por f; o resto será o polinômio reduzido buscado.

Então, uma forma de representação de D_0 é

$$D_0 = \sum_{i=0}^{2m-2} d_i x^i. \tag{5.1}$$

Seja D_0 polinômio a reduzir módulo f

$$D_{red} \equiv D_0 \pmod{f(x)}. \tag{5.2}$$

O número máximo de passos de redução, em função do expoente a, foi definido anteriormente no Capítulo 3, que se baseia no trabalho de Sunar e Koç (1999):

$$k_a = \left\lfloor \frac{m-2}{m-a} \right\rfloor + 1 \tag{5.3}$$

Contudo, nesta família o valor de a é $b + c$, logo

$$k_{b+c} = \left\lfloor \frac{2b + c - 2}{2b + c - b - c} \right\rfloor + 1 = \left\lfloor \frac{c - 2}{b} \right\rfloor + 3 = \begin{cases} 2 & \text{se } c = 1, \\ 3 & \text{se } c > 1. \end{cases} \quad (5.4)$$

Utilizando a mesma fórmula, pode-se calcular os valores para b e c,

$$k_b = \left\lfloor \frac{2b + c - 2}{2b + c - b} \right\rfloor + 1 = \left\lfloor \frac{b - 2}{b + c} \right\rfloor + 2 = 2, \quad (5.5)$$

e

$$k_c = \left\lfloor \frac{2b + c - 2}{2b + c - c} \right\rfloor + 1 = \left\lfloor \frac{c - 2}{2b} \right\rfloor + 2 = \begin{cases} 1 & \text{se } c = 1, \\ 2 & \text{se } c > 1. \end{cases} \quad (5.6)$$

Isto demonstra que esta família de pentanômios não necessita mais que 3 passos de redução. Este fato será necessário para as etapas apresentadas nas seções subsequentes.

5.1 REDUÇÕES

Nesta seção são apresentadas as equações para as reduções, dado o número de passos de redução. A Figura 20 ilustra que é possível desmembrar D_0 em duas partes. A parte A_0 representa os termos de D_0 com grau maior ou igual a m, enquanto a parte B_0 representa a parte dos elementos com grau menor que m. A parte A_0 é divida por f e seu resto, denominado de D_1, é utilizado na mesma operação até que o grau seja menor que m. Para esta família, quando $k_{b+c} = 2$, tem-se $A_2 = 0$, e para $k_{b+c} = 3$, tem-se $A_3 = 0$, o que torna desnecessário proceder mais reduções.

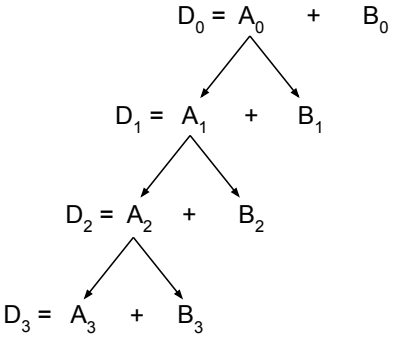

Figura 20 – Passos de redução.

Para o caso onde $k_{b+c} = 2$, a partir de A_2, não é mais necessário fazer a redução. Dessa forma, $A_2 = 0$ e a Equação (5.7) é aquela que representa o elemento reduzido.

$$D_{red} = B_0 + B_1 + B_2. \tag{5.7}$$

Do mesmo modo, para o caso onde $k_{b+c} = 3$, a partir de A_3, não é mais necessário reduzir. Deste modo, $A_3 = 0$ e a Equação (5.8) é aquela que representa o elemento reduzido.

$$D_{red} = B_0 + B_1 + B_2 + B_3. \tag{5.8}$$

Como o número máximo de reduções para está família é 3, as equações serão até $A_3 = 0$.

5.1.1 Número de termos para A_r e B_r

Seja $G(i) = 1$ e i um valor inteiro, se $i > 0$ e $G(i) = 0$, se $i \leq 0$. Seja r um passo de redução. Dado o número preciso de termos para A_r e B_r, para $r \geq 0$, utiliza-se o k_{b+c}, k_b e k_c apresentado nas Equações (5.3), (5.5) e (5.6). Tem-se:

1. O número de termos, isto é, os valores de intervalo do coeficiente são $A_0 = 1$ e $B_0 = 1$.

2. Para $r > 0$, O número de termos de A_r é $G(k_{b+c} - r) + G(k_b - r) + G(k_c - r)$ e o número de termos de B_r é 4 vezes o número de termos de A_{r-1}.

5.1.2 Determinação de A_0 e B_0

Conforme descrito anteriormente, D_0 pode ser repartido em duas partes. Verifica-se que a parte A_0 contém os os termos de grau m a $2m-2$, sendo assim necessária a redução.

$$D_0 = A_0 + B_0 = \sum_{i=m}^{2m-2} d_i x^i + \sum_{i=0}^{m-1} d_i x^i. \tag{5.9}$$

Tem-se então

$$A_0 = \sum_{i=m}^{2m-2} d_i x^i,$$

e

$$B_0 = \sum_{i=0}^{m-1} d_i x^i. \tag{5.10}$$

5.1.3 Determinação de A_1 e B_1

Dividindo-se o termo de maior grau de A_0 por f e tomando-se o resto, tem-se

$$D_1 = \sum_{i=0}^{m-2} d_{i+m} x^{i+a} + \sum_{i=0}^{m-2} d_{i+m} x^{i+b} + \sum_{i=0}^{m-2} d_{i+m} x^{i+c} + \sum_{i=0}^{m-2} d_{i+m} x^i.$$

Fazendo-se uma mudança de variáveis para que o expoente de x seja i, tem-se:

$$D_1 = \sum_{i=a}^{m+a-2} d_{i+m-a} x^{i+a} + \sum_{i=b}^{m+b-2} d_{i+m-b} x^{i+b} + \sum_{i=c}^{m+c-2} d_{i+m-c} x^{i+c} + \sum_{i=0}^{m-2} d_{i+m} x^i.$$

Dessa forma, o expoente de x é i. Agora, separando a parte a reduzir da parte reduzida, tem-se

$$A_1 = \sum_{i=m}^{m+a-2} d_{i+(m-a)}x^i + \sum_{i=m}^{m+b-2} d_{i+(m-b)}x^i + \sum_{i=m}^{m+c-2} d_{i+(m-c)}x^i, \tag{5.11}$$

$$B_1 = \sum_{i=a}^{m-1} d_{i+(m-a)}x^i + \sum_{i=b}^{m-1} d_{i+(m-b)}x^i + \sum_{i=c}^{m-1} d_{i+(m-c)}x^i + \sum_{i=0}^{m-2} d_{i+m}x^i.$$

Como neste caso $m = 2b + c$ e $a = b + c$, tem-se:

$$B_1 = \sum_{i=b+c}^{2b+c-1} d_{i+b}x^i + \sum_{i=b}^{2b+c-1} d_{i+b+c}x^i + \sum_{i=c}^{2b+c-1} d_{i+2b}x^i + \sum_{i=0}^{2b+c-2} d_{i+2b+c}x^i. \tag{5.12}$$

5.1.4 Determinação de A_2 e B_2

Dividindo-se o maior grau de A_1 por f e tomando-se o resto, tem-se $D_2 = D_{2_a} + D_{2_b} + D_{2_c}$, onde D_{2_a}, D_{2_b} e D_{2_c} referenciam as reduções dos três termos da Equação 5.11.

$$D_{2_a} = \sum_{i=0}^{a-2} d_{i+2m-a}x^i(x^a + x^b + x^c + 1). \tag{5.13}$$

Separando-se D_{2_a} nas parcelas a reduzir e já reduzidas, tem-se

$$A_{2_a} = \sum_{i=m}^{2a-2} d_{i+2m-2a}x^i$$

e

$$B_{2_a} = \sum_{i=a}^{m-1} d_{i+2m-2a}x^i + \sum_{i=b}^{a+b-2} d_{i+2m-a-b}x^i + \sum_{i=c}^{a+c-2} d_{i+2m-a-c}x^i + \sum_{i=0}^{a-2} d_{i+2m-a}x^i.$$

Substituindo-se $m = 2b + c$ e $a = b + c$, tem-se

$$A_{2_a} = \sum_{i=2b+c}^{2b+2c-2} d_{i+2b}x^i, \tag{5.14}$$

e

$$B_{2_a} = \sum_{i=b+c}^{2b+c-1} d_{i+2b}x^i + \sum_{i=b}^{2b+c-2} d_{i+2b+c}x^i + \sum_{i=c}^{b+2c-2} d_{i+3b}x^i + \sum_{i=0}^{b+c-2} d_{i+3b+c}x^i. \tag{5.15}$$

Procedendo a redução do segundo termo da Equação 5.11, tem-se

$$D_{2_b} = \sum_{i=a}^{a+b-2} d_{i+2m-a-b}x^i + \sum_{i=b}^{2b-2} d_{i+2m-2b}x^i + \sum_{i=c}^{b+c-2} d_{i+2m-b-c}x^i + \sum_{i=0}^{b-2} d_{i+2m-b}x^i. \tag{5.16}$$

Pode-se verificar que D_{2_b} já está reduzido. Assim:

$$A_{2_b} = 0,$$

e

$$B_{2_b} = \sum_{i=b+c}^{2b+c-2} d_{i+2b+c}x^i + \sum_{i=b}^{2b-2} d_{i+2b+2c}x^i + \sum_{i=c}^{b+c-2} d_{i+3b+c}x^i + \sum_{i=0}^{b-2} d_{i+3b+2c}x^i. \tag{5.17}$$

Procedendo a redução do terceiro termo da Equação 5.11, tem-se

$$D_{2_c} = \sum_{i=a}^{a+c-2} d_{i+2m-a-c}x^i + \sum_{i=b}^{b+c-2} d_{i+2m-b-c}x^i + \sum_{i=c}^{2c-2} d_{i+2m-2c}x^i + \sum_{i=0}^{c-2} d_{i+2m-c}x^i. \tag{5.18}$$

Novamente, é possível verificar que D_{2_c} já está reduzido. Assim:

$$A_{2_c} = 0, \tag{5.19}$$

e

$$B_{2_c} = \sum_{i=b+c}^{b+2c-2} d_{i+3b}x^i + \sum_{i=b}^{b+c-2} d_{i+3b+c}x^i + \sum_{i=c}^{2c-2} d_{i+4b}x^i + \sum_{i=0}^{c-2} d_{i+4b+c}x^i. \tag{5.20}$$

Assim A_2 é dado por

$$A_2 = A_{2_a} + A_{2_b} + A_{2_c} = \sum_{i=m}^{2a-2} d_{i+2m-2a}x^i, \tag{5.21}$$

e B_2 é dado por

$$B_2 = B_{2_a} + B_{2_b} + B_{2_c},$$

ou seja

$$\begin{aligned} B_2 = &\sum_{i=b+c}^{2b+c-1} d_{i+2b}x^i + \sum_{i=c}^{b+2c-2} d_{i+3b}x^i + \sum_{i=b+c}^{b+2c-2} d_{i+3b}x^i + \\ &\sum_{i=c}^{2c-2} d_{i+4b}x^i + \sum_{i=b}^{2b+c-2} d_{i+2b+c}x^i + \sum_{i=b+c}^{2b+c-2} d_{i+2b+c}x^i + \\ &\sum_{i=b}^{2b-2} d_{i+2b+2c}x^i + \sum_{i=0}^{b+c-2} d_{i+3b+c}x^i + \sum_{i=c}^{b+c-2} d_{i+3b+c}x^i + \\ &\sum_{i=b}^{b+c-2} d_{i+3b+c}x^i + \sum_{i=0}^{b-2} d_{i+3b+2c}x^i + \sum_{i=0}^{c-2} d_{i+4b+c}x^i. \end{aligned} \tag{5.22}$$

5.1.5 Determinação de A_3 e B_3

Dividindo-se o maior termo de A_2, isto é, a Equação 5.21 por f e tomando-se o resto, tem-se:

$$D_3 = \sum_{i=b+c}^{b+2c-2} d_{i+3b}x^i + \sum_{i=b}^{b+c-2} d_{i+3b+c}x^i + \sum_{i=c}^{2c-2} d_{i+4b}x^i + \sum_{i=0}^{c-2} d_{i+4b+c}x^i.$$

Pode-se mostrar que D_3 já está reduzido. Assim

$$A_3 = 0, \tag{5.23}$$

e

$$B_3 = \sum_{i=b+c}^{b+2c-2} d_{i+3b}x^i + \sum_{i=b}^{b+c-2} d_{i+3b+c}x^i + \sum_{i=c}^{2c-2} d_{i+4b}x^i + \sum_{i=0}^{c-2} d_{i+4b+c}x^i.$$

5.1.6 Determinação de D_{red} para $k_{b+c} = 2$

Só ocorre $k_{b+c} = 2$ quando $c = 1$. Neste caso, D_{red} da Equação 5.7 será dado por:

$$
\begin{aligned}
D_{red} = & \sum_{i=0}^{2b} d_i x^i + \sum_{i=b+1}^{2b} d_{i+b} x^i + \sum_{i=1}^{b} d_{i+2b} x^i + \\
& \sum_{i=1}^{b} d_{i+3b} x^i + \sum_{i=b}^{2b} d_{i+b+1} x^i + \sum_{i=0}^{b-1} d_{i+2b+1} x^i + \\
& \sum_{i=b+1}^{2b-1} d_{i+2b+1} x^i + \sum_{i=b}^{2b-2} d_{i+2b+2} x^i + d_{3b+1} + \\
& \sum_{i=0}^{b-2} d_{i+3b+2} x^i .
\end{aligned}
\tag{5.24}
$$

Uma análise cuidadosa da Equação 5.24, isto é, a verificação dos coeficientes repetidos nos intervalos, mostra que as operações a seguir são utilizadas mais de uma vez:

$$
T_1(j) = \sum_{i=0}^{b-2} (d_{i+2b+1} + d_{i+3b+2}) x^{i+j} \ , T_2(j) = d_{3b} x^j ,
$$

$$
T_3(j) = d_{3b+1} x^j , \ T_4(j) = \sum_{i=0}^{b-1} (d_{i+2b+1} + d_{i+3b+1}) x^{i+j} .
$$

Pode-se então reescrever a Equação 5.24 como

$$
\begin{aligned}
D_{red} = & B_0 + T_1(0) + T_1(b) + T_1(b+1) + T_2(b-1) + \\
& T_2(2b-1) + T_2(2b) + T_3(0) + T_3(2b) + T_4(1)
\end{aligned}
\tag{5.25}
$$

A Figura 21 ilustra de uma forma gráfica essas operações.
O número de operações de XOR é dado por

$$
N_{\oplus} = 3m - 2;
$$

pode-se verificar na figura que o atraso é de

$$
Delay = 3T_X .
$$

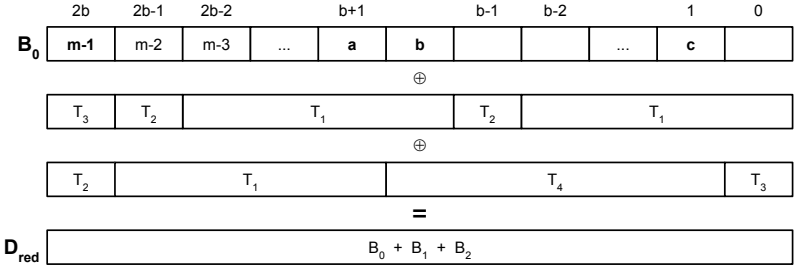

Figura 21 – Representação gráfica das operações de redução para $k_{b+c} = 2$.

5.1.6.1 Exemplo numérico para $k_{b+c} = 2$

Para um melhor entendimento das equações anteriores e da otimização, será demonstrado um exemplo ilustrado e numérico. Para isto, tem-se $f(x) = x^{13} + x^7 + x^6 + x + 1$. A Figura 22 é a redução gerada por f. Na figura, os elementos acima da linha amarela representam as equações A_0 e B_0 e abaixo representam os elementos A_1 e B_1. Percebe-se que a figura não apresenta ainda nenhuma otimização.

$2m-2$											m	$2b$ $m-1$	$2b-1$	$2b-2$			$b+1$ a	b b	$b-1$				c	0
24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
–	–	–	–	–	–	24	23	22	21	20	19	18	17	16	15	14	13	–	–	–	–	–	–	–
–	–	–	–	–	–	–	24	23	22	21	20	19	18	17	16	15	14	13	–	–	–	–	–	–
–	–	–	–	–	–	–	–	–	–	–	–	24	23	22	21	20	19	18	17	16	15	14	13	–
–	–	–	–	–	–	–	–	–	–	–	–	–	24	23	22	21	20	19	18	17	16	15	14	13
■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
–	–	–	–	–	–	–	–	–	–	–	–	24	23	22	21	20	19	–	–	–	–	–	–	–
–	–	–	–	–	–	–	–	–	–	–	–	–	24	23	22	21	20	–	–	–	–	–	–	–
–	–	–	–	–	–	–	–	–	–	–	–	–	24	23	22	21	20	19	–	–	–	–	–	–
–	–	–	–	–	–	–	–	–	–	–	–	–	–	24	23	22	21	20	–	–	–	–	–	–
–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	24	23	22	21	20	19	–
–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	24	23	22	21	20	–
–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	24	23	22	21	20	19
–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	24	23	22	21	20

Figura 22 – Redução modular gerada pelo pentanômio $f(x) = x^{13} + x^7 + x^6 + x + 1$.

A Figura 23 apresenta a retirada dos elementos repetidos. Entretanto, nesta imagem ainda é possível identificar redundâncias.

A Figura 24 é uma representação intermediária onde se consegue observar onde estão as principais redundâncias da família $x^{2b+c}+x^{b+c}+x^b+x^c+1$. Para o exemplo, $T_1(0) = \sum_{i=0}^{4}(d_{i+13} + d_{i+20})x^i$, e para os $T_1(6)$, $T_1(7)$ e os valores de $T_3(0) = d_{19}x^0$ e $T_3(12) = d_{19}x^12$.

Por fim, a Figura 25 apresenta todas as equações representadas

$2b$	$2b-1$				$b+1$	b	$b-1$						
$m-1$					a	b					c	0	
12	11	10	9	8	7	6	5	4	3	2	1	0	
18	17	16	15	14	13	–	–	–	–	–	–	–	–
–	24	23	22	21	20	–	–	–	–	–	–	–	–
19	18	17	16	15	14	13	–	–	–	–	–	–	–
–	–	24	23	22	21	20	–	–	–	–	–	–	–
–	–	–	–	–	–	–	18	17	16	15	14	13	–
–	–	–	–	–	–	–	–	18	17	16	15	14	13
–	–	–	–	–	–	–	–	–	24	23	22	21	20
–	–	–	–	–	–	–	24	23	22	21	20	19	–
–	–	–	–	–	–	–	–	–	–	–	–	–	19

Figura 23 – Representação gráfica das operações de redução sem otimizações para $k_{b+c} = 2$ utilizando $f(x) = x^{13} + x^7 + x^6 + x + 1$.

$2b$	$2b-1$				$b+1$	b	$b-1$						
$m-1$					a	b					c	0	
12	11	10	9	8	7	6	5	4	3	2	1	0	
18	17	16	15	14	13	–	–	–	–	–	–	–	–
–	24	23	22	21	20	–	–	–	–	–	–	–	–
19	18	17	16	15	14	13	–	–	–	–	–	–	–
–	–	24	23	22	21	20	–	–	–	–	–	–	–
–	–	–	–	–	–	–	18	17	16	15	14	13	–
–	–	–	–	–	–	–	–	18	17	16	15	14	13
–	–	–	–	–	–	–	–	–	24	23	22	21	20
–	–	–	–	–	–	–	24	23	22	21	20	19	–
–	–	–	–	–	–	–	–	–	–	–	–	–	19

Figura 24 – Representação gráfica das operações de redução com operações $T_1(j)$ e $T_3(j)$ marcadas utilizando $f(x) = x^{13}+x^7+x^6+x+1$.

em $T_1(j)$, $T_2(j)$, $T_3(j)$ e $T_4(j)$. A parte em azul representa $T_1(j)$, onde se tem $T_1(j)$ iniciando em 0, 6 e 7. A verde $T_2(j)$, sendo apenas um termo, inicia-se em 5, 11 e 12. A rosa $T_3(j)$, também sendo apenas um termo, inicia-se em 0 e 12. Por último, a vermelha $T_4(j)$, que é o conjunto de termos que se inicia em 1 e termina em 6.

5.1.6.2 Algoritmo de D_{red} para $k_{b+c} = 2$ para $f(x) = x^{2b+1} + x^{b+1} + x^b + x + 1$

Com base nas equações apresentadas anteriormente, é possível gerar o Algoritmo 5. A entrada do algoritmo é um vetor de bits de

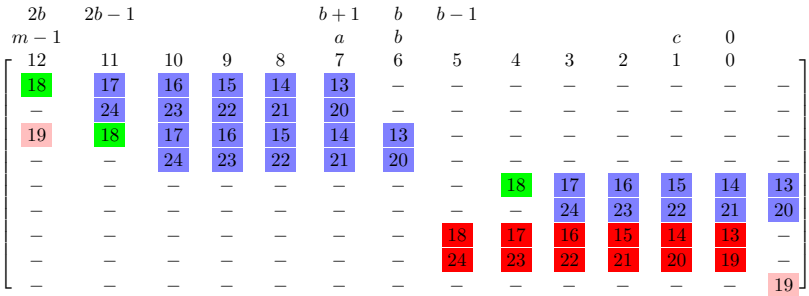

Figura 25 – Representação gráfica das operações de redução com operações $T_1(j)$, $T_2(j)$, $T_3(j)$ e $T_4(j)$ realçadas utilizando $f(x) = x^{13} + x^7 + x^6 + x + 1$.

tamanho $4b + 1$, o qual representa um polinômio multiplicado de grau até $2m - 2$.

Teorema 9 *O Algoritmo 5 apresenta corretamente a redução modular polinomial de grau até* $2m - 2$ *sobre* $\mathbb{F}_2$ *por* $f(x) = x^{2b+1} + x^{b+1} + x^b + x + 1$, *envolvendo* $N_{\oplus} = 3m - 2 = 6b + 1$ *operações XORs e atraso de* $3T_X$.

A demonstração do teorema é dada nos somatórios quando $k_{b+c} = 2$. Isto é, a Equação 5.25, que apresenta os somatórios e variáveis temporárias envolvidas.

5.1.7 Determinação de D_{red} para $k_{b+c} = 3$

Utilizando-se a Equação 5.8, isto é:

$$D_{red} = B_0 + B_1 + B_2 + B_3$$

desta forma, pode-se reescrever D_{red} como

Algoritmo 5: Algoritmo de D_{red} para $k_{b+c} = 2$

Entrada: $D_0 = d[4b \dots 0]$ vetor de bits de tamanho $4b + 1$
Saída: D_{red} vetor de bits com tamanho de $m - 1$

```
para i ← 0 até b − 2 faça
    T1[i] ← d[i + 2b + 1] ⊕ d[i + 3b + 2];
para i ← 0 até b − 1 faça
    T4[i] ← d[i + 2b + 1] ⊕ d[i + 3b + 1];
Dred[0] ← d[0] ⊕ T1[0] ⊕ d[3b + 1];
para i ← 1 até b − 2 faça
    Dred[i] ← d[i] ⊕ T1[i] ⊕ T4[i − 1];
Dred[b − 1] ← d[b − 1] ⊕ d[3b] ⊕ T4[b − 2];
Dred[b] ← d[b] ⊕ T1[0] ⊕ T4[b − 1];
para i ← b + 1 até 2b − 2 faça
    Dred[i] ← d[i] ⊕ T1[i − b] ⊕ T1[i − b − 1];
Dred[2b − 1] ← d[2b − 1] ⊕ d[3b] ⊕ T1[b − 2];
Dred[2b] ← d[2b] ⊕ d[3b + 1] ⊕ d[3b];
retorna Dred;
```

$$
\begin{aligned}
D_{red} = & \sum_{i=0}^{2b+c-1} d_i x^i + \sum_{i=b+c}^{2b+c-1} d_{i+b} x^i + \sum_{i=c}^{b+c-1} d_{i+2b} x^i + \\
& \sum_{i=c}^{b+2c-2} d_{i+3b} x^i + \sum_{i=b}^{2b+c-1} d_{i+b+c} x^i + \sum_{i=0}^{b-1} d_{i+2b+c} x^i + \\
& \sum_{i=b+c}^{2b+c-2} d_{i+2b+c} x^i + \sum_{i=b}^{2b-2} d_{i+2b+2c} x^i + \sum_{i=0}^{c-1} d_{i+3b+c} x^i + \\
& \sum_{i=0}^{b-2} d_{i+3b+2c} x^i .
\end{aligned}
\tag{5.26}
$$

Uma análise cuidadosa da Equação 5.26 mostra as seguintes operações são utilizadas mais de uma vez:

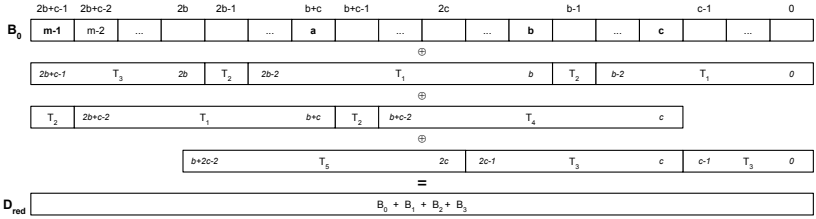

Figura 26 – Representação gráfica das operações de redução para $k_{b+c} = 3$.

$$T_1(j) = \sum_{i=0}^{b-2}(d_{i+2b+c} + d_{i+3b+2c})x^{i+j},\ T_2(j) = d_{3b+c-1}x^j,$$
$$T_3(j) = \sum_{i=0}^{c-1} d_{i+3b+c}x^{i+j},\ T_4(j) = \sum_{i=0}^{b-2} d_{i+2b+c}x^{i+j},$$
$$T_5(j) = \sum_{i=0}^{b-2} d_{i+3b+2c}x^{i+j}.$$

Pode-se então reescrever a Equação 5.26 como

$$\begin{aligned} D_{red} =& B_0 + T_1(0) + T_1(b) + T_1(b+c) + T_2(b-1) + \\ & T_2(b+c-1) + T_2(2b-1) + T_2(2b+c-1) + \\ & T_3(0) + T_3(c) + T_3(2b) + T_4(c) + T_5(2c). \end{aligned} \tag{5.27}$$

A Figura 26 ilustra de forma gráfica essas operações.

Após as remoções de redundâncias e termos repetidos, o número de XORs é dado por

$$N_{\oplus} = 6b + 3c - 2 = 3m - 2.$$

O atraso é dado por

$$Delay = 3T_X.$$

5.1.7.1 Algoritmo de D_{red} para $k_{b+c} = 3$ onde $f(x) = x^{2b+c} + x^{b+c} + x^b + x^c + 1$

Com base nas equações apresentadas anteriormente, é possível gerar o Algoritmo 6. A entrada do algoritmo é um vetor de bits de

tamanho $2b + c$, o qual representa um polinômio multiplicado de grau até $2m - 2$.

Algoritmo 6: Algoritmo de D_{red} para $k_{b+c} = 3$ onde $f(x) = x^{2b+c} + x^{b+c} + x^b + x^c + 1$

Entrada: $D_0 = d[2b + c - 1 \ldots 0]$ vetor de bits de tamanho $2b + c$ que representa o polinômio a ser reduzido
Saída: D_{red} vetor de bits de tamanho $m - 1$ com o polinômio de entrada reduzido

```
1  para i ← 0 até b − 2 faça
2      T1[i] ← d[i + 2b + 1] ⊕ d[i + 3b + 2c];
3  para i ← 0 até c − 1 faça
4      Dred[i] ← d[i] ⊕ T1[i];
5  para i ← c até b − 2 faça
6      Dred[i] ← d[i] ⊕ T1[i] ⊕ d[i + 2b];
7  Dred[b − 1] ← d[b − 1] ⊕ d[3b + c − 1] ⊕ d[3b − 1];
8  para i ← b até b + c − 2 faça
9      Dred[i] ← d[i] ⊕ T1[i − b] ⊕ d[i + 2b];
10 Dred[b + c − 1] ← d[b + c − 1] ⊕ d[3b + c − 1] ⊕ T1[c − 1];
11 para i ← b + c até 2b − 2 faça
12     Dred[i] ← d[i] ⊕ T1[i − b] ⊕ T1[i − b − c];
13 Dred[2b − 1] ← d[2b − 1] ⊕ d[3b + c − 1] ⊕ T1[b − c − 1];
14 para i ← 2b até 2b + c − 2 faça
15     Dred[i] ← d[i] ⊕ T1[i − b − c] ⊕ d[i + b + c];
16 Dred[2b + c − 1] ← d[2b + c − 1] ⊕ d[3b + c − 1] ⊕ d[3b − 1];
17 para i ← 0 até 2b − 2 faça
18     Dred[i] ← Dred[i] ⊕ d[i + 3b + c];
19 para i ← c até b + 2c − 2 faça
20     Dred[i] ← Dred[i] ⊕ d[i + 3b];
21 return Dred;
```

Teorema 10 *O Algoritmo 6 apresenta corretamente a redução modular polinomial de um polinômio de grau até* $2m - 2$ *sobre* $\mathbb{F}_2$ *por* $f(x) = x^{2b+c} + x^{b+c} + x^b + x^c + 1$, *utilizando* $N_{\oplus} = 3m - 2 = 6b + 3c - 2$ *operações XORs e atraso de* $3T_X$.

A demonstração do teorema é dada nos somatórios quando $k_{b+c} = 3$. Isto é, a Equação 5.27, que apresenta os somatórios e variáveis temporárias envolvidas.

5.1.8 Caso especial $b = 2c$

Quando o polinômio for do formato $f(x) = x^{5c} + x^{3c} + x^{2c} + x^c + 1$, o número de *XORs* será menor que os anteriores, conforme demonstrado a seguir.

$$D_{red} = \sum_{i=0}^{5c-1} d_i x^i + \sum_{i=3c}^{5c-1} d_{i+2c} x^i + \sum_{i=c}^{3c-1} d_{i+4c} x^i + \sum_{i=c}^{4c-2} d_{i+6c} x^i + \sum_{i=2c}^{5c-1} d_{i+3c} x^i + \sum_{i=0}^{2c-1} d_{i+5c} x^i + \sum_{i=3c}^{5c-2} d_{i+5c} x^i + \sum_{i=2c}^{4c-2} d_{i+6c} x^i + \sum_{i=0}^{c-1} d_{i+7c} x^i + \sum_{i=0}^{2c-2} d_{i+8c} x^i, \tag{5.28}$$

$$\begin{aligned} T_1(j) &= \sum_{i=c}^{2c-2} (d_{i+5c} + d_{i+4c}) x^{i+j}, \\ T_2(j) &= \sum_{i=c}^{2c-2} (d_{i+8c} + d_{i+6c}) x^{i+j}, \\ T_3(j) &= d_{8c-1} x^j, \\ T_4(j) &= \sum_{i=0}^{c-1} d_{i+8c} x^{i+j}, \\ T_5(j) &= \sum_{i=0}^{c-1} d_{i+5c} x^{i+j}, \\ T_6(j) &= \sum_{i=0}^{c-2} d_{i+7c} x^{i+j}, \\ T_7(j) &= \sum_{i=4c}^{5c-1} d_{i+2c} x^{i+j}. \end{aligned} \tag{5.29}$$

A Figura 27 ilustra de forma gráfica essas operações.
Após as remoções de redundâncias e termos repetidos, o número

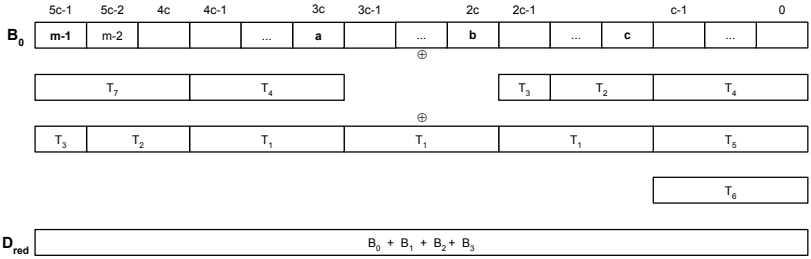

Figura 27 – Representação gráfica das operações de redução para $k_{b+c} = 3$ quando $b = 2c$.

de *XORs* é dado por:

$$N_{\oplus} = \frac{12}{5}m - 1 = 12c - 1.$$

O número de operações *XORs* está perto de $2.4m$, o que representa cerca de 20% a menos que os outros pentanômios da mesma família. Pentanômios irredutíveis deste tipo são raros, mas eles existem para os graus 5, 155 e 4805. O grau 155 é utilizado na prática, conforme apresentado por Agnew, Mullin e Vanstone (1993).

5.1.8.1 Caso especial $b = 2c = 2$

Este caso é especial pois ocorre quando $b = 2c$ e $c = 1$, isto é, o $k_{b+c} = 2$ e existe apenas um pentanômio a considerar: $x^5 + x^3 + x^2 + x + 1$. Este mesmo pentanômio é apresentado no trabalho de Rodríguez-Henríquez e Koç (2003) com um número de *XORs* é de 18.

$$\begin{aligned} D_{red} =& d_0 + d_5 + d_7 + d_8 + (d_1 + t + d_7)x + \\ & (d_2 + t)x^2 + (d_3 + t + d_8)x^3 + (d_4 + d_6 + d_7)x^4, \end{aligned} \tag{5.30}$$

onde $t = d_5 + d_6$. Neste caso obtemos $N_{\oplus} = \frac{12}{5}m - 1 = 11$ *XORs*.

5.2 COMPARAÇÕES COM FAMÍLIAS EXISTENTES

A comparação entre os tipos de pentanômios conhecidos e a nova família apresentada neste trabalho está detalhada nas Tabelas 18 e 19.

Algoritmo 7: Algoritmo de D_{red} para $f(x) = x^{5c} + x^{3c} + x^{2c} + x^c + 1$.

Entrada: $D_0 = d[5c-1 \ldots 0]$ bits vector of length $5c$
Saída: D_{red}

for $i \leftarrow 0$ **até** $c-2$ **do**
  $T_1[i] \leftarrow d[i+6c] \oplus d[i+5c]$;
para $i \leftarrow 0$ **até** $c-2$ **faça**
  $T_2[i] \leftarrow d[i+9c] \oplus d[i+7c]$;
para $i \leftarrow 0$ **até** $c-2$ **faça**
  $D_{red}[i] \leftarrow d[i] \oplus d[i+8c] \oplus d[i+5c] \oplus d[i+7c]$;
$D_{red}[c-1] \leftarrow d[c-1] \oplus d[9c-1] \oplus d[6c-1]$;
para $i \leftarrow c$ **até** $2c-2$ **faça**
  $D_{red}[i] \leftarrow d[i] \oplus T_1[i-c] \oplus T_2[i-c]$;
$D_{red}[2c-1] \leftarrow d[2c-1] \oplus d[8c-1] \oplus T_1[c-1]$;
para $i \leftarrow 2c$ **até** $3c-1$ **faça**
  $D_{red}[i] \leftarrow d[i] \oplus T_1[i-2c]$;
para $i \leftarrow 3c$ **até** $4c-1$ **faça**
  $D_{red}[i] \leftarrow d[i] \oplus T_1[i-3c] \oplus d[i+5c]$;
para $i \leftarrow 4c$ **até** $5c-2$ **faça**
  $D_{red}[i] \leftarrow d[i] \oplus T_2[i-4c] \oplus d[i+2c]$;
$D_{red}[5c-1] \leftarrow d[5c-1] \oplus d[8c-1] \oplus d[7c-1]$;
retorna D_{red};

Para uma comparação mais justa, foi adicionado o valor da multiplicação que é $(m-1)^2$. Não foi incluído o valor dos atrasos na portas *XORs*, uma vez que todos apresentam ou estão muito próximos de $3T_X$. Ainda nesta tabela, percebe-se que o pentanômio proposto apresenta uma excelente performance.

A Tabela 20 apresenta o número de pentanômios irredutíveis por família até o grau 1024. Existem muitos pentanômios para as famílias do tipo C.2, C.3 e C.4, mas elas apresentam uma complexidade alta. As famílias do tipo C.5, C.6 e C.7 apresentam, combinadas, um total de 585 pentanômios e a C.9, proposta nesta dissertação, apresenta 728. Também é incluído na tabela, o número de pentanômios para os graus recomendados pelo NIST. É importante destacar que a família proposta apresenta um número menor de *XORs*, uma vez que as famílias que seriam melhores, que seriam C.5, C.6, C.7 e C.8, não apresentam pentanômios com grau 163, 283 e 571.

Por fim, nas Tabelas 21, 22 e 23 são listados os menores valores de *XORs* para cada família. Percebe-se que a família apresentada nesta

Tabela 18 – Formato das famílias da literatura e a apresentada neste trabalho.

Tipo	Famílias de Pentanômios	Autor
C.1	$x^m + x^a + x^b + x^c + 1$, $1 \leq c < b < a \leq \frac{m}{2}$	(REYHANI-MASOLEH; HASAN, 2004)
C.2	$x^m + x^{b+1} + x^b + x^{b-1} + 1$, $1 < b \leq \frac{m}{2} - 1$	(REYHANI-MASOLEH; HASAN, 2004; FAN; HASAN, 2006)
C.3	$x^m + x^{b+1} + x^b + x + 1$, $1 < b \leq \frac{m}{2} - 1$	(REYHANI-MASOLEH; HASAN, 2004; FAN; HASAN, 2006)
C.4	$x^m + x^{m-c} + x^b + x^c + 1$, $b = 2c$ or $b = m - 2c$	(CILARDO, 2013)
C.5	$x^m + x^{m-s} + x^{m-2s} + x^{m-3s} + 1$, $\frac{m-1}{4} \leq s \leq \frac{m-1}{3}$	(REYHANI-MASOLEH; HASAN, 2004)
C.6	$x^m + x^{m-s} + x^{m-2s} + x^{m-3s} + 1$, $\frac{m-1}{5} \leq s < \frac{m-1}{4}$	(REYHANI-MASOLEH; HASAN, 2004)
C.7	$x^m + x^{m-s} + x^{m-2s} + x^{m-3s} + 1$, $\frac{m-1}{8} \leq s < \frac{m-1}{5}$	(REYHANI-MASOLEH; HASAN, 2004)
C.8	$x^{m=4c} + x^{3c} + x^{2c} + x^c + 1$, $c = 5^i$ and $i \geq 0$	(HASAN; WANG; BHARGAVA, 1992)
C.9	$x^{m=2b+c} + x^{b+c} + x^b + x^c + 1$, $c \geq 1$, $b \neq 2c$	Família proposta
C.10	$x^{m=5c} + x^{3c} + x^{2c} + x^c + 1$, $c \geq 1$	Família proposta

Tabela 19 – Comparação entre famílias da literatura e a apresentada neste trabalho.

Tipo	No. de XORs
C.1	$m^2 + 2m - 3$
C.2	$m^2 + m + b - 3$
C.3	$m^2 + m$
C.4	$m^2 + m + 5c - 2$
C.5	$m^2 + m - s - 1$
C.6	$m^2 + 2m - 5s - 2$
C.7	$m^2 + m - 2$
C.8	$m^2 - \frac{1}{4}m$
C.9	$m^2 + m - 1$
C.10	$m^2 + \frac{2}{5}m$

Tabela 20 – Número de pentanômios irredutíveis com grau entre 5 e 1024 por família e número de pentanômios irredutíveis de graus do NIST.

Tipo	No. de Irredutíveis	163	283	571
C.2	1,676	3	2	2
C.3	2,025	1	2	0
C.4	1,400	2	0	6
C.5	211	0	0	0
C.6	133	0	0	0
C.7	181	0	0	0
C.8	4	0	0	0
C.9	726	2	2	1
C.10	2	0	0	0

Tabela 21 – Melhores pentanômios para grau 163.

Pentanômio	Tipo	#XORs
$x^{163} + x^{68} + x^{67} + x^{66} + 1$	C.2	$27,009$
$x^{163} + x^{60} + x^{59} + x + 1$	C.3	$26,732$
$x^{163} + x^{117} + x^{71} + x^{46} + 1$	C.4	$26,960$
$x^{163} + x^{7} + x^{6} + x^{3} + 1$	NIST	$26,815$
$x^{163} + x^{100} + x^{63} + x^{37} + 1$	C.9	**26,731**

Tabela 22 – Melhores pentanômios para grau 283.

Pentanômio	Tipo	#XORs
$x^{283} + x^{25} + x^{24} + x^{23} + 1$	C.2	$80,580$
$x^{283} + x^{60} + x^{59} + x + 1$	C.3	$80,372$
$x^{283} + x^{12} + x^{7} + x^{5} + 1$	NIST	$80,386$
$x^{283} + x^{172} + x^{111} + x^{61} + 1$	C.9	**80,371**

dissertação é a que apresenta a menor quantidade de *XORs* para os graus de interesse 163, 283 e 571. Portanto, esta é a melhor família encontrada na literatura para esses graus.

No Apêndice A são apresentados em uma tabela todos os pentanômios irredutíveis, até grau 1024, da família proposta neste capítulo. Na tabela, são apresentados os coeficientes do pentanômio, juntamente com o número de *XORs* para a sua computação.

5.3 CONCLUSÃO

Ao longo deste capítulo foi demonstrado a redução modular para a família $f(x) = x^{2b+c} + x^{b+c} + x^{b} + x^{c} + 1$. Demonstrou-se como é efetuada a redução de um elemento em $\mathbb{F}_{2^m}$, utilizando f como polinômio irredutível.

Durante o processo de redução, apresentou-se equações para a redução. Durante o equacionamento da redução foram detectadas redundâncias de operações que não eram necessárias. Desta forma, ocorre

Tabela 23 – Melhores pentanômios para grau 571.

Pentanômio	Tipo	#XORs
$x^{571} + x^{105} + x^{104} + x^{103} + 1$	C.2	$327,204$
$x^{571} + x^{549} + x^{44} + x^{22} + 1$	C.4	$326,720$
$x^{571} + x^{10} + x^{5} + x^{2} + 1$	NIST	$326,903$
$x^{571} + x^{353} + x^{218} + x^{135} + 1$	C.9	**326,611**

a diminuição da complexidade do processo de redução quando é utilizado f.

Quando esta família foi comparada com as demais existentes na literatura, a família proposta apresentou boas propriedades na redução modular polinomial. Desta forma, apresentou-se os melhores resultados para os graus 163, 283 e 571 recomendados pelo NIST.

6 CONCLUSÃO

Este trabalho apresentou dois resultados importantes: um algoritmo para a contagem de operações *XORs* dado um polinômio irredutível e a família de pentanômios $x^{2b+c} + x^{b+c} + x^b + x^c + 1$ com $b > c$, que apresenta ótimos resultados em termos de espaço e tempo. Estes resultados tem como base a teoria e toda uma pesquisa efetuada sobre corpos finitos e a sua aritmética, mais precisamente em corpos binários. Conforme demonstrado, a família proposta neste trabalho apresenta uma melhora significativa na literatura conhecida. Desta forma, o modelo matemático apresentado e demais resultados do mesmo se tornaram um artigo submetido para a revista *IEEE Computers on Transactions* (BANEGAS; CUSTÓDIO; PANARIO, 2015).

Pode-se ressaltar a incompletude de famílias não presentes na literatura para pentanômios com o formato $x^m + x^a + x^b + x^c + 1$, com o $a > \frac{m}{2}$. Em parte, esta falta está relacionada á complexidade de análise, conforme foi demonstrado neste trabalho, e também pela padronização de pentanômios por entidades, como, por exemplo, o NIST.

Felizmente, conforme a pesquisa efetuada apresentou, a necessidade de melhorar as operações matemáticas em corpos binários vem crescendo e, com isto, o questionamento da padronização também. Neste ponto, vale destacar o diferencial dos procedimentos metodológicos deste trabalho, o qual fez um levantamento das famílias de pentanômios, métodos de redução modular e a heurística de algoritmos gulosos. Desta forma, foi possível a geração do algoritmo que então possibilitou a descoberta de novas famílias de polinômios. A família $x^{b+2c} + x^{b+c} + x^b + x^c + 1$ com $b > c$, por exemplo, parece ser experimentalmente excelente. Como estudo futuro, sugere-se uma análise formal desta família.

6.1 TRABALHOS FUTUROS

Dado que nem todos os resultados do algoritmo foram analisados, existe uma lacuna disponível a ser trabalhada. Isto é, foi analisada formalmente apenas uma família de pentanômios. A família de pentanômios que foi apresentada nos resultados do Conta-XOR, $x^{b+2c} + x^{b+c} + x^b + x^c + 1$ com $b > c$, necessita de um trabalho formal, conforme o apresentado no Capítulo 5. Contudo, existem outras famílias de pentanômios a serem estudados para verificar a complexidade, uma vez que o algoritmo Conta-XOR indicou que são famílias

com uma performance excelente.

Neste âmbito de bits e implementações, outro ponto de extensão deste trabalho é o desenvolvimento da análise da complexidade da aritmética modular em palavras, conforme está descrito no livro de Hankerson, Menezes e Vanstone (2003) e no trabalho de Scott (2007). Além de operações, como a aritmética modular, em corpos finitos, a busca por polinômios ótimos abrange diversos casos, como Banegas e Custódio (2015) apresentam na geração de números pseudoaleatórios.

REFERÊNCIAS

AGNEW, G.; MULLIN, R.; VANSTONE, S. An implementation of elliptic curve cryptosystems over $\mathbb{F}_{2^{155}}$. **Selected Areas in Communications, IEEE Journal on**, v. 11, n. 5, p. 804–813, Jun 1993. ISSN 0733-8716.

BANEGAS, G.; CUSTÓDIO, R.; PANARIO, D. A new class of irreducible pentanomials for efficient field arithmetic in $GF(2^m)$. **Computers, IEEE Transactions on**, Sep 2015. Submetido à publicação em 5 de setembro de 2015.

BANEGAS, G. S.; CUSTÓDIO, R. F. An efficient keystream for cryptographic applications. **International Journal of Computer Applications in Technology**, 2015.

BARNARD, S. **Higher Algebra**. [S.l.]: Read Books, 2008. ISBN 9781443730860.

BROWN, M. et al. Software implementation of the nist elliptic curves over prime fields. In: **TOPICS IN CRYPTOLOGY â“ CT-RSA 2001, VOLUME 2020 OF LNCS**. [S.l.]: Springer, 2001. p. 250–265.

CILARDO, A. Fast parallel $GF(2^m)$ polynomial multiplication for all degrees. **Computers, IEEE Transactions on**, v. 62, n. 5, p. 929–943, May 2013. ISSN 0018-9340.

COHEN, H. et al. **Handbook of Elliptic and Hyperelliptic Curve Cryptography, Second Edition**. 2nd. ed. [S.l.]: Chapman & Hall/CRC, 2012. ISBN 1439840008, 9781439840009.

COLBOURN, C. J.; DINITZ, J. H. **Handbook of Combinatorial Designs**. [S.l.]: CRC press, 2006.

DESCHAMPS, J. P.; IMANA, J. L.; SUTTER, G. D. **Hardware Implementation of Finite-Field Arithmetic**. New York, NY, USA: The McGraw-Hill Companies, Inc., 2009. ISBN 9780071545815.

FAN, H.; HASAN, M. A. Fast bit parallel-shifted polynomial basis multipliers in $GF(2^n)$. **Circuits and Systems I: Regular Papers, IEEE Transactions on**, IEEE, v. 53, n. 12, p. 2606–2615, 2006.

FAN, H.; HASAN, M. A. A survey of some recent bit-parallel $GF(2^n)$ multipliers. **Finite Fields and Their Applications**, Elsevier, v. 32, p. 5–43, 2015.

GOLOMB, S. W. **Shift Register Sequences**. Laguna Hills, CA, USA: Aegean Park Press, 1981. ISBN 0894120484.

GOLOMB, S. W.; GONG, G. **Signal Design for Good Correlation: For Wireless Communication, Cryptography, and Radar**. New York, NY, USA: Cambridge University Press, 2004. ISBN 0521821045.

HANKERSON, D.; HERNANDEZ, J. L.; MENEZES, A. Software implementation of elliptic curve cryptography over binary fields. In: **Cryptographic Hardware and Embedded Systems â" CHES 2000**. [S.l.]: Springer Berlin Heidelberg, 2000, (Lecture Notes in Computer Science, v. 1965). p. 1–24. ISBN 978-3-540-41455-1.

HANKERSON, D.; MENEZES, A. J.; VANSTONE, S. **Guide to Elliptic Curve Cryptography**. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2003. ISBN 038795273X.

HASAN, M.; WANG, M.; BHARGAVA, V. Modular construction of low complexity parallel multipliers for a class of finite fields $GF(2^m)$. **Computers, IEEE Transactions on**, v. 41, n. 8, p. 962–971, Aug 1992. ISSN 0018-9340.

KAPLAN, B.; DUCHON, D. Combining qualitative and quantitative methods information systems research: A case study. **Manage. Inf. Syst. Q.**, Society for Information Management and The Management Information Systems Research Center, Minneapolis, MN, USA, v. 12, n. 4, p. 571–586, dez. 1988. ISSN 0276-7783.

LIDL, R.; NIEDERREITER, H. **Introduction to Finite Fields and Their Applications**. New York, NY, USA: Cambridge University Press, 1986. ISBN 0-521-30706-6.

MASUDA, A.; PANARIO, D. **Tópicos de corpos finitos com aplicações em criptografia e teoria de códigos**. Rio de Janeiro, RJ, BR: IMPA, 2007.

MATH, S. **Sage: Open Source Mathematics Software**. November 2013.

MULLEN, G. L.; PANARIO, D. **Handbook of Finite Fields**. [S.l.]: CRC Press, 2013.

NIST. **NIST Special Publication 800-53 Revision 3 Recommended Security Controls for Federal Information Systems and Organizations**. Paramount, CA: CreateSpace, 2012. ISBN 1470100363, 9781470100360.

REYHANI-MASOLEH, A.; HASAN, M. Low complexity bit parallel architectures for polynomial basis multiplication over $GF(2^m)$. **Computers, IEEE Transactions on**, v. 53, n. 8, p. 945–959, Aug 2004. ISSN 0018-9340.

RODRÍGUEZ-HENRÍQUEZ, F.; KOÇ, C. Parallel multipliers based on special irreducible pentanomials. **Computers, IEEE Transactions on**, v. 52, n. 12, p. 1535–1542, Dec 2003. ISSN 0018-9340.

SCOTT, M. Optimal irreducible polynomials for $GF(2^m)$ arithmetic. **IACR Cryptology ePrint Archive**, v. 2007, p. 192, 2007.

SUNAR, B.; KOÇ, C. K. Mastrovito multiplier for all trinomials. **Computers, IEEE Transactions on**, v. 48, n. 5, p. 522–527, May 1999. ISSN 0018-9340.

SWAN, R. G. Factorization of polynomials over finite fields. **Pacific Journal of Mathematics**, Pacific Journal of Mathematics, A Non-profit Corporation, v. 12, n. 3, p. 1099–1106, 1962.

VIEIRA, C. **Proposta de um modelo de implantação de tecnologias de informação e comunicação para prestadores de serviços logísticos**. 181 p. Dissertaï¿½ï¿½o (Mestrado) — Programa de Pós-Graduação em Engenharia de Produção - Universidade Federal de Santa Catarina, 2012.

WAZLAWICK, R. S. Capítulo 5 - escrita da monografia. In: WAZLAWICK, R. S. (Ed.). **Metodologia De Pesquisa Para Ciência Da Computação**. [S.l.]: Elsevier Editora Ltda., 2009. p. 97 – 121. ISBN 978-85-352-3522-7.

WU, H. Bit-parallel finite field multiplier and squarer using polynomial basis. **Computers, IEEE Transactions on**, IEEE, v. 51, n. 7, p. 750–758, 2002.

XIONG, X.; FAN, H. $GF(2^n)$ bit-parallel squarer using generalised polynomial basis for new class of irreducible pentanomials. **Electronics Letters**, v. 50, n. 9, p. 655–657, April 2014. ISSN 0013-5194.

APÊNDICE A – Pentanômios Irredutíveis da forma $x^{2b+c} + x^{b+c} + x^b + x^c + 1$

Este apêndice contém a lista de todos os pentanômios irredutíveis de grau até **1000**, para a família $x^{2b+c} + x^{b+c} + x^b + x^c + 1$.

Tabela 24: Tabela com os pentanômios irredutvéis de grau até **1000**, para a família $x^{2b+c} + x^{b+c} + x^b + x^c + 1$.

m,a,b, c, $N_\oplus$	m,a,b, c, $N_\oplus$
5, 3, 2, 1, 43	8, 5, 3, 2, 120
11, 6, 5, 1, 231	13, 7, 6, 1, 325
16, 9, 7, 2, 496	23, 14, 9, 5, 1035
25, 13, 12, 1, 1225	25, 14, 11, 3, 1225
29, 19, 10, 9, 1653	30, 17, 13, 4, 1770
37, 23, 14, 9, 2701	40, 23, 17, 6, 3160
45, 28, 17, 11, 4005	46, 29, 17, 12, 4186
47, 31, 16, 15, 4371	48, 27, 21, 6, 4560
52, 27, 25, 2, 5356	53, 32, 21, 11, 5565
59, 32, 27, 5, 6903	60, 31, 29, 2, 7140
60, 37, 23, 14, 7140	62, 37, 25, 12, 7626
65, 37, 28, 9, 8385	66, 35, 31, 4, 8646
71, 43, 28, 15, 10011	72, 45, 27, 18, 10296
73, 42, 31, 11, 10585	78, 49, 29, 20, 12090
81, 50, 31, 19, 13041	83, 45, 38, 7, 13695
85, 47, 38, 9, 14365	86, 49, 37, 12, 14706
93, 59, 34, 25, 17205	94, 49, 45, 4, 17578
96, 49, 47, 2, 18336	99, 61, 38, 23, 19503
103, 63, 40, 23, 21115	107, 61, 46, 15, 22791
108, 63, 45, 18, 23220	108, 67, 41, 26, 23220
110, 69, 41, 28, 24090	111, 59, 52, 7, 24531
114, 67, 47, 20, 25878	116, 73, 43, 30, 26796
117, 76, 41, 35, 27261	118, 73, 45, 28, 27730
120, 75, 45, 30, 28680	123, 77, 46, 31, 30135
130, 67, 63, 4, 33670	131, 69, 62, 7, 34191
136, 85, 51, 34, 36856	141, 91, 50, 41, 39621
145, 77, 68, 9, 41905	145, 82, 63, 19, 41905
146, 79, 67, 12, 42486	149, 80, 69, 11, 44253
153, 97, 56, 41, 46665	155, 92, 63, 29, 47895
155, 96, 59, 37, 47895	156, 85, 71, 14, 48516
158, 97, 61, 36, 49770	159, 82, 77, 5, 50403
164, 97, 67, 30, 53628	164, 99, 65, 34, 53628
176, 89, 87, 2, 61776	180, 93, 87, 6, 64620
180, 111, 69, 42, 64620	188, 117, 71, 46, 70500
190, 109, 81, 28, 72010	190, 113, 77, 36, 72010
194, 111, 83, 28, 75078	200, 105, 95, 10, 79800
202, 107, 95, 12, 81406	205, 107, 98, 9, 83845
205, 124, 81, 43, 83845	205, 128, 77, 51, 83845
210, 119, 91, 28, 87990	211, 124, 87, 37, 88831
213, 139, 74, 65, 90525	215, 111, 104, 7, 92235
219, 140, 79, 61, 95703	220, 119, 101, 18, 96580
225, 113, 112, 1, 101025	227, 137, 90, 47, 102831
228, 147, 81, 66, 103740	229, 124, 105, 19, 104653
233, 154, 79, 75, 108345	235, 133, 102, 31, 110215
236, 139, 97, 42, 111156	237, 143, 94, 49, 112101
240, 121, 119, 2, 114960	243, 133, 110, 23, 117855
244, 139, 105, 34, 118828	244, 145, 99, 46, 118828
251, 133, 118, 15, 125751	251, 148, 103, 45, 125751
253, 148, 105, 43, 127765	253, 160, 93, 67, 127765
254, 161, 93, 68, 128778	257, 170, 87, 83, 131841
261, 131, 130, 1, 135981	261, 164, 97, 67, 135981
265, 162, 103, 59, 140185	265, 173, 92, 81, 140185
270, 153, 117, 36, 145530	271, 151, 120, 31, 146611
272, 153, 119, 34, 147696	276, 145, 131, 14, 152076
281, 141, 140, 1, 157641	281, 186, 95, 91, 157641
284, 169, 115, 54, 161028	285, 167, 118, 49, 162165
290, 183, 107, 76, 167910	291, 161, 130, 31, 169071
293, 168, 125, 43, 171405	300, 161, 139, 22, 179700
300, 185, 115, 70, 179700	305, 174, 131, 43, 185745
313, 158, 155, 3, 195625	316, 209, 107, 102, 199396

Continua na próxima página

Tabela 24 – *Continuação da página anterior*

m,a,b, c, $N_{\oplus}$	m,a,b, c, $N_{\oplus}$
317, 208, 109, 99, 200661	324, 179, 145, 34, 209628
324, 189, 135, 54, 209628	325, 179, 146, 33, 210925
326, 193, 133, 60, 212226	327, 206, 121, 85, 213531
330, 207, 123, 84, 217470	332, 203, 129, 74, 220116
337, 222, 115, 107, 226801	341, 223, 118, 105, 232221
342, 197, 145, 52, 233586	342, 217, 125, 92, 233586
348, 213, 135, 78, 241860	348, 219, 129, 90, 241860
355, 181, 174, 7, 251695	355, 229, 126, 103, 251695
357, 203, 154, 49, 254541	359, 226, 133, 93, 257403
360, 207, 153, 54, 258840	363, 184, 179, 5, 263175
365, 196, 169, 27, 266085	365, 203, 162, 41, 266085
365, 243, 122, 121, 266085	369, 245, 124, 121, 271953
377, 210, 167, 43, 283881	377, 233, 144, 89, 283881
382, 205, 177, 28, 291466	388, 225, 163, 62, 300700
391, 251, 140, 111, 305371	395, 248, 147, 101, 311655
398, 209, 189, 20, 316410	400, 203, 197, 6, 319600
402, 267, 135, 132, 322806	403, 233, 170, 63, 324415
407, 258, 149, 109, 330891	410, 219, 191, 28, 335790
415, 254, 161, 93, 344035	418, 259, 159, 100, 349030
420, 217, 203, 14, 352380	420, 259, 161, 98, 352380
421, 271, 150, 121, 354061	430, 281, 149, 132, 369370
432, 243, 189, 54, 372816	436, 285, 151, 134, 379756
439, 291, 148, 143, 385003	440, 231, 209, 22, 386760
443, 253, 190, 63, 392055	443, 260, 183, 77, 392055
450, 227, 223, 4, 404550	454, 281, 173, 108, 411778
457, 253, 204, 49, 417241	461, 299, 162, 137, 424581
463, 259, 204, 55, 428275	463, 270, 193, 77, 428275
468, 255, 213, 42, 437580	468, 273, 195, 78, 437580
475, 241, 234, 7, 450775	475, 272, 203, 69, 450775
477, 316, 161, 155, 454581	482, 247, 235, 12, 464166
485, 267, 218, 49, 469965	485, 316, 169, 147, 469965
494, 321, 173, 148, 487578	503, 283, 220, 63, 505515
509, 255, 254, 1, 517653	516, 303, 213, 90, 531996
517, 316, 201, 115, 534061	523, 285, 238, 47, 546535
529, 346, 183, 163, 559153	530, 303, 227, 76, 561270
534, 269, 265, 4, 569778	535, 294, 241, 53, 571915
540, 271, 269, 2, 582660	540, 275, 265, 10, 582660
540, 299, 241, 58, 582660	540, 319, 221, 98, 582660
540, 335, 205, 130, 582660	541, 295, 246, 49, 584821
545, 353, 192, 161, 593505	546, 343, 203, 140, 595686
551, 294, 257, 37, 606651	551, 339, 212, 127, 606651
556, 301, 255, 46, 617716	557, 312, 245, 67, 619941
557, 360, 197, 163, 619941	558, 349, 209, 140, 622170
562, 367, 195, 172, 631126	563, 368, 195, 173, 633375
566, 373, 193, 180, 640146	569, 289, 280, 9, 646953
570, 327, 243, 84, 649230	572, 333, 239, 94, 653796
573, 332, 241, 91, 656085	577, 361, 216, 145, 665281
580, 365, 215, 150, 672220	582, 333, 249, 84, 676866
585, 374, 211, 163, 683865	586, 355, 231, 124, 686206
587, 388, 199, 189, 688551	589, 376, 213, 163, 693253
592, 297, 295, 2, 700336	596, 331, 265, 66, 709836
600, 315, 285, 30, 719400	600, 345, 255, 90, 719400
601, 338, 263, 75, 721801	603, 392, 211, 181, 726615
606, 321, 285, 36, 733866	612, 353, 259, 94, 748476
613, 360, 253, 107, 750925	613, 392, 221, 171, 750925
617, 365, 252, 113, 760761	619, 340, 279, 61, 765703
625, 381, 244, 137, 780625	626, 399, 227, 172, 783126
631, 379, 252, 127, 795691	635, 364, 271, 93, 805815
635, 376, 259, 117, 805815	639, 374, 265, 109, 816003
642, 427, 215, 212, 823686	643, 345, 298, 47, 826255
647, 402, 245, 157, 836571	648, 325, 323, 2, 839160
649, 390, 259, 131, 841753	650, 371, 279, 92, 844350
653, 379, 274, 105, 852165	656, 331, 325, 6, 860016
660, 341, 319, 22, 870540	660, 357, 303, 54, 870540
672, 343, 329, 14, 902496	675, 361, 314, 47, 910575
676, 419, 257, 162, 913276	680, 357, 323, 34, 924120
680, 425, 255, 170, 924120	682, 375, 307, 68, 929566
685, 432, 253, 179, 937765	685, 448, 237, 211, 937765

Continua na próxima página

Tabela 24 – *Continuação da página anterior*

m,a,b, c, $N_{\oplus}$	m,a,b, c, $N_{\oplus}$
689, 381, 308, 73, 948753	689, 386, 303, 83, 948753
690, 403, 287, 116, 951510	693, 427, 266, 161, 959805
703, 383, 320, 63, 987715	703, 402, 301, 101, 987715
708, 415, 293, 122, 1001820	709, 360, 349, 11, 1004653
715, 449, 266, 183, 1021735	715, 461, 254, 207, 1021735
719, 394, 325, 69, 1033203	720, 405, 315, 90, 1036080
725, 396, 329, 67, 1050525	729, 389, 340, 49, 1062153
732, 435, 297, 138, 1070916	734, 449, 285, 164, 1076778
740, 391, 349, 42, 1094460	741, 440, 301, 139, 1097421
743, 482, 261, 221, 1103355	750, 453, 297, 156, 1124250
753, 485, 268, 217, 1133265	755, 384, 371, 13, 1139295
757, 383, 374, 9, 1145341	758, 397, 361, 36, 1148370
761, 501, 260, 241, 1157481	762, 483, 279, 204, 1160526
765, 508, 257, 251, 1169685	767, 503, 264, 239, 1175811
769, 465, 304, 161, 1181953	769, 509, 260, 249, 1181953
775, 403, 372, 31, 1200475	775, 434, 341, 93, 1200475
777, 490, 287, 203, 1206681	778, 427, 351, 76, 1209790
780, 411, 369, 42, 1216020	780, 481, 299, 182, 1216020
782, 461, 321, 140, 1222266	785, 438, 347, 91, 1231665
788, 485, 303, 182, 1241100	789, 511, 278, 233, 1244253
802, 491, 311, 180, 1285606	803, 417, 386, 31, 1288815
804, 409, 395, 14, 1292028	804, 507, 297, 210, 1292028
810, 459, 351, 108, 1311390	813, 424, 389, 35, 1321125
817, 413, 404, 9, 1334161	819, 476, 343, 133, 1340703
820, 485, 335, 150, 1343980	820, 495, 325, 170, 1343980
821, 424, 397, 27, 1347261	828, 435, 393, 42, 1370340
834, 439, 395, 44, 1390278	836, 425, 411, 14, 1396956
839, 442, 397, 45, 1407003	839, 554, 285, 269, 1407003
852, 553, 299, 254, 1450956	853, 532, 321, 211, 1454365
857, 537, 320, 217, 1468041	860, 457, 403, 54, 1478340
860, 521, 339, 182, 1478340	861, 539, 322, 217, 1481781
869, 480, 389, 91, 1509453	869, 511, 358, 153, 1509453
873, 494, 379, 115, 1523385	877, 452, 425, 27, 1537381
879, 526, 353, 173, 1544403	881, 481, 400, 81, 1551441
883, 532, 351, 181, 1558495	883, 557, 326, 231, 1558495
885, 503, 382, 121, 1565565	890, 527, 363, 164, 1583310
892, 581, 311, 270, 1590436	893, 492, 401, 91, 1594005
894, 521, 373, 148, 1597578	900, 483, 417, 66, 1619100
900, 525, 375, 150, 1619100	900, 555, 345, 210, 1619100
902, 521, 381, 140, 1626306	903, 539, 364, 175, 1629915
908, 503, 405, 98, 1648020	913, 465, 448, 17, 1666225
916, 509, 407, 102, 1677196	916, 539, 377, 162, 1677196
921, 605, 316, 289, 1695561	923, 484, 439, 45, 1702935
926, 465, 461, 4, 1714026	926, 533, 393, 140, 1714026
929, 506, 423, 83, 1725153	930, 527, 403, 124, 1728870
933, 520, 413, 107, 1740045	937, 593, 344, 249, 1755001
940, 585, 355, 230, 1766260	942, 553, 389, 164, 1773786
948, 627, 321, 306, 1796460	949, 596, 353, 243, 1800253
957, 563, 394, 169, 1830741	958, 517, 441, 76, 1834570
963, 625, 338, 287, 1853775	964, 591, 373, 218, 1857628
967, 487, 480, 7, 1869211	967, 587, 380, 207, 1869211
972, 567, 405, 162, 1888596	972, 603, 369, 234, 1888596
978, 543, 435, 108, 1911990	978, 579, 399, 180, 1911990
979, 534, 445, 89, 1915903	981, 544, 437, 107, 1923741
983, 558, 425, 133, 1931595	984, 501, 483, 18, 1935528
990, 621, 369, 252, 1959210	991, 594, 397, 197, 1963171
995, 501, 494, 7, 1979055	996, 609, 387, 222, 1983036
997, 568, 429, 139, 1987021	1000, 575, 425, 150, 1999000
1002, 571, 431, 140, 2007006	1003, 537, 466, 71, 2011015
1010, 583, 427, 156, 2039190	1010, 603, 407, 196, 2039190
1012, 555, 457, 98, 2047276	1012, 657, 355, 302, 2047276
1014, 649, 365, 284, 2055378	1017, 602, 415, 187, 2067561
1019, 601, 418, 183, 1039379	