UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
CAMPUS DE ILHA SOLTEIRA

JOSÉ VIRGILIO DE OLIVEIRA JÚNIOR

MINIMIZAÇÃO DE FUNÇÕES MAJORITÁRIAS DE 3 ENTRADAS COMO UM PROBLEMA DE PROGRAMAÇÃO NÃO LINEAR INTEIRA BINÁRIA

Ilha Solteira - SP
2025

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

JOSÉ VIRGILIO DE OLIVEIRA JÚNIOR

MINIMIZAÇÃO DE FUNÇÕES MAJORITÁRIAS DE 3 ENTRADAS COMO UM PROBLEMA DE PROGRAMAÇÃO NÃO LINEAR INTEIRA BINÁRIA

Tese apresentada à Universidade Estadual Paulista (UNESP), Faculdade de Engenharia de Ilha Solteira, para obtenção do título de Doutor em Engenharia Elétrica.

Área de Concentração: Automação.

Orientador: Prof. Dr. Alexandre César Rodrigues da Silva

Ilha Solteira - SP
2025

FICHA CATALOGRÁFICA
Desenvolvida pela Diretoria Técnica de Biblioteca e Documentação

O48m

Oliveira Júnior, José Virgilio de.
Minimização de funções majoritárias de 3 entradas como um problema de programação não linear inteira binária / José Virgilio de Oliveira Júnior. -- Ilha Solteira: [s.n.], 2025
111 f. : il.

Tese (doutorado) - Universidade Estadual Paulista. Faculdade de Engenharia de Ilha Solteira. Área de conhecimento: Automação, 2025

Orientador: Alexandre César Rodrigues da Silva
Inclui bibliografia

1. Álgebra booleana. 2. Cplex. 3. Lógica majoritária. 4. Portas lógicas. 5. Programação matemática.

Elaborada por Raiane da Silva Santos - CRB-8/9999

IMPACTO POTENCIAL DESTA PESQUISA

Embora o Brasil não atue diretamente no desenvolvimento de nanotecnologias de computação, esta pesquisa ratifica o fato de que o país detém conhecimento técnico e científico de nível internacional no que se trata da modelagem matemática para a análise e síntese dos circuitos destas tecnologias. Foi apresentada uma teoria inovadora para a problemática da minimização e otimização de circuitos lógicos combinacionais com portas lógicas majoritárias de 3 entradas, que pode vir a se tornar a base para o desenvolvimento de teorias e métodos mais sofisticados utilizados nestas nanotecnologias. Confirma a competência da instituição na formação de especialistas nesta vertente tecnológica, que podem atuar como professores e pesquisadores, ampliando regional e nacionalmente o conhecimento adquirido e desenvolvido. A pesquisa também destaca a versatilidade do uso da programação matemática e da linguagem de programação computacional, que devem ser ferramentas a serem amplamente divulgadas para que possam ser cada vez mais utilizadas no desenvolvimento econômico, social e tecnológico em uma nação com dimensões continentais.

POTENTIAL IMPACT OF THIS RESEARCH

Although Brazil is not directly involved in the development of computing nanotechnologies, this research confirms that the country possesses international-level technical and scientific knowledge regarding mathematical modeling for the analysis and synthesis of circuits in these technologies. An innovative theory was presented for the problem of minimizing and optimizing combinational logic circuits with 3-input majority logic gates, which could become the basis for the development of more sophisticated theories and methods used in these nanotechnologies. It confirms the institution's competence in training specialists in this technological field, who can work as teachers and researchers, expanding the acquired and developed knowledge regionally and nationally. The research also highlights the versatility of using mathematical programming and computational programming languages, which should be widely disseminated tools so that they can be increasingly used in the economic, social, and technological development of a nation with continental dimensions.

UNIVERSIDADE ESTADUAL PAULISTA

Câmpus de Ilha Solteira

CERTIFICADO DE APROVAÇÃO

TÍTULO DA TESE: Minimização de Funções Majoritárias de 3 Entradas como um Problema de Programação não Linear Inteira Binária

AUTOR: JOSÉ VIRGILIO DE OLIVEIRA JÚNIOR
ORIENTADOR: ALEXANDRE CESAR RODRIGUES DA SILVA

Aprovado como parte das exigências para obtenção do Título de Doutor em Engenharia Elétrica, área: Automação pela Comissão Examinadora:

Prof. Dr. ALEXANDRE CESAR RODRIGUES DA SILVA (Participaçao Presencial)
Departamento de Engenharia Elétrica / Faculdade de Engenharia de Ilha Solteira - UNESP

Prof. Dr. JOSE ROBERTO SANCHES MANTOVANI (Participaçao Presencial)
Departamento de Engenharia Elétrica / Faculdade de Engenharia de Ilha Solteira - UNESP

Prof. Dr. JONATAS BOAS LEITE (Participaçao Presencial)
Departamento de Engenharia Elétrica / Faculdade de Engenharia de Ilha Solteira - UNESP

Prof. Dr. EDSON ANTONIO BATISTA (Participaçao Virtual)
Departamento de Engenharia Elétrica / Faculdade de Engenharias, Arquitetura e Urbanismo e Geografia - FAENG/UFMS

Prof. Dr. VLADEMIR DE JESUS SILVA OLIVEIRA (Participaçao Virtual)
Curso de Engenharia Elétrica / Faculdade de Ciências Exatas e Tecnológicas - FACET/UNEMAT

Ilha Solteira, 06 de novembro de 2025.

AGRADECIMENTOS

Agradeço aos meus pais e irmãos, pela família sólida e pelo apoio que sempre concederam neste processo.

Ao Professor Dr. Alexandre César Rodrigues da Silva, pela oportunidade e pela autonomia concedidas, que permitiram o desenvolvimento adequado deste trabalho.

Aos professores, técnicos e funcionários do Departamento de Engenharia Elétrica pela atenção e prestatividade que concedem aos alunos.

A todos os professores, técnicos, servidores e funcionários da FEIS que atuam como engrenagens, mantendo em pleno funcionamento toda a instituição.

Ao CNPq, pela concessão da bolsa de pesquisa (processo 152852/2021-2).

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001.

RESUMO

Com a tecnologia CMOS atingindo limitações físicas, nanotecnologias emergentes baseadas em portas lógicas majoritárias têm ganhado destaque. Um dos grandes desafios é a minimização dos circuitos que utilizam essa arquitetura. Tradicionalmente, métodos de otimização dependem do uso de funções majoritárias primitivas. Esta tese demonstra que essa dependência não é obrigatória e, como objetivo principal, apresenta uma nova metodologia para a minimização de funções booleanas expressas na forma majoritária de 3 entradas. O cerne da proposta baseia-se na introdução dos conceitos de funções booleanas fundamentais, que restringem as funções primitivas, e das classes de complexidade de uma função majoritária, para estruturar a busca pela solução minimizada. Deste modo, esta teoria apresenta a problemática da minimização como um problema de programação não linear inteira binária. Esta teoria foi implementada utilizando a linguagem Python e o solucionador CPLEX, obtendo-se resultados experimentais que validaram a abordagem, minimizando funções majoritárias de 3 entradas com até 15 variáveis e 5 níveis, obtendo circuitos otimizados que minimizam o número de portas e níveis lógicos. O trabalho estabelece, portanto, um novo paradigma teórico e computacional para a síntese de circuitos em lógica majoritária, podendo ser a base para o desenvolvimento de técnicas de otimização mais eficientes no futuro.

Palavras-chave: álgebra booleana; cplex; lógica majoritária; portas lógicas; programação matemática.

ABSTRACT

With CMOS technology reaching physical limitations, emerging nanotechnologies based on majority logic gates have gained prominence. One of the major challenges is the minimization of circuits using this architecture. Traditionally, optimization methods depend on the use of primitive majority functions. This thesis demonstrates that this dependence is not mandatory and, as its main objective, presents a new methodology for minimizing Boolean functions expressed in 3-input majority form. The core of the proposal is based on the introduction of the concepts of fundamental Boolean functions, which restrict primitive functions, and the complexity classes of a majority function, to structure the search for the minimized solution. Thus, this theory presents the problem of minimization as a binary integer nonlinear programming problem. This theory was implemented using the Python language and the CPLEX solver, obtaining experimental results that validated the approach, minimizing 3-input majority functions with up to 15 variables and 5 levels, obtaining optimized circuits that minimize the number of gates and logic levels. This work therefore establishes a new theoretical and computational paradigm for the synthesis of circuits in majority logic, and may form the basis for the development of more efficient optimization techniques in the future.

Keywords: boolean algebra; cplex; logic gates; majority logic; mathematical programming.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
bit	*binary digit*
CCFM	Classe de Complexidade de uma Função Majoritária
CI	Circuito Integrado
CMOS	*Complementary Metal-Oxide-Semiconductor*
FBF	Função Booleana Fundamental
MLP01	*Majority Linear Programming 0 or 1*
MOSFET	*Metal Oxide Semiconductor Field Effect Transistor*
POS	*Product of Sums*
PPNLIB	Problema de Programação Não Linear Inteira Binária
QCA	*Quantum-dot Cellular Automata*
RAM	*Random Access Memory*
SOP	*Sum of Products*
TTL	*Transistor-Transistor Logic*

SUMÁRIO

1 INTRODUÇÃO

Numa das Álgebras Booleanas propostas por Boole (1854), os números 0 e 1 podem ser operados de uma forma própria por meio das operações de adição, multiplicação e inversão. Esta álgebra estabeleceu o alicerce matemático para toda uma tecnologia desenvolvida décadas depois com a aplicação em circuitos elétricos chaveados, proposta por Shannon (1938) e considerada uma das teorias científicas mais revolucionárias da história, que foi precursora no surgimento da computação eletrônica e digital. A computação e os sistemas digitais sofreram outra grande revolução com o surgimento das válvulas, diodos e transistores.

Com a rápida evolução dos transistores, a tecnologia *CMOS (Complementary Metal Oxide Semiconductor)* propagou-se rapidamente com o surgimento dos transistores *MOSFET (Metal Oxide Semiconductor Field Effect Transistor)*. Estes transistores constituem a base da arquitetura do projeto dos modernos dispositivos CI (Circuito Integrado), microprocessador e memória *RAM (Random Access Memory)*, dentre outros. Estes dispositivos operam com uma enorme quantidade de portas lógicas e representam a estrutura básica de um sistema digital, de modo que as informações são transmitidas por meio de sinais elétricos, simulando as constantes da álgebra booleana e suas operações de forma aplicada (Franco, 2015).

Nas últimas décadas, muitas pesquisas têm sido realizadas visando desenvolver tecnologias alternativas àquelas que empregam a tecnologia *CMOS*, no que parece ser um provável fim da Lei de Moore (Zhang, et al., 2023), que prevê o dobro de transistores em um CI a cada 2 anos, aproximadamente. Muitas nanotecnologias emergentes tem projetado circuitos baseados na denominada lógica majoritária, conhecidamente mais expressiva e eficiente em aspectos de hardware que a lógica booleana convencional. Uma forte pretensão destas pesquisas é lidar com limitações físicas atualmente enfrentadas pela tecnologia *CMOS*.

Paralelamente, lidar com o gargalo de Von Neumann, que trata dos dilemas de operação entre processador e memória, tem impulsionado o desenvolvimento de novas arquiteturas computacionais, como proposto por Rafiq et al. (2024) no desenvolvimento de estruturas para computação aproximada utilizando portas lógicas majoritárias de 3 entradas. A computação aproximada é um paradigma emergente com eficiência de área e energia.

A noção circuital majoritária é comercialmente conhecida, por exemplo, através do CI TC4530BP (Toshiba, 1985) e do CI MC14530B (Motorola, 1995) que disponibilizam um par de circuitos que retornam como saída a maioria de 5 entradas e são baseados na tecnologia *TTL (Transistor-Transistor Logic)*. Naturalmente, como estes CI's são projetados com portas lógicas convencionais, não se enquadram na lógica majoritária proposta.

Décadas antes de sua popularização, a lógica majoritária já era estudada como uma possível facilitadora na implementação de circuitos lógicos. Utilizando-se de radiofrequência, Wigington (1959) mostrou que era possível implementar portas lógicas com mais eficiência através do transporte da informação nas fases dos sinais senoidais. Quando aplicada a esta nova lógica, prometia maior desempenho, visto que já haviam limitações na velocidade de processamento para a tecnologia da época. Uma teoria lógica majoritária foi inicialmente proposta por Lindaman (1960), que lhe atribuiu um formalismo matemático.

Atualmente, a lógica majoritária tem sido empregada na construção de diversos elementos computacionais, como memórias RAM (Lakshmi; Reuben; Pudi, 2022), somadores e multiplicadores (Lagid et al., 2021) e circuitos lógicos com autômatos celulares da tecnologia *QCA (Quantum-dot Cellular Automata)* (Raj et al., 2022). A proposta do trabalho aqui apresentado não é discorrer a respeito da estrutura física de tecnologias em desenvolvimento, mas sim propor uma nova metodologia que permita obter circuitos simplificados e otimizados com funções majoritárias visando serem utilizados nestas tecnologias.

1.1 O PROBLEMA DA OTIMIZAÇÃO MAJORITÁRIA

No sucesso da tecnologia *CMOS* é crucial a simplificação e minimização das funções booleanas e dos circuitos lógicos combinacionais correspondentes. Isso reduz os custos de produção, de espaço físico e do consumo de energia. Assim, pode-se aumentar a eficiência total destes sistemas digitais. Na minimização de funções booleanas destacam-se o método Mapa de Karnaugh (Karnaugh, 1953) e o método de Quine-McCluskey (Mccluskey, 1956). A execução adequada do Mapa de Karnaugh fornece o formato algébrico mais minimizado de uma função booleana.

Os circuitos lógicos combinacionais e sequenciais são mais facilmente implementados com portas lógicas majoritárias, geralmente resultando em menos

componentes do que os obtidos com as portas lógicas convencionais. No entanto, a lógica majoritária também apresenta desafios únicos em várias instâncias, como o design aritmético e a síntese. Portanto, é imperativo que sejam desenvolvidos métodos para a simplificação e a minimização destes circuitos. De fato, ainda não existe um método robusto e consagrado para a minimização de funções majoritárias, no que se refere à redução da quantidade de variáveis e da quantidade de níveis. O processo de minimização não é elementar, até mesmo para um sistema de pequeno porte. Portanto é crucial o desenvolvimento de métodos que possibilitem uma minimização eficiente.

São precursores dos processos de otimização majoritária as noções de funções majoritárias primitivas propostas por Wang et al. (2015) e as bases de teoremas e axiomas da lógica majoritária proposta por Chattopadhyay et al. (2016). A utilização das funções primitivas tem sido feita de forma intrínseca aos problemas de otimização. Num cenário onde se utilizam portas lógicas majoritárias, um dos desafios é desenvolver técnicas que permitam obter uma função, proveniente de uma tabela-verdade, em seu formato majoritário simplificado e otimizado. Assim, diversas heurísticas tem sido adotadas com o objetivo de obter métodos eficientes na simplificação de funções majoritárias.

O método exact_mig foi pioneiro na obtenção de resultados consideráveis, fazendo uso do conceito de grafos e de funções primitivas, para obter o que chama de síntese exata (Soeken et al., 2017). Também utilizando a ideia de grafos, Chu et al. (2019) estabelece vínculos entre portas majoritárias e portas XOR. Riener et al. (2019) utiliza o conceito de funções monotônicas para obtenção de funções majoritárias.

A Programação Matemática foi um recurso utilizado por Ferraz (2018) e no método 3MS que explora mais de um critério de custo na obtenção das funções majoritárias (FERRAZ, 2022). A pesquisa apresentada neste trabalho pretendia ser uma progressão da pesquisa desenvolvida por Oliveira Júnior (2021) no método *MLP01 (Majority Linear Programming 0 or 1)*, mostrando que não seriam mais necessárias as funções primitivas e continuando com a programação linear inteira binária. Porém, de maneira inovadora, convergiu para uma análise de programação não linear inteira binária.

Os atuais métodos ainda atingem poucas variáveis e poucos níveis da estrutura majoritária de 3 entradas, de modo que existe uma grande lacuna a ser preenchida

neste contexto. Este trabalho visa propor uma nova metodologia para a obtenção de funções majoritárias minimizadas de 3 entradas. São discutidas questões relativas à necessidade das funções primitivas e uso da programação matemática não linear binária. Este trabalho também não visa atender a uma tecnologia específica, mas sim ser uma plataforma que possa ser utilizada naquelas que utilizam portas lógicas majoritárias de 3 entradas.

1.2 OBJETIVOS DA PESQUISA

A programação matemática inteira foi um recurso utilizado por Ferraz (2018) para a síntese de funções majoritárias de 3 entradas. Também foi utilizada no método 3MS que explora mais de um critério de custo na obtenção das funções majoritárias (Ferraz, 2022). Inicialmente, a pesquisa apresentada neste trabalho tinha a pretensão de ser uma progressão da pesquisa desenvolvida por Oliveira Júnior (2021) no método *MLP01* (*Majority Linear Programming 0 or 1*) mostrando que, além de não serem mais necessárias todas as funções primitivas, podia-se ampliar o uso da programação linear inteira binária explorando mais variáveis e níveis. Porém, de maneira inovadora, a pesquisa convergiu para uma análise utilizando programação não linear inteira binária.

Os atuais métodos ainda atingem poucas variáveis e poucos níveis da estrutura majoritária de 3 entradas, de modo que existe uma grande lacuna a ser preenchida neste contexto. Inicialmente, este trabalho demonstra que não há obrigatoriedade no uso de todas as funções majoritárias primitivas, mas sim parte delas. A partir desta constatação, definem-se as funções booleanas fundamentais.

Posteriormente propõe-se uma nova metodologia para a obtenção de funções majoritárias minimizadas de 3 entradas, definindo-se as classes de complexidade de uma função majoritária. Também são definidas as variáveis funcionais e as variáveis vetoriais, mostrando que encontrar os bits de uma função do problema precede sua identificação. A partir destes conceitos é desenvolvida uma nova metodologia de minimização modelada como um problema de programação não linear inteira binária.

São discutidas questões relativas à necessidade das funções primitivas e o uso da programação matemática não linear inteira binária para o processo de minimização de funções majoritárias de 3 entradas. Este trabalho não visa atender a uma

tecnologia específica, mas sim ser uma plataforma que possa ser utilizada naquelas que utilizam portas lógicas majoritárias de 3 entradas.

1.3 ESTRUTURA DE DESENVOLVIMENTO

No Capítulo 2 é descreve-se uma breve revisão da Álgebra Booleana e como ela é empregada no projeto de circuitos lógicos combinacionais, de modo a auxiliar na compreensão da teoria desenvolvida. No Capítulo 3 é apresentada e desenvolvida a Lógica Majoritária de 3 entradas dentro dos princípios da álgebra booleana, para que se possa obter uma equação fundamental que atua como um dos princípios mais importantes desta teoria. Compreender estas duas lógicas é fundamental para o que foi desenvolvido. Como um dos objetivos desta pesquisa, no Capítulo 4 são discutidas as funções majoritárias primitivas e demonstrado que seu uso não é uma obrigatoriedade em processos de otimização. Também é definido o conceito de função booleana fundamental, outro princípio para a concepção desta teoria.

No Capítulo 5 são estabelecidas as bases de uma nova proposta teórica para a minimização de funções majoritárias utilizando programação matemática e no Capítulo 6 é elaborado o modelo que descreve a proposta de solução para este problema. No Capítulo 7 é apresentada uma proposta de implementação utilizando linguagem Python e o solucionador CPLEX. Alguns exemplos de aplicação e os resultados obtidos são apresentados no Capítulo 8. Por fim, no Capítulo 9 são apresentadas as conclusões decorrentes desta pesquisa.

2 ÁLGEBRA BOOLEANA E OS CIRCUITOS DIGITAIS

No desenvolvimento do pensamento humano, tanto na Filosofia quanto nas Ciências Exatas, o raciocínio lógico tem importância crucial. Neste contexto, a Lógica Proposicional é fundamental na progressão do estudo de lógicas matemáticas. Uma proposição pode ser definida como qualquer declaração afirmativa completa que possa ser avaliada como verdadeira ou falsa, mas nunca com os dois conceitos simultâneos (Daghlian, 1986). Por exemplo, dentro do universo da matemática dos números reais, a sentença $0 + 4 = 7$ é uma proposição falsa, de modo que a sentença $0 + 4 \neq 7$ é uma proposição verdadeira. Assim, uma proposição nunca pode ser classificada simultaneamente com os dois valores lógicos: se ela for verdadeira, não poderá ser falsa; sendo falsa, não poderá ser verdadeira.

Através do desenvolvimento e evolução desta Lógica Proposicional, foram obtidas novas teorias lógicas que puderam realizar interações entre as proposições como sendo entidades matemáticas. Uma *Álgebra Booleana* constitui qualquer estrutura lógica que contenha um conjunto de números, operações algébricas para estes números e alguns teoremas que dispensam demonstração (Boole, 1854).

2.1 ÁLGEBRA BOOLEANA BINÁRIA

Embora seja constituída de poucos elementos estruturais, aqui será utilizada a uma álgebra booleana formada apenas pelos números 0 e 1, que compõem o conjunto $\mathbb{B}$. São três as operações algébricas definidas neste conjunto: a adição, a multiplicação e a inversão. A operação de *adição* ou *soma* de dois números X e Y de $\mathbb{B}$, indicada por $X + Y$, é estabelecida conforme os resultados apresentados na Tabela 1. Assim, é possível efetuar a adição de quaisquer dois números de $\mathbb{B}$.

Tabela 1 – Operação booleana de adição.

X	Y	X + Y
0	0	0
0	1	1
1	0	1
1	1	1

Fonte: Produção do próprio autor.

A operação de *multiplicação* ou *produto* de dois números X e Y de $\mathbb{B}$, indicada por $X \cdot Y$, é estabelecida conforme os resultados apresentados na Tabela 2. Assim, é possível efetuar a multiplicação de quaisquer dois números de $\mathbb{B}$. De uma forma semelhante ao que ocorre em outras álgebras, a notação da operação booleana de multiplicação pode ser feita omitindo o ponto gráfico, de modo que há a equivalência $X \cdot Y = XY$. Esta notação simplificada auxilia nas operações extensas.

Tabela 2 – Operação booleana de multiplicação.

X	Y	X · Y
0	0	0
0	1	0
1	0	0
1	1	1

Fonte: Produção do próprio autor.

A operação de *inversão* de um número X de $\mathbb{B}$, indicada por $\bar{X}$, é estabelecida conforme os resultados apresentados na Tabela 3. Assim, é possível efetuar a inversão de qualquer número de $\mathbb{B}$. Para a operação de inversão será utilizada a equivalência de notacional $\bar{X} = X'$, que facilitará tipograficamente os cálculos extensos e poderá conferir mais elegância às sentenças.

Tabela 3 – Operação booleana de inversão.

X	$\bar{X}$
0	1
1	0

Fonte: Produção do próprio autor.

Da forma como foram definidas as três operações desta álgebra booleana binária é possível efetuar a adição e a multiplicação de quaisquer dois números de $\mathbb{B}$ e a inversão de qualquer número de $\mathbb{B}$. Numa primeira análise, um sistema tão pequeno composto por dois números e três operações algébricas pode parecer irrelevante. Porém, quando aplicado adequadamente é altamente poderoso. Simplificando, deste ponto em diante, esta álgebra booleana binária é referida apenas como álgebra booleana.

2.1.1 Teoremas da Álgebra Booleana

Para que a álgebra booleana tenha mais consistência e aplicabilidade, vários teoremas podem ser incorporados a ela em decorrência da definição de suas três operações algébricas. Na Tabela 4 estão 15 teoremas da álgebra booleana que são necessários ao desenvolvimento dos próximos capítulos. Os teoremas B_{14} e B_{15}, conhecidos como *Teoremas de De Morgan*, têm uma importância de destaque.

Tabela 4 – Teoremas da Álgebra Booleana.

REFERÊNCIA	TEOREMA
B_1	$X + 0 = X$
B_2	$X + 1 = 1$
B_3	$X + X = X$
B_4	$X + \bar{X} = 1$
B_5	$X \cdot 0 = 0$
B_6	$X \cdot 1 = X$
B_7	$X \cdot X = X$
B_8	$X \cdot \bar{X} = 0$
B_9	$X + Y = Y + X$
B_{10}	$X \cdot Y = Y \cdot X$
B_{11}	$X + Y + Z = (X + Y) + Z = X + (Y + Z)$
B_{12}	$X \cdot Y \cdot Z = (X \cdot Y) \cdot Z = X \cdot (Y \cdot Z)$
B_{13}	$X \cdot (Y + Z) = X \cdot Y + X \cdot Z$
B_{14}	$(X + Y)' = \bar{X} \cdot \bar{Y}$
B_{15}	$(X \cdot Y)' = \bar{X} + \bar{Y}$

Fonte: Produção do próprio autor.

Com exceção dos teoremas B_{11} e B_{12}, todos os outros 13 teoremas podem ser demonstrados utilizando o método de indução perfeita, no qual se deve verificar a validade de todas as possibilidades para os números booleanos X, Y e Z. Como exemplo, nas Tabelas 5 e 6 estão demonstrados os teoremas B_{14} e B_{15}, respectivamente. As operações de adição e de multiplicação foram definidas para apenas dois números, de modo que os teoremas B_{11} e B_{12} podem ser interpretados como axiomas ou até mesmo como metodologias para se operar com mais de dois números nestas operações.

Tabela 5 – Demonstração do teorema B_{14}.

X	Y	$(X+Y)'$	$\bar{X}\cdot\bar{Y}$
0	0	$(0+0)'=\bar{0}=1$	$\bar{0}\cdot\bar{0}=1\cdot 1=1$
0	1	$(0+1)'=\bar{1}=0$	$\bar{0}\cdot\bar{1}=1\cdot 0=0$
1	0	$(1+0)'=\bar{1}=0$	$\bar{1}\cdot\bar{0}=0\cdot 1=0$
1	1	$(1+1)'=\bar{1}=0$	$\bar{1}\cdot\bar{1}=0\cdot 0=0$

Fonte: Produção do próprio autor.

Tabela 6 – Demonstração do teorema B_{15}.

X	Y	$(X\cdot Y)'$	$\bar{X}+\bar{Y}$
0	0	$(0\cdot 0)'=\bar{0}=1$	$\bar{0}+\bar{0}=1+1=1$
0	1	$(0\cdot 1)'=\bar{0}=1$	$\bar{0}+\bar{1}=1+0=1$
1	0	$(1\cdot 0)'=\bar{0}=1$	$\bar{1}+\bar{0}=0+1=1$
1	1	$(1\cdot 1)'=\bar{1}=0$	$\bar{1}+\bar{1}=0+0=0$

Fonte: Produção do próprio autor.

Nas Tabelas 1, 2 e 3 ficam estabelecidas as três operações algébricas da álgebra booleana de dois números. Os teoremas B_{11} e B_{12} fornecem uma forma de se executar as operações com mais de dois números. A associatividade estabelecida por estes dois teoremas e a comutatividade dos teoremas B_9 e B_{10} podem ser expandidas a uma quantidade ilimitada de números.

Existe uma hierarquia entre as operações algébricas booleanas. A inversão deve ser efetuada primeiro, seguida da multiplicação e da adição. A correta aplicação da hierarquia das operações, da hierarquia no uso de parênteses e dos teoremas permite efetuar quaisquer sequências de operações algébricas, por mais extensas que sejam. Para exemplificar, em (1) são efetuadas algumas operações booleanas com a aplicação deste princípio. Como pode ser observado, a correta execução das operações e da ordenação dos parênteses permite efetuar cálculos extensos que se reduzem a um único número.

$$0+1+(\bar{0}\cdot 1+(\bar{1}+0\cdot\bar{1}+1))\cdot 1\cdot\bar{0}=(0+1)+(1\cdot 1+(0+0\cdot 0+1))\cdot 1\cdot 1=1+(1+(0+0+1))\cdot(1\cdot 1)=1+(1+((0+0)+1))\cdot 1=1+(1+(0+1))\cdot 1=1+(1+1)\cdot 1=1+1=1 \quad (1)$$

2.2 O EMPREGO DA ÁLGEBRA BOOLEANA NOS SISTEMAS DIGITAIS

Décadas após a proposta da álgebra booleana, constatou-se que ela poderia ser aplicada em circuitos elétricos chaveados, utilizados para a representação e solução de problemas lógicos (Shannon, 1938). A utilização dos números booleanos 0 e 1 para representar números na base binária, aliada ao surgimento das válvulas, dos diodos e dos transistores permitiram todo um desenvolvimento da computação. Este sistema estrutural continua sendo utilizado até hoje nos circuitos digitais computacionais mais modernos.

2.2.1 Tabela-verdade e função booleana

Considere um sistema arbitrário formado por 4 botões e uma lâmpada, de modo que dependendo da combinação de botões pressionados esta lâmpada pode ou não acender. Designando por A, B, C e D as situações de acionamento dos 4 botões e por S a situação de acendimento da lâmpada, é possível criar a Tabela 7, que comporta todas as 16 situações possíveis, utilizando a álgebra booleana. Neste caso, o número 0 indica que o não acionamento de um botão ou lâmpada; o número 1 indica o acionamento.

Uma estrutura tabular que descreva todas as possibilidades de um sistema recebe o nome de *tabela-verdade*. Em projetos de estruturas lógicas, incluindo os de estruturas digitais, uma tabela-verdade pode descrever o comportamento de um sistema, listando todos os arranjos possíveis para as entradas e todas as possibilidades de saída decorrentes destas entradas (Wakerly, 2006). Portanto, a Tabela 7 representa a tabela-verdade do sistema arbitrário proposto, na qual A, B, C e D são as *entradas* do sistema e S é a *saída* do sistema.

Cada uma das 16 linhas da tabela-verdade do exemplo corresponde a 1 possibilidade de arranjo das 4 entradas vinculadas à sua saída correspondente. De forma geral, em um sistema de m entradas há 2^m linhas em sua tabela-verdade. Relativamente a estas entradas, pode haver quantas saídas forem necessárias; no exemplo há apenas 1 saída.

Em um sistema expresso por uma tabela-verdade, de forma semelhante ao que ocorre com as funções na álgebra de números reais, é possível estabelecer uma correspondência direta entre os valores ordenados das entradas e os valores das

saídas. Para o exemplo em questão, a equação (2) lista todas as 16 relações do sistema. A esta relação de correspondência dá-se o nome de *função booleana* (Rosen, 2019). Também há uma correspondência direta entre a função booleana e sua respectiva tabela-verdade. Assim como nas funções reais, numa função booleana, a cada um dos valores de entrada corresponde apenas um valor de saída.

Tabela 7 – Tabela-verdade de um sistema.

A	B	C	D	S
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	0

Fonte: Produção do próprio autor.

2.2.2 Expressão algébrica de uma função booleana

Tratando as entradas e as saídas de um sistema como variáveis de uma função booleana, é possível estabelecer uma regra algébrica entre elas. Para que se possa desenvolver uma expressão algébrica que represente uma função booleana utilizando suas variáveis, é necessário introduzir dois conceitos. Considere um sistema lógico de m variáveis e seus 2^m arranjos dos números booleanos contidos nas linhas de sua tabela-verdade. A cada arranjo está associada um tipo de função representada pelo produto de todas estas m variáveis, podendo estar na forma original ou inversa, que recebe o nome de *mintermo*. Dualmente, cada função representada pela soma destas

m variáveis, que pode estar na forma original ou inversa, recebe o nome de *maxtermo* (Kohavi; Jha, 2010). Para os mintermos, a presença da variável inversa é determinada pelo valor 0 da variável na linha da tabela-verdade. Dualmente, a presença da variável inversa no maxtermo é determinada pelo valor 1 da variável na linha da tabela-verdade. Na Tabela 8 estão representados de forma genérica os mintermos e maxtermos para m variáveis booleanas.

$$\begin{cases} (0,0,0,0) \mapsto 1 \\ (0,0,0,1) \mapsto 1 \\ (0,0,1,0) \mapsto 0 \\ (0,0,1,1) \mapsto 0 \\ (0,1,0,0) \mapsto 1 \\ (0,1,0,1) \mapsto 1 \\ (0,1,1,0) \mapsto 0 \\ (0,1,1,1) \mapsto 0 \\ (1,0,0,0) \mapsto 0 \\ (1,0,0,1) \mapsto 1 \\ (1,0,1,0) \mapsto 0 \\ (1,0,1,1) \mapsto 1 \\ (1,1,0,0) \mapsto 0 \\ (1,1,0,1) \mapsto 0 \\ (1,1,1,0) \mapsto 1 \\ (1,1,1,1) \mapsto 0 \end{cases} \tag{2}$$

Tabela 8 – Mintermos e maxtermos de um sistema com m variáveis.

V_1	V_2	$\cdots$	V_m	MINTERMOS	MAXTERMOS	ORDEM
0	0	$\cdots$	0	$\bar{V}_1 \cdot \bar{V}_2 \cdot \cdots \cdot \bar{V}_m$	$V_1 + V_2 + \cdots + V_m$	0
0	0	$\cdots$	1	$\bar{V}_1 \cdot \bar{V}_2 \cdot \cdots \cdot V_m$	$V_1 + V_2 + \cdots + \bar{V}_m$	1
0	0	$\cdots$	0	$\bar{V}_1 \cdot \bar{V}_2 \cdot \cdots \cdot \bar{V}_m$	$V_1 + V_2 + \cdots + V_m$	2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
1	1	$\cdots$	1	$V_1 \cdot V_2 \cdot \cdots \cdot V_m$	$\bar{V}_1 + \bar{V}_2 + \cdots + \bar{V}_m$	2^m - 3
1	1	$\cdots$	0	$V_1 \cdot V_2 \cdot \cdots \cdot \bar{V}_m$	$\bar{V}_1 + \bar{V}_2 + \cdots + V_m$	2^m - 2
1	1	$\cdots$	1	$V_1 \cdot V_2 \cdot \cdots \cdot V_m$	$\bar{V}_1 + \bar{V}_2 + \cdots + \bar{V}_m$	2^m - 1

Fonte: Produção do próprio autor.

Portanto, a um sistema de m variáveis estão vinculados 2^m mintermos e 2^m maxtermos, ordenados de 0 a $m-1$, da primeira à m-ésima linha da tabela-verdade, respectivamente. Retomando o sistema do exemplo, na Tabela 9 estão representados todos os mintermos e maxtermos relativos às variáveis A, B, C e D.

A partir da tabela-verdade de um sistema, é possível obter sua função booleana algébrica de duas formas: somando seus mintermos correspondentes, ou seja, realizando a chamada soma de produtos ou *SOP (Sum of Products),* ou também multiplicando seus maxtermos, ou seja, realizando um produto de somas ou *POS (Product of Sums)*. Na coluna de saída da tabela-verdade, cada elemento 1 indica a presença do mintermo correspondente e cada elemento 0 indica a presença de um maxtermo correspondente. É comum representar o i-ésimo mintermo por m_i e o i-ésimo maxtermo por M_i, onde sempre é verificado $M_i = \overline{m}_i \Leftrightarrow m_i = \overline{M}_i$. A soma dos mintermos ou o produto dos maxtermos garantem que sejam obtidos na função algébrica todos os valores corretos da tabela-verdade, independentemente dos valores das variáveis do sistema.

Tabela 9 – Mintermos e maxtermos para o sistema de 4 entradas.

A	B	C	D	MINTERMOS	MAXTERMOS	ORDEM
0	0	0	0	$\bar{A}\cdot\bar{B}\cdot\bar{C}\cdot\bar{D}$	$A+B+C+D$	0
0	0	0	1	$\bar{A}\cdot\bar{B}\cdot\bar{C}\cdot D$	$A+B+C+\bar{D}$	1
0	0	1	0	$\bar{A}\cdot\bar{B}\cdot C\cdot\bar{D}$	$A+B+\bar{C}+D$	2
0	0	1	1	$\bar{A}\cdot\bar{B}\cdot C\cdot D$	$A+B+\bar{C}+\bar{D}$	3
0	1	0	0	$\bar{A}\cdot B\cdot\bar{C}\cdot\bar{D}$	$A+\bar{B}+C+D$	4
0	1	0	1	$\bar{A}\cdot B\cdot\bar{C}\cdot D$	$A+\bar{B}+C+\bar{D}$	5
0	1	1	0	$\bar{A}\cdot B\cdot C\cdot\bar{D}$	$A+\bar{B}+\bar{C}+D$	6
0	1	1	1	$\bar{A}\cdot B\cdot C\cdot D$	$A+\bar{B}+\bar{C}+\bar{D}$	7
1	0	0	0	$A\cdot\bar{B}\cdot\bar{C}\cdot\bar{D}$	$\bar{A}+B+C+D$	8
1	0	0	1	$A\cdot\bar{B}\cdot\bar{C}\cdot D$	$\bar{A}+B+C+\bar{D}$	9
1	0	1	0	$A\cdot B\cdot C\cdot D$	$\bar{A}+B+\bar{C}+D$	10
1	0	1	1	$A\cdot\bar{B}\cdot C\cdot D$	$\bar{A}+B+\bar{C}+\bar{D}$	11
1	1	0	0	$A\cdot B\cdot\bar{C}\cdot\bar{D}$	$\bar{A}+\bar{B}+C+D$	12
1	1	0	1	$A\cdot B\cdot\bar{C}\cdot D$	$\bar{A}+\bar{B}+C+\bar{D}$	13
1	1	1	0	$A\cdot B\cdot C\cdot\bar{D}$	$\bar{A}+\bar{B}+\bar{C}+D$	14
1	1	1	1	$A\cdot B\cdot C\cdot D$	$\bar{A}+\bar{B}+\bar{C}+\bar{D}$	15

Fonte: Produção do próprio autor.

Para expressar a variável S do exemplo como uma função algébrica das variáveis A, B, C e D, inspeciona-se ordenadamente a coluna de saída da tabela-verdade. Deste modo, em (3) a função S está na forma de SOP e em (4) a função S está na forma de POS. Daqui em diante, por ser mais adequado à proposta, é utilizado apenas o formato SOP.

$$S = m_0 + m_1 + m_4 + m_5 + m_9 + m_{11} + m_{14} = \bar{A} \cdot \bar{B} \cdot \bar{C} \cdot \bar{D} + \bar{A} \cdot \bar{B} \cdot \bar{C} \cdot D + \bar{A} \cdot B \cdot \bar{C} \cdot \bar{D} + \bar{A} \cdot B \cdot \bar{C} \cdot D + A \cdot \bar{B} \cdot \bar{C} \cdot D + A \cdot \bar{B} \cdot C \cdot D + A \cdot B \cdot C \cdot \bar{D} \quad (3)$$

$$S = M_2 + M_3 + M_6 + M_7 + M_8 + M_{10} + M_{12} + M_{13} + M_{15} = (A + B + \bar{C} + D) \cdot (A + B + \bar{C} + \bar{D}) \cdot (A + \bar{B} + \bar{C} + D) \cdot (A + \bar{B} + \bar{C} + \bar{D}) \cdot (\bar{A} + B + C + D) \cdot (\bar{A} + B + \bar{C} + D) \cdot (\bar{A} + \bar{B} + C + D) \cdot (\bar{A} + \bar{B} + C + \bar{D}) \cdot (\bar{A} + \bar{B} + \bar{C} + \bar{D}) \quad (4)$$

2.3 MINIMIZAÇÃO DE FUNÇÕES BOOLEANAS E CIRCUITOS DIGITAIS

É muito comum que as variáveis booleanas sejam representadas por letras maiúsculas. Diferentemente do que ocorre nas funções reais, as funções booleanas em seu formato algébrico não contêm números booleanos explícitos, sendo formadas apenas por variáveis em operações de adição, multiplicação e inversão. Uma inspeção em (2) permite concluir que, embora tenha sido obtida uma representação algébrica, esta aparenta ser muito extensa para um sistema de pequeno porte. De fato, isso ocorre porque é natural que após a obtenção de uma função booleana como uma soma de mintermos, esta contenha redundâncias que tornam sua expressão mais extensa que o necessário. Assim, a função booleana está em seu formato *SOP* mais redundante, contendo elementos que são dispensáveis.

2.3.1 Mapa de Karnaugh e função minimizada

Uma função booleana algébrica está em sua forma mais minimizada quando contém a menor quantidade de elementos, condição visualmente observada pelo comprimento de sua representação. Uma função minimizada está vinculada às facilidades promovidas por sua utilização. Para tratar e eliminar as redundâncias presentes nas funções algébricas são utilizados os métodos de minimização como,

por exemplo, o *Mapa de Karnaugh*. Este método corresponde a uma estrutura geométrica na forma de quadro que representa uma função booleana mapeando seus valores e eliminando redundâncias (Karnaugh, 1953). Neste processo de minimização é utilizada a equivalência $XY + X\bar{Y} = X$, cuja demonstração utilizando os teoremas B13, B4 e B6, respectivamente, é feita em (5).

$$XY + X\bar{Y} = X(Y + \bar{Y}) = X \cdot 1 = X \tag{5}$$

Na estrutura retangular do mapa de Karnaugh, os valores da variável de saída provenientes da tabela-verdade ou dos mintermos da *SOP* não minimizada são dispostos de forma que quadrados vizinhos representem mintermos que variam em uma única posição, para que se possa aplicar o princípio mostrado em (5). Devem ser agrupados os números 1 que formem retângulos com todos os outros números 1 de sua vizinhança. A função minimizada será formada pela soma das parcelas simplificadas pelos retângulos, nos quais devem ser omitidas as variáveis que aparecem na forma original e inversa; as restantes devem ser mantidas na forma em que aparecem nos retângulos.

Na Figura 1 está representado o mapa de Karnaugh do sistema do exemplo proposto, de modo que obtém-se a função minimizada $S = \bar{A}\bar{C} + A\bar{B}D + ABC\bar{D}$. Como pode ser verificado, a nova expressão algébrica da função possui um formato bem mais simples que o da *SOP* inicial. Embora minimizada, continua sendo uma função algébrica no formato de soma de produtos das variáveis.

Figura 1 – Mapa de Karnaugh de um sistema de 4 variáveis.

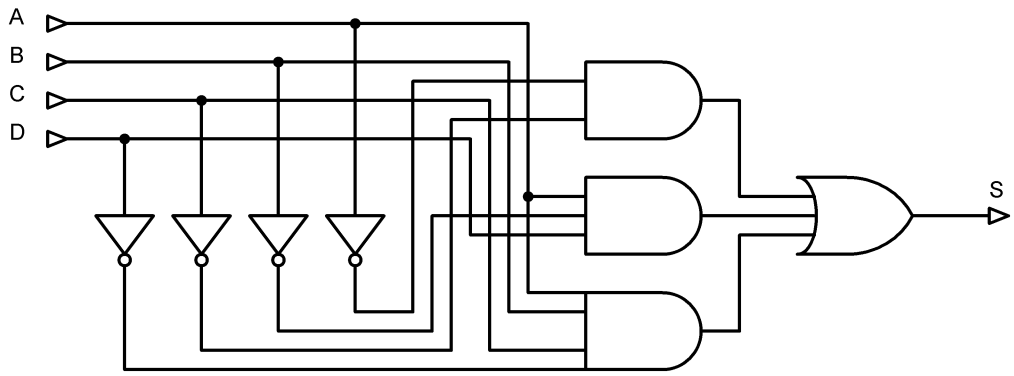
	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	1	1	0	0
$\bar{A}B$	1	1	0	0
AB	0	0	0	1
$A\bar{B}$	0	1	1	0

Fonte: Produção do próprio autor.

2.3.2 Circuito lógico combinacional

Uma vez obtida uma função booleana minimizada, é possível implementá-la por meio de um circuito em um sistema lógico digital, fazendo uso das portas lógicas. As portas lógicas implementam eletronicamente as operações algébricas booleanas. A porta lógica *OR* representa a operação de adição, a porta lógica *AND* representa a operação de multiplicação e a porta lógica *NOT* representa a operação de inversão. Se estas portas lógicas são utilizadas para implementar um circuito de modo que os valores das saídas dependam apenas dos valores das entradas, sem dependência temporal de memória ou realimentação, então este recebe o nome de circuito lógico combinacional (Sedra et al., 2020). Na Figura 2 está sendo representado o circuito lógico combinacional da função booleana S.

Figura 2 – Circuito lógico combinacional para a função S.



Fonte: Produção do próprio autor.

2.4 FORMA VETORIAL DE UMA FUNÇÃO BOOLEANA

Numa tabela-verdade que representa uma função booleana, a quantidade m de variáveis booleanas de entrada determina as 2^m linhas e, consequentemente, a quantidade 2^m dos 0's e 1's presentes na coluna de uma variável booleana que represente uma saída. Assim, cada variável de saída está associada a uma matriz coluna com 2^m elementos, compostos por 0's e 1's. Então, a matriz coluna ou coluna-verdade, correspondente a uma variável de saída, contém todas as informações relativas a essa função. Neste trabalho, essa matriz coluna ou sua correspondente matriz linha, transposta da primeira, é chamada de *forma vetorial*.

Portanto, a forma vetorial constitui uma maneira compacta de se representar uma função booleana, de modo que dispensa a extensão de uma tabela-verdade, pois contém intrinsecamente as mesmas informações. Para o sistema do exemplo, a forma vetorial da função S está representada pela matriz linha de (6) e foi obtida como a matriz transposta da matriz coluna da saída S na tabela-verdade. É utilizada a forma vetorial em coluna quando for conveniente.

$$S = [\,1\;1\;0\;0\;1\;1\;0\;0\;0\;1\;0\;1\;0\;0\;1\;0\,] \tag{6}$$

Os sistemas digitais lidam com estruturas que assumem, fundamentalmente, dois estados distintos. Um exemplo clássico é o de um sistema que pode estar ligado ou desligado. Deste modo, é natural e eficiente associar estes dois estados aos dois números booleanos 0 e 1. Esta unidade básica de informação corresponde a um dígito binário, nomeado de *bit (binary digit)* (Moss; Tocci; Widmer, 2018). Uma sequência de 8 bits é denominada *byte*. Um bit é a menor unidade de informação para transmissão e armazenamento de dados constitui uma base para toda a computação moderna.

3 LÓGICA MAJORITÁRIA

A noção e conceito a respeito de maioria não é recente. No que se refere à Língua Portuguesa, a palavra *maioria* faz referência à maior parte de algo ou a seu maior número e a palavra *majoritário* representa aquilo que se refere à maioria (Michaelis, 2002). Em um contexto matemático, a maioria é um subconjunto que contém uma quantidade de elementos superior à metade da quantidade do conjunto do qual faz parte.

Aqui será utilizada a noção de que uma estrutura majoritária contém múltiplas entradas e uma única saída, de modo que o valor da saída é o valor da maioria das entradas (Wigington, 1959). Para evitar os casos indeterminados, deve haver uma quantidade ímpar destas entradas. Neste capítulo são utilizadas as definições e métodos apresentados no Capítulo 2 para descrever o sistema majoritário como um subsistema da álgebra booleana.

3.1 FUNÇÃO MAIORIA DE 3 ENTRADAS

Da mesma forma, tanto no CI TC4530BP (Toshiba, 1985) quanto no CI MC14530B (Motorola, 1985), há um circuito eletrônico de transistores MOSFET que implementam um sistema que fornece na saída o valor que mais ocorre entre as 5 entradas. Aqui é analisado o sistema semelhante em que a saída é o valor que mais ocorre entre as 3 entradas, cuja tabela-verdade é representada na Tabela 10.

Tabela 10 – Sistema da maioria de 3 entradas.

X	Y	Z	M
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Fonte: Produção do próprio autor.

Com o intuito de obter-se uma função algébrica que relacione a saída M com as entradas X, Y e Z, tem-se a função SOP mostrada em (7), que não está em sua forma minimizada. Para minimizá-la aplica-se o mapa de Karnaugh a esta função, cuja representação está na Figura 3. Após a aplicação do mapa, obtém-se a função minimizada para M expressa por (8).

$$M = m_3 + m_5 + m_6 + m_7 = \bar{X}YZ + X\bar{Y}Z + XY\bar{Z} + XYZ \tag{7}$$

Figura 3 – Mapa de Karnaugh para a função maioria de 3 entradas.

	$\bar{Y}\bar{Z}$	$\bar{Y}Z$	YZ	$Y\bar{Z}$
$\bar{X}$	0	0	1	0
X	0	1	1	1

Fonte: Produção do próprio autor.

$$M = XY + XZ + YZ \tag{8}$$

Observando a coluna da variável M na tabela-verdade obtém-se a forma vetorial da função, correspondendo ao que está representado em (9). Conforme discutido na próxima subseção, está sendo avaliada uma função contendo 3 variáveis ou, numa análise física, informações provenientes de 3 trechos de um barramento. Quando este barramento contém mais elementos, um conjunto destas funções pode construir outras mais complexas.

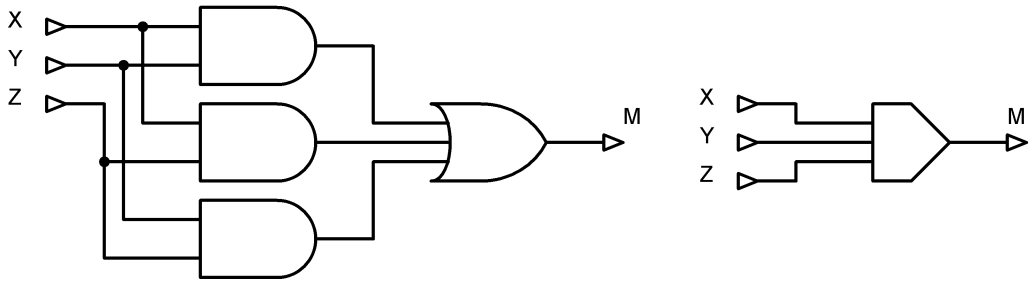
$$M = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} \qquad M = [\,0\;0\;0\;1\;0\;1\;1\;1\,] \tag{9}$$

3.1.1 Lógica majoritária e porta lógica majoritária

A função minimizada para a maioria M demonstra que no universo da tecnologia CMOS seriam necessárias 3 portas lógicas *AND* de 2 entradas e 1 porta lógica *OR* de 3 entradas para sua implementação. Porém, um advento primordial das novas tecnologias que se utilizam das funções maiorias é o fato de ser mais simples implementar um dispositivo físico único que forneça essa saída ao invés deste conjunto de portas convencionais. Este dispositivo físico que fornece a maioria de 3 entradas pode ser caracterizado como uma porta lógica majoritária. Na Figura 4 é estabelecida esta equivalência e apresentada a simbologia esquemática de uma porta lógica majoritária básica de 3 entradas.

Uma vez discutida a estrutura da função maioria, pode-se agora formalizar o sistema baseado neste princípio. Uma função lógica majoritária de 3 entradas e m variáveis é a função que associa à cada 3 números booleanos de entrada o número que mais ocorre dentre eles. O valor m refere-se à quantidade de variáveis disponíveis no sistema para ocupar as entradas, ou seja, a quantidade de variáveis presentes na tabela-verdade do sistema. Uma quantidade de variáveis maior que 3 implica em um sistema que permite a obtenção de funções mais simples interligadas para a formação de funções compostas, conforme é discutido no Capítulo 4. Portanto, qualquer saída terá uma quantidade 2^m de bits que irão compor sua forma vetorial.

Figura 4 – Equivalência entre as portas lógicas *CMOS* e a majoritária.



Fonte: Produção do próprio autor.

Para formalizar um pouco mais esta lógica, serão utilizados o resultado de (8) e a notação estabelecida por Chattopadhyay et al. (2016). Assim, em (10) está representada uma função majoritária M de 3 entradas X, Y e Z na notação algébrica mais utilizada na comunidade acadêmica. Esta equação, que aqui será chamada de *Relação Fundamental* desta lógica majoritária, devido sua importante e ampla

aplicabilidade, representa a estrutura matemática básica para a obtenção e representação de qualquer função majoritária de 3 entradas através de operações booleanas. Nesta função, M é a variável dependente das variáveis independentes X, Y e Z. Naturalmente, os números booleanos 0 e 1 podem ser entradas constantes de uma função majoritária e outras funções majoritárias mais complexas podem ser criadas fazendo a saída de uma ser a entrada de outra.

$$M(X,Y,Z) = XY + XZ + YZ \tag{10}$$

Esta relação fundamental representa a plataforma estrutural da teoria que é desenvolvida neste trabalho. Permite uma correspondência direta entre a notação majoritária e a notação booleana tradicional. Com ela é possível decodificar uma estrutura majoritária complexa em uma função booleana, ainda que extensa, mas composta apenas de adições, multiplicações e inversões. A inspeção criteriosa da relação fundamental permitiu todo o avanço da teoria e metodologia desenvolvida, que são apresentadas nos próximos capítulos.

De uma forma geral, a lógica majoritária de 3 entradas trata de interações relativas às maiorias dos bits destas entradas. E esta característica, em tese, permite projetar qualquer circuito de portas lógicas convencionais.

3.2 TEOREMAS DA LÓGICA MAJORITÁRIA

A lógica majoritária também é contemplada com diversos axiomas e teoremas, dos quais os que são necessários às discussões dos próximos capítulos estão na Tabela 11. Todos os 7 teoremas apresentados podem ser demonstrados pelo método de indução perfeita, porém também podem ser demonstrados utilizando a relação fundamental, que é utilizada para demonstrar os teoremas M_4 e M_5. As demonstrações são apresentadas em (11) e (12), respectivamente. Essas demonstrações ressaltam, novamente, a importância e o potencial da relação fundamental. Note que, após a decomposição da forma majoritária, são utilizados teoremas da álgebra booleana para obterem-se as relações finais.

$$M(0,X,Y) = 0 \cdot X + 0 \cdot Y + X \cdot Y = 0 + 0 + X \cdot Y = 0 + X \cdot Y = X \cdot Y \tag{11}$$

$$M(1,X,Y) = 1 \cdot X + 1 \cdot Y + X \cdot Y = X + Y \cdot (1 + Y) = X + Y \cdot 1 = X + Y \qquad (12)$$

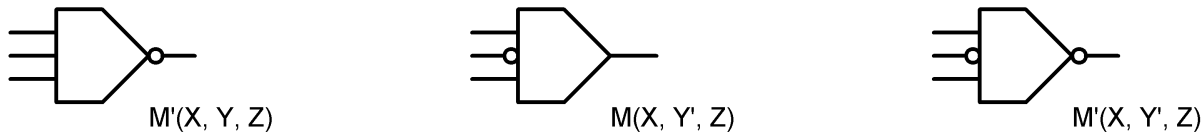
A correta interpretação e aplicação destes teoremas da lógica majoritária permitem expandir a porta lógica majoritária de 3 entradas básica em outras três variações, cujas simbologias esquemáticas são ilustradas na Figura 5. Observe que, em decorrência do teorema M_6, não se faz necessária a criação de uma porta lógica com duas ou três entradas inversas, as quais seriam equivalentes à primeira e à terceira variações, respectivamente.

Tabela 11 – Teoremas da lógica majoritária de 3 entradas.

REFERÊNCIA	TEOREMA
M_1	$M(X,Y,Z) = M(X,Z,Y) = \cdots = M(Z,Y,X)$
M_2	$M(X,X,Y) = X$
M_3	$M(X,\bar{X},Y) = Y$
M_4	$M(0,X,Y) = X \cdot Y$
M_5	$M(1,X,Y) = X + Y$
M_6	$\bar{M}(X,Y,Z) = M(\bar{X},\bar{Y},\bar{Z})$
M_7	$M(X,Y,M(Z,W,Y)) = M(Z,Y,M(X,W,Y))$

Fonte: Produção do próprio autor.

Figura 5 – Variações da porta lógica majoritária de 3 entradas.



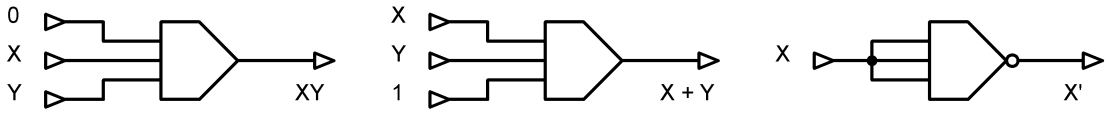
Fonte: Produção do próprio autor.

Os teoremas M_4 e M_5 permitem a implementação das operações algébricas booleanas de multiplicação e adição, respectivamente, utilizando a porta lógica majoritária. Uma aplicação conjunta dos teoremas M_6 e M_2 permite a implementação da operação algébrica booleana de inversão. Essas equivalências são apresentadas na Figura 6.

Uma vez que a porta lógica majoritária de 3 entradas pode ser utilizada para implementar as operações algébricas booleanas, espera-se que todo circuito lógico combinacional implementado com portas lógicas convencionais possa ser projetado

no sistema da lógica majoritária de 3 entradas. Isso comprova a versatilidade desta lógica, sendo capaz de reproduzir estruturas de outros sistemas.

Figura 6 – Operações booleanas implementadas com portas lógicas majoritárias.



Fonte: Produção do próprio autor.

3.3 FUNÇÕES MAJORITÁRIAS PRIMITIVAS

Uma *função majoritária primitiva* de 3 entradas e m variáveis ou, resumidamente, *primitiva*, é qualquer função em sua forma mais simplificada que, sendo implementada fisicamente, necessitaria de, no máximo, 1 única porta lógica majoritária de 3 entradas (Wang, et al., 2015). Para uma melhor compreensão, a função primitiva é qualquer função decorrente da aplicação nas 3 entradas de quaisquer constantes ou variáveis booleanas disponíveis no sistema. Como exemplo, na Tabela 12 são listadas todas as 40 funções majoritárias primitivas para $m = 3$.

Para uma certeza da obtenção de todas as funções primitivas, bastaria estabelecer uma ordem progressiva nos arranjos das 3 entradas e eliminar as várias repetições. Para o caso do exemplo, o primeiro arranjo seria $M(0,0,0)$ que corresponde à primitiva $P_1 = 0$, assim como os dois próximos arranjos $M(0,0,1)$ e $M(0,0,A)$. O último arranjo seria $M(\bar{C},\bar{C},\bar{C})$, que corresponde à primitiva $P_8 = \bar{C}$, primeiramente obtida no arranjo $M(0,1,\bar{C})$. A última primitiva é $P_{40} = M(\bar{A},\bar{B},\bar{C})$, também obtida, devido à comutatividade, nos arranjos $M(\bar{B},\bar{A},\bar{C})$ e $M(\bar{B},\bar{C},\bar{A})$

As 8 primeiras primitivas relativas a $m = 3$ dispensam o uso de uma porta lógica para sua implementação e por isso a definição informa o uso de, no máximo, 1 porta de 3 entradas. Embora a quantidade de primitivas seja única (Oliveira Júnior, 2021), seu formato algébrico não é. O uso do teorema M_6 permite obter formatos mais compactos como, por exemplo, nas primitivas $P_{16} = M(0,\bar{A},\bar{C}) = \overline{M}(1,A,C)$. Note que $P_{16} = \overline{M}(1,A,C) \neq M(1,A,C) = P_{22}$. A depender da tecnologia de implementação ou do custo vinculado à inversão, um formato pode ter mais relevância que o outro. Conforme discutido nos próximos capítulos, aqui é utilizada a porta com saída não inversa. As entradas, quando se fizer necessário, podem ser todas inversas.

Segundo seus idealizadores, uma função primitiva pode ser classificada em 5 grupos. No grupo *C (Constant)* estão as funções constantes 0 e 1. No grupo *S (Single)* estão todas as funções equivalentes a uma única variável, é o caso das funções A e B. No grupo *AND* estão as funções primitivas que representam o produto de duas variáveis, como por exemplo a função $M(0, A, B)$. No grupo *OR* estão as funções primitivas que representam a adição de duas variáveis, como por exemplo a função $M(1, A, C)$. Como último grupo, em *T (Three)* estão as funções primitivas que contém apenas variáveis, incluindo as inversas, como por exemplo a função $M(A, \bar{B}, C)$. Estas mesmas definições se aplicam às funções majoritárias com qualquer quantidade m de variáveis.

Tabela 12 – Funções primitivas para 3 variáveis.

ÍNDICE	FORMA MAJORITÁRIA	ÍNDICE	FORMA MAJORITÁRIA
P_1	0	P_{21}	$M(1, A, B)$
P_2	1	P_{22}	$M(1, A, C)$
P_3	A	P_{23}	$M(1, A, \bar{B})$
P_4	B	P_{24}	$M(1, A, \bar{C})$
P_5	C	P_{25}	$M(1, B, C)$
P_6	$\bar{A}$	P_{26}	$M(1, B, \bar{C})$
P_7	$\bar{B}$	P_{27}	$M(1, \bar{A}, B)$
P_8	$\bar{C}$	P_{28}	$M(1, \bar{A}, \bar{C})$
P_9	$M(0, A, B)$	P_{29}	$M(1, \bar{A}, \bar{B})$
P_{10}	$M(0, A, C)$	P_{30}	$M(1, \bar{A}, \bar{C})$
P_{11}	$M(0, A, \bar{B})$	P_{31}	$M(1, \bar{B}, C)$
P_{12}	$M(0, A, \bar{C})$	P_{32}	$M(1, \bar{B}, \bar{C})$
P_{13}	$M(0, B, C)$	P_{33}	$M(A, B, C)$
P_{14}	$M(0, B, \bar{C})$	P_{34}	$M(A, B, \bar{C})$
P_{15}	$M(0, \bar{A}, B)$	P_{35}	$M(A, \bar{B}, C)$
P_{16}	$M(0, \bar{A}, \bar{C})$	P_{36}	$M(A, \bar{B}, \bar{C})$
P_{17}	$M(0, \bar{A}, \bar{B})$	P_{37}	$M(\bar{A}, B, C)$
P_{18}	$M(0, \bar{A}, \bar{C})$	P_{38}	$M(\bar{A}, B, \bar{C})$
P_{19}	$M(0, \bar{B}, C)$	P_{39}	$M(\bar{A}, \bar{B}, C)$
P_{20}	$M(0, \bar{B}, \bar{C})$	P_{40}	$M(\bar{A}, \bar{B}, \bar{C})$

Fonte: Produção do próprio autor.

3.4 MINIMIZAÇÃO DE FUNÇÕES MAJORITÁRIAS

Considere o sistema arbitrário com tabela-verdade apresentada na Tabela 13. A função booleana não minimizada da função de saída $T = [\ 1\ 1\ 0\ 1\ ]$ está representada em (13). Utilizando os teoremas da álgebra booleana, considere as simplificações algébricas feitas de duas formas, conforme apresentado em (14) e (15). As duas equivalências obtidas não representam a melhor minimização.

Tabela 13 – Tabela-verdade arbitrária.

Y	X	T
0	0	1
0	1	1
1	0	0
1	1	1

Fonte: Produção do próprio autor.

$$T = m_0 + m_1 + m_3 = \bar{X}\bar{Y} + \mathrm{X}\bar{Y} + XY \tag{13}$$

$$T = \bar{X}\bar{Y} + X\bar{Y} + XY = \bar{Y} \cdot (\bar{X} + X) + XY = \bar{Y} \cdot 1 + XY = \bar{Y} + XY \tag{14}$$

$$T = \bar{X}\bar{Y} + X\bar{Y} + XY = \bar{X}\bar{Y} + \mathrm{X} \cdot (\bar{Y} + Y) = \bar{X}\bar{Y} + \mathrm{X} \cdot 1 = \bar{X}\bar{Y} + X \tag{15}$$

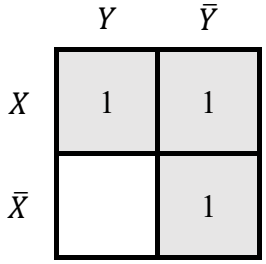
Porém, quando se aplica o Mapa de Karnaugh à função Z, conforme é ilustrado na Figura 7, obtém-se a função equivalente $T = X + \bar{Y}$, mais minimizada que aquelas obtidas pela simplificação algébrica primária. Quando o mapa de Karnaugh é empregado corretamente, fornece a função booleana mais minimizada, pois identifica com facilidade as simplificações que não são tão óbvias quando executadas algebricamente.

3.4.1 A problemática da minimização majoritária

Como visto no exemplo anterior, minimizar uma função não é uma tarefa rudimentar, até mesmo quando se trata de uma função simples. Portanto, são

necessários o desenvolvimento de novos métodos de minimização e o aprimoramento daqueles existentes.

Figura 7 – Mapa de Karnaugh.



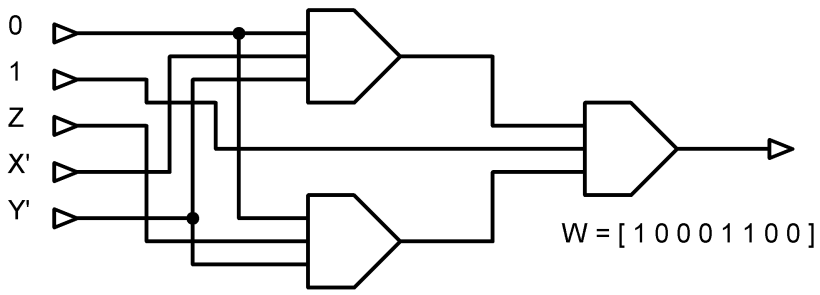
Fonte: Produção do próprio autor.

3.4.2 A problemática da minimização majoritária

Como visto no exemplo anterior, minimizar uma função não é uma tarefa rudimentar, até mesmo quando se trata de uma função simples. Portanto, são necessários o desenvolvimento de novos métodos de minimização e o aprimoramento daqueles existentes.

Considere agora a função arbitrária $W = [\,1\,0\,0\,0\,1\,1\,0\,0\,]$ de 3 variáveis X, Y e Z. Esta função na forma mais minimizada após o uso do Mapa de Karnaugh fornece a equivalência $W = \bar{X}\bar{Y} + \bar{Y}Z$. Esta função contém dois produtos e uma adição, de modo que pode ser facilmente implementada com 3 portas lógicas majoritárias, resultando no circuito apresentado na Figura 8. No entanto, embora implemente uma função simples, esta forma majoritária não está minimizada.

Figura 8 – Circuito lógico majoritário combinacional não minimizado.

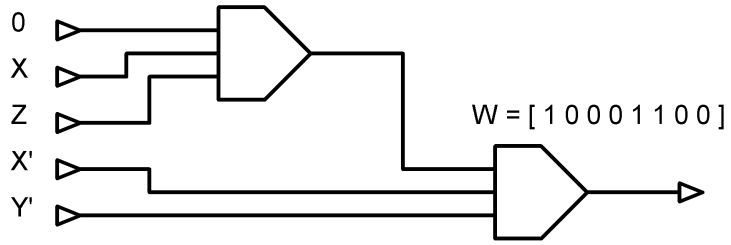


Fonte: Produção do próprio autor.

Observe que o circuito majoritário da Figura 8 representa a função lógica majoritária $W = M(1, M(0, \bar{X}, \bar{Y}), M(0, Z, \bar{Y}))$, de modo que é constituída por 3 portas lógicas majoritárias. Utilizando o método *MLP01*, obtém-se a função lógica majoritária minimizada $W = M(\bar{X}, \bar{Y}, M(0, X, Z))$. Nesta outra representação, há o emprego de apenas 2 portas lógicas majoritárias. O circuito combinacional para essa outra representação está representado na Figura 9. Claramente, o projeto do segundo circuito tem um custo menor de implementação.

Portanto, a minimização de funções majoritárias representa um grande desafio a ser superado. Aqui, o processo de minimização de uma função majoritária é tratado como sendo uma tentativa de obter-se a função mais minimizada. No próximo capítulo é desenvolvido o início da apresentação de uma nova teoria para este processo de minimização.

Figura 9 – Circuito lógico majoritário combinacional minimizado.



Fonte: Produção do próprio autor.

4 FUNÇÕES BOOLEANAS FUNDAMENTAIS

A noção de função primitiva se popularizou entre as linhas de pesquisa sobre métodos de simplificação de funções majoritárias. Seu emprego ganhou destaque nos métodos propostos por Soeken et al. (2017) e Chu et al. (2019). Ainda segundo seus idealizadores, as funções primitivas representam os elementos básicos na formação de qualquer função majoritária. Deste modo, obter uma função majoritária seria determinar como estas funções primitivas estão distribuídas na estrutura majoritária. E obter a função majoritária minimizada seria determinar o posicionamento mais eficiente das funções primitivas dentro da estrutura majoritária.

Em oposição a esta tendência, neste capítulo é introduzido e discutido um dos alicerces que compõem a linha de pesquisa que é apresentada neste trabalho. Assim, propõe-se não mais utilizar o artifício das funções primitivas, dando ênfase à ideia de função booleana fundamental.

4.1 FUNÇÕES BOOLEANAS DE VÁRIAS VARIÁVEIS

No universo do conjunto dos números reais, uma função real de 1 única variável real representa uma regra de associação na qual fica estabelecida que, a cada número real x, está associado um outro número real $f(x)$, de modo que nunca há valores distintos de $f(x)$ para um mesmo valor de x (Leithold, 1994). O conjunto de todos os valores possíveis para a variável x na função recebe o nome de Domínio, que será representado por $Dom(f)$, e o conjunto de todos os valores possíveis para $f(x)$ recebe o nome de Imagem, que será representado por $Im(f)$. Assim, $f: \mathbb{R} \to \mathbb{R}$ representa uma função real f de $\mathbb{R}$ em $\mathbb{R}$, na qual $Dom(f) \subset \mathbb{R}$ e $Im(f) \subset \mathbb{R}$.

Funções reais de 1 única variável real podem ser operadas matematicamente como se operam os próprios números reais. No entanto há uma operação exclusiva entre essas funções, chamada composição. Considerando duas funções reais de 1 uma única variável real, f_1 e f_2, então a função composta de f_1 com f_2, representada por $f_1 \circ f_2$, pode ser definida como sendo a função que associa a cada número real x do $Dom(f_2)$ o número real $(f_1 \circ f_2)(x) = f_1(f_2(x))$ da $Im(f_1)$ (Thomas; Weir; Hass, 2012). Assim, para um valor real de x, o resultado real de $f_2(x)$ é utilizado como o número real que é aplicado em $f_1(x)$. A operação de composição é exemplificada em (16).

$$\begin{cases} f_1(x) = x^2 \\ f_2(x) = 2x + 1 \end{cases} \Rightarrow (f_1 \circ f_2)(x) = f_1(f_2(x)) = (2x+1)^2 = 4x^2 + 4x + 1 \qquad (16)$$

4.1.1 Funções reais de várias variáveis reais

Uma função real de n variáveis reais é uma regra de associação na qual fica estabelecida que, a cada n-upla de números reais $(x_1, x_2, \dots, x_n)$ está associado um outro número real $f(x_1, x_2, \dots, x_n)$ (Stewart, 2013). De forma semelhante ao definido para uma função real de 1 única variável real, o conjunto de todas as n-uplas reais possíveis para a aplicação na função é o seu Domínio e o conjunto de todos os valores reais possíveis para f recebe o nome de Imagem. Assim, tem-se $f\colon \mathbb{R}^n \to \mathbb{R}$, em que $Dom(f) \subset \mathbb{R}^n$ e $Im(f) \subset \mathbb{R}$.

Nas funções reais de várias variáveis reais também é possível estender a noção de composição de funções. Considerando as funções $f, f_1, f_2, \dots, f_p$ de n variáveis reais, em que $p \leq n$, então a função composta de f com $f_1, f_2, \dots, f_p$, representada por $f \circ (f_1, f_2, \dots, f_p)$, pode ser definida como sendo a função que associa a cada n-upla real $(x_1, x_2, \dots, x_n)$ o número real $(f \circ (f_1, f_2, \dots, f_p))(x_1, x_2, \dots, x_n)$. Assim, uma n-upla $(x_1, x_2, \dots, x_n)$ produz a p-upla $(f_1, f_2, \dots, f_p)$ sendo que cada função f_i atua na variável x_i correspondente de f. Para exemplificar, em (17) está representado o processo de composição de uma função de 4 variáveis. Observe que, ainda que a função composta seja independente da variável x_4, continua sendo uma função das 4 variáveis x_1, x_2, x_3 e x_4.

$$\begin{cases} f(x_1, x_2, x_3, x_4) = 2x_1 + x_2 + 4x_3 - 10x_4 \\ f_1(x_1, x_2, x_3, x_4) = x_1 + x_2 \\ f_2(x_1, x_2, x_3, x_4) = x_2 \\ f_4(x_1, x_2, x_3, x_4) = 2x_3 \end{cases} \Rightarrow (f \circ (f_1, f_2, f_4))(x_1, x_2, x_3, x_4) =$$

$$= 2(x_1 + x_2) + x_2 + 4x_3 - 10(2x_3) = 2x_1 + 3x_2 - 16x_3 \qquad (17)$$

4.1.2 Funções booleanas de várias variáveis booleanas

É possível fazer uma analogia entre o conceito de função real de várias variáveis reais e o conceito de funções em $\mathbb{B}$. Uma função booleana f de m variáveis booleanas

é uma regra de associação na qual fica estabelecida que, a cada m-upla de números booleanos $(B_1, B_2, \dots, B_m)$, está associado um número booleano $f(B_1, B_2, \dots, B_m)$, em que B_i é uma variável booleana genérica dentre as m. Uma função booleana f de várias variáveis booleanas possui um Domínio $Dom(f) \subset \mathbb{B}^m$ e um conjunto Imagem $Im(f) \subset \mathbb{B}$. Por conveniência e organização, neste trabalho as variáveis booleanas são representadas por letras maiúsculas e incluídas em ordem alfabética como, por exemplo, a função $f(A,B) = A + \bar{B}$, que representa uma função booleana de 2 variáveis booleanas. Numa aplicação desta função tem-se que, se $A = 1$ e $B = 0$, então $f(A,B) = f(1,0) = 1 + \bar{0} = 1 + 1 = 1$.

Em decorrência das definições apresentadas, é notório que todas as funções majoritárias M de 3 entradas e m variáveis representam funções booleanas de m variáveis booleanas, sendo $Dom(M) \subset \mathbb{B}^m$ e $Im(M) \subset \mathbb{B}$. Exemplificando, considere a função de 5 variáveis $f = [\,0\,1\,1\,1\,0\,1\,1\,1\,0\,0\,0\,1\,0\,0\,0\,1\,0\,1\,1\,1\,0\,1\,1\,1\,0\,0\,0\,1\,0\,0\,0\,1\,]$, que é uma das primitivas *T* para $m = 5$. A Equação 18 representa f em sua forma majoritária e expandida pela relação fundamental. Embora em sua expressão algébrica haja apenas as variáveis A, B e D, tem-se $Dom(f) = \mathbb{B}^5$ e as variáveis C e E, que não compõem a expressão algébrica da função, estão implicitamente anuladas na mesma.

$$f(A,B,C,D,E) = M(A,\bar{B},D) = A\bar{B} + AD + \bar{B}D = A\bar{B} + AD + \bar{B}D + 0 \cdot C + 0 \cdot E \qquad (18)$$

4.2 FUNÇÕES BOOLEANAS COMPOSTAS

Para as funções booleanas de várias variáveis booleanas também se pode aplicar o conceito de composição de funções. Considere as funções booleanas f, f_1, f_2, ... e f_p de m variáveis booleanas, com $p \leq m$, em que os índices $i \leq p$ referem-se aos índices das variáveis B_i. A função composta de f com f_1, f_2, ... e f_p, representada por $f \circ (f_1, f_2, \dots, f_p)$, pode ser definida como sendo a função que associa cada m-upla de números booleanos $(B_1, B_2, \dots, B_m)$ a um outro número booleano, representado por $(f \circ (f_1, f_2, \dots, f_p))(B_1, B_2, \dots, B_m) = f(f_1(B_1, B_2, \dots, B_m), \dots, f_p(B_1, B_2, \dots, B_m))$. Portanto, a função composta faz com que para cada m-upla do $Dom(M) \subset \mathbb{B}^m$, os resultados de $f_1(B_1, B_2, \dots, B_m)$, $f_2(B_1, B_2, \dots, B_m)$, ... e $f_p(B_1, B_2, \dots, B_m)$ sejam aplicados,

ordenadamente, às variáveis correspondentes de f. Como exemplo, em (19) está representado o processo de composição de funções de 5 variáveis.

$$\begin{cases} f(A,B,C,D,E) = A + \bar{B} + AD \\ f_1(A,B,C,D,E) = B \\ f_2(A,B,C,D,E) = 1 \\ f_4(A,B,C,D,E) = AE \end{cases} \Rightarrow (f \circ (f_1, f_2, f_4))(A,B,C,D,E) = f_1(A,B,C,D,E) +$$

$$\bar{f_2}(A,B,C,D,E) + f_1(A,B,C,D,E) \cdot f_4(A,B,C,D,E) = B + \bar{1} + BAE = B + ABE \qquad (19)$$

É muito importante respeitar a ordenação das funções neste processo de composição. Com base na ideia de composição, as funções majoritárias podem ser interpretadas como sendo formadas, a depender de sua complexidade, por várias composições de funções. Assim, por exemplo, a função $M(A, M(1,B,C), M(1,\bar{B},\bar{C}))$, que representa a função $[\,0\;1\;1\;0\;1\;1\;1\;1\,]$ de $m = 3$ variáveis, pode ser interpretada como sendo a composição da função $f(A,B,C) = M(A,B,C)$ com as funções $f_2(A,B,C) = M(1,B,C)$ e $f_3(A,B,C) = M(1,\bar{B},\bar{C})$.

4.2.1 Funções booleanas compostas e as primitivas

Cada uma de todas as m variáveis e todas as m variáveis inversas contidas numa função booleana recebem o nome de *literal* (Maini, 2007). Assim, numa função majoritária as entradas podem ser as constantes booleanas, os literais e outras funções majoritárias, estabelecendo relações de composição. No desenvolvimento da teoria que é apresentada à frente, uma função genérica é representada pela letra minúscula f para que, inclusive, não conflite com a representação da variável F.

Chama-se *nível* cada camada onde, ao menos uma porta lógica, recebe como entrada a saída de outra, em um nível inferior. Por exemplo, a função de $m = 4$ variáveis $[\,0\;1\;0\;1\;0\;1\;0\;1\;0\;0\;0\;0\;0\;0\;0\;0\,] = M(0,A,\bar{D})$ possui 1 nível, a função de $m = 3$ variáveis $[\,0\;0\;1\;0\;0\;1\;0\;1\,] = M(M(0,A,C), M(0,\bar{A},\bar{C}), M(1,B,C))$ possui 2 níveis, e assim por diante. Assim, funções majoritárias com mais de 1 nível podem ser interpretadas como sendo obtidas pela composição da função majoritária básica $M(A,B,C)$ com as funções constantes, literais e primitivas de modo que ao final, na equivalência booleana, a função continua a depender apenas dos literais e das

constantes booleanas. Portanto, o uso de funções primitivas não é necessário para a determinação de uma função majoritária simplificada. O conjunto de primitivas constitui um banco de dados armazenado, para que se inicie um processo de minimização apenas para funções com mais de 1 nível.

Numa comparação, considere uma calculadora que para efetuar pequenas adições, ao invés de executar o processo de adição processual, buscasse em sua memória estes pequenos resultados previamente armazenados. A quantidade de primitivas para m variáveis é expressa pelo polinômio de (20) (Oliveira Júnior, 2021). Na Tabela 14 estão listadas as quantidades de primitivas até $m = 10$ variáveis.

$$p(m) = \frac{4}{3}m^3 + \frac{2}{3}m + 2 \tag{20}$$

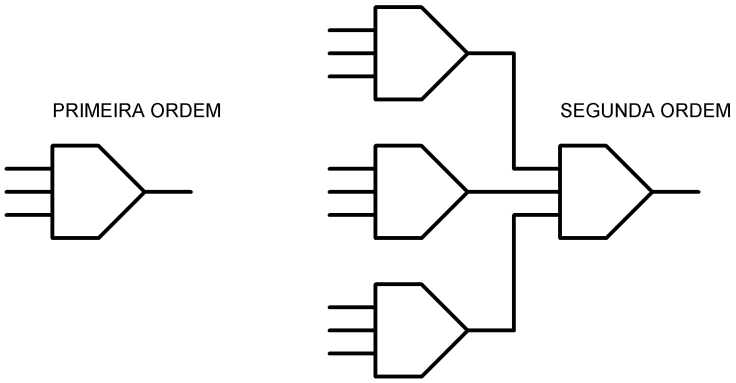
Tabela 14 – Quantidade de primitivas para m variáveis.

VARIÁVEIS	PRIMITIVAS
3	40
4	90
5	172
6	294
7	464
8	690
9	980
10	1342

Fonte: Produção do próprio autor.

Nos métodos de minimização de funções majoritárias que se utilizam das primitivas, devem ser criados bancos de dados para armazená-las. Deste modo, o problema de minimização de funções com 1 nível está solucionado. Um raciocínio semelhante poderia ser utilizado para resolver o problema de funções de 2 níveis. Se for feita uma listagem exaustiva com todas as funções majoritárias de 2 níveis, excluindo-se as repetições, seriam obtidos novos bancos de dados contendo centenas de milhares ou milhões de funções, a depender de m. Assim, seriam criadas classes de primitivas de segunda ordem, conforme é representado na Figura 10. Essa analogia poderia se estender na obtenção de primitivas de outras ordens.

Figura 10 – Conceito de primitivas de outras ordens.



Fonte: Produção do próprio autor.

4.3 FUNÇÃO BOOLEANA FUNDAMENTAL

Na seção anterior, foi discutido que as funções majoritárias primitivas não devem constituir uma necessidade no problema da obtenção de uma função majoritária simplificada e como poderia ser onerosa a obtenção e o armazenamento destes dados. Além disso, o mecanismo de primitivas pode parecer uma metodologia de tentativa e erro, o que não seria adequado. Deve-se atentar ao fato de que as 3 entradas das funções majoritárias, aliadas ao seu modo de operação, representam um mecanismo de processamento das variáveis, não sendo imperativo que todas façam parte da expressão algébrica minimizada. Também, deve-se observar que é a operação de composição das funções majoritárias que permite obter a variabilidade em sua extensa quantidade. Ao final, obter uma função majoritária minimizada é conhecer a disposição de constantes e literais em sua arquitetura.

Como é mostrado no Capítulo 5, o uso de funções primitivas dificultaria o desenvolvimento do método a ser apresentado. A vertente que se defende nesta proposta é a de que, de fato, se deva encontrar as possíveis posições das 2 funções constantes, 0 e 1, e dos $2m$ literais. Portanto, conclui-se que o uso do conceito de primitivas não é obrigatório em uma metodologia de minimização, mas sim das constantes e literais. Para excluir o uso das primitivas, não chamá-las de constantes e literais, e unificar os grupos *C* e *S*, é criado um novo conceito. *Função Booleana Fundamental* (FBF) de um sistema com m variáveis é qualquer uma das $2m + 2$ funções representadas pelas funções constantes e funções literais.

Para o conjunto de todas as $2m+2$ funções booleanas fundamentais é utilizada a nomenclatura f^m. Estas funções são indexadas, de modo que a i-ésima função seja representada por f_i. São ordenadas de $f_1 = 0$ até f_{2m+2}, seguindo a ordem alfabética das variáveis, seguidas de suas inversas.

Exemplificando, nas Tabelas 15 e 16 estão listadas as funções booleanas fundamentais relativas a sistemas de 3 e 4 variáveis, respectivamente. Assim, tem-se $f^3 = \{0, A, B, C, \bar{A}, \bar{B}, \bar{C}, 1\}$ e $f^4 = \{0, A, B, C, D, \bar{A}, \bar{B}, \bar{C}, \bar{D}, 1\}$. Neste ponto, é adequado enunciar o primeiro de vários lemas que conduziram o desenvolvimento da teoria apresentada neste trabalho. O lema 1 representa o princípio para a proposta do próximo capítulo.

▪ *Lema 1:* Uma função majoritária minimizada é função das funções booleanas fundamentais.

Tabela 15 – Funções booleanas fundamentais para 3 variáveis.

FORMA ALGÉBRICA	FORMA VETORIAL
0	$[\,0\,0\,0\,0\,0\,0\,0\,0\,]$
A	$[\,0\,1\,0\,1\,0\,1\,0\,1\,]$
B	$[\,0\,0\,1\,1\,0\,0\,1\,1\,]$
C	$[\,0\,0\,0\,0\,1\,1\,1\,1\,]$
$\bar{A}$	$[\,1\,0\,1\,0\,1\,0\,1\,0\,]$
$\bar{B}$	$[\,1\,1\,0\,0\,1\,1\,0\,0\,]$
$\bar{C}$	$[\,1\,1\,1\,1\,0\,0\,0\,0\,]$
1	$[\,1\,1\,1\,1\,1\,1\,1\,1\,]$

Fonte: Produção do próprio autor.

Tabela 16 – Funções booleanas fundamentais para 4 variáveis.

FORMA ALGÉBRICA	FORMA VETORIAL
0	$[\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,]$
A	$[\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,]$
B	$[\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,]$
C	$[\,0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,1\,1\,]$
D	$[\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,]$
$\bar{A}$	$[\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,]$
$\bar{B}$	$[\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,]$
$\bar{C}$	$[\,1\,1\,1\,1\,0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,]$
$\bar{D}$	$[\,1\,1\,1\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,]$
1	$[\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,]$

Fonte: Produção do próprio autor.

5 A PROPOSTA DE UMA NOVA METODOLOGIA

Deve ser um preceito básico de uma teoria ou método da obtenção de funções majoritárias minimizadas que a função obtida seja a mais compacta ou otimizada possível. Isso significa que à função majoritária obtida esteja associada uma mínima quantidade de portas lógicas. Minimizar uma função em sua forma majoritária de 3 entradas trata de encontrar o posicionamento adequado das FBF dentro da topologia da função de modo a produzir o formato mais simples.

Considere, por exemplo, a função de 3 variáveis $f = [\,0\,1\,0\,1\,0\,1\,0\,1\,] = A$, ou seja, f é uma das FBF. Embora $f = A$ seja o formato mais minimizado, f pode ser expressa de outras formas mais complexas e menos otimizadas. Em (21) são apresentadas 3 equivalências para essa função. Na primeira equivalência seriam necessárias 3 portas lógicas majoritárias em 2 níveis para a implementação de f, na segunda seriam necessárias 2 portas em 2 níveis e na terceira seria necessária 1 porta em 1 nível. Ainda assim, em todas as 3 equivalências não foi atingida a maior minimização. Logo, sempre é possível, por meio de redundâncias, representar uma função majoritária em um formato mais complexo e menos otimizado.

$$M(A, M(0, A, B), M(C, \bar{B}, 1)) = M(B, \bar{B}, M(A, C, \bar{C})) = M(0, 1, A) = A \qquad (21)$$

5.1 CLASSES DE COMPLEXIDADE DAS FUNÇÕES MAJORITÁRIAS

Numa estrutura majoritária de 3 entradas é a topologia da arquitetura das funções e a distribuição das FBF que determinam a natureza de sua complexidade. À medida que as entradas sejam outras funções majoritárias, pelo mecanismo de composição, as possibilidades atingem um limite até ser necessária a adição de outro nível, de modo que o processo pode ser repetido. Teoricamente, este processo pode ser repetido ilimitadamente, dando origem a estruturas de alta complexidade. Neste contexto, propõe-se o conceito de *Classe de Complexidade de uma Função Majoritária* (CCFM), que é medida pela variável c.

Portanto, c é a CCFM da função f, sendo um número inteiro não negativo e crescente, a partir de 0, com a complexidade da função. Utilizando a CCFM, pode-se mostrar que a cada valor de c estão associados uma arquitetura de função majoritária

e uma topologia de circuito lógico muito bem determinados. Para exemplificar sua aplicação, serão analisadas algumas destas classes.

5.1.1 As primeiras classes de complexidade

Para cada classe descrita será incluída a apresentação de um circuito lógico esquemático correspondente à sua topologia, para que haja uma interpretação física adequada do conceito de CCFM. Em seguida, é citado um exemplo de função com 4 variáveis. Para cada circuito, está implícito um barramento de dados fornecendo os $2m + 2$ sinais possíveis correspondentes às FBF.

▪ $CCFM = 0$: Representada por todas as FBF, correspondendo a qualquer função com o formato $f = X_1$. Obtém-se essa função recebendo um único sinal de informação diretamente do barramento de dados, dispensando o uso de porta lógica majoritária. Em (22) apresenta-se um exemplo para esta classe de funções e na Figura 11.a está representado o circuito lógico combinacional correspondente.

$$f = [\,0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,1\,1\,1\,1\,] = C \tag{22}$$

Figura 11 – Topologia de circuito majoritário para $c = 0$ e $c = 1$.

Fonte: Produção do próprio autor.

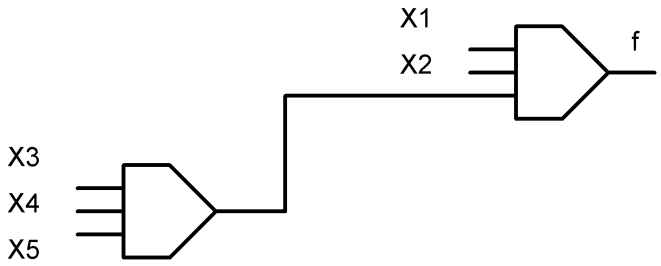
▪ $CCFM = 1$: Corresponde às funções com o formato $f = M(X_1, X_2, X_3)$. São compostas por 1 única porta lógica majoritária em 1 único nível. Há 3 entradas recebendo sinais de informação do barramento de dados. Em (23) apresenta-se um exemplo para esta classe de funções e na Figura 11.b está representado o circuito lógico combinacional correspondente.

$$f = [\,0\,1\,0\,1\,0\,1\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,] = M(0, A, \overline{D}) \tag{23}$$

▪ $CCFM = 2$: Relativa às funções com o formato $f = M(X_1, X_2, M(X_3, X_4, X_5))$. São compostas por 2 portas lógicas majoritárias ocupando 2 níveis. Há 5 entradas recebendo sinais de informação do barramento de dados. Em (24) apresenta-se um exemplo para esta classe de funções e na Figura 12 está representado o circuito lógico combinacional correspondente.

$$f = [\,1\,0\,1\,0\,0\,0\,1\,0\,1\,1\,1\,1\,1\,0\,1\,1\,] = M(A, \overline{D}, M(C, \overline{B}, 1)) \quad (24)$$

Figura 12 – Topologia de circuito majoritário para $c = 2$.

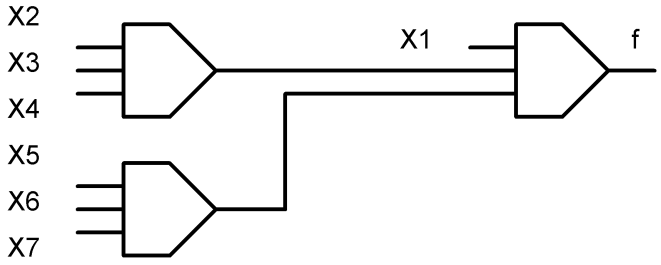


Fonte: Produção do próprio autor.

▪ $CCFM = 3$: São compostas por 2 portas lógicas majoritárias que ocupam 2 níveis. Representa funções com o formato $f = M(X_1, M(X_2, X_3, X_4), M(X_5, X_6, X_7))$. Há 7 entradas recebendo sinais de informação do barramento de dados. Em (25) apresenta-se um exemplo para esta classe de funções e na Figura 13 está representado o circuito lógico combinacional correspondente.

$$f = [\,1\,0\,1\,0\,0\,0\,1\,0\,1\,1\,1\,1\,0\,0\,1\,0\,] = M(\overline{B}, M(0, C, \overline{D}), M(A, \overline{D}, 1)) \quad (25)$$

Figura 13 – Topologia de circuito majoritário para $c = 3$.

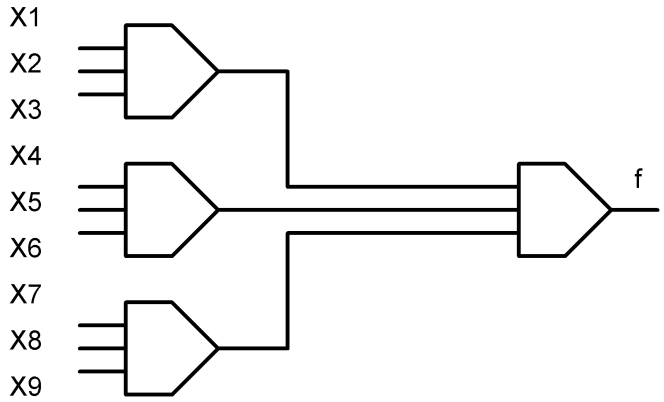


Fonte: Produção do próprio autor.

▪ $CCFM = 4$: São compostas por 4 portas lógicas majoritárias que ocupam 2 níveis. As funções com o formato $f = M(M(X_1, X_2, X_3), M(X_4, X_5, X_6), M(X_7, X_8, X_9))$ representam esta classe. Há 9 entradas recebendo sinais de informação do barramento de dados. Em (26) apresenta-se um exemplo para esta classe de funções e na Figura 14 está representado o circuito lógico combinacional correspondente.

$$f = [\,0\,0\,1\,1\,1\,0\,1\,1\,0\,0\,0\,0\,1\,0\,0\,0\,] = M((0, C, \bar{A}), M(0, \bar{C}, \bar{D}), M(B, C, 1)) \qquad (26)$$

Figura 14 – Topologia de circuito majoritário para $c = 4$.



Fonte: Produção do próprio autor.

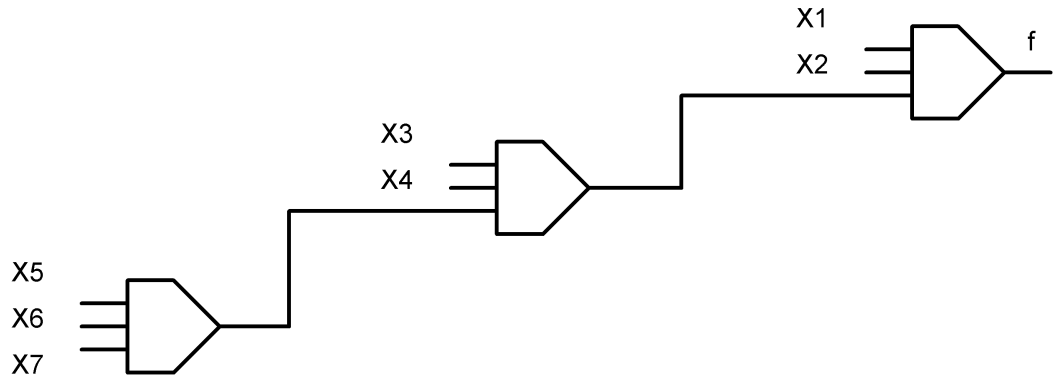
▪ $CCFM = 5$: São compostas por 3 portas lógicas majoritárias que ocupam 3 níveis. As funções com o formato $f = M(X_1, X_2, M(X_3, X_4, M(X_5, X_6, X_7)))$ representam esta classe. Há 7 entradas recebendo sinais de informação do barramento de dados. Em (27) apresenta-se um exemplo para esta classe de funções e na Figura 15 está sendo representado o circuito lógico combinacional correspondente.

$$f = [\,0\,0\,0\,1\,0\,0\,0\,1\,0\,0\,0\,1\,0\,1\,0\,1\,] = M(0, A, M(B, 1, M(C, D, \bar{A}))) \qquad (27)$$

▪ $CCFM = 6$: São compostas por 4 portas lógicas majoritárias, com 1 no nível 1, 1 no nível 2 e 2 no nível 3. Essas funções possuem um formato do tipo $f = M(X_1, X_2, M(X_3, M(X_4, X_5, X_6), M(X_7, X_8, X_9)))$. Há 9 entradas recebendo sinais de informação do barramento de dados. Em (28) apresenta-se um exemplo para esta classe de funções e na Figura 16 está sendo representado o circuito lógico combinacional correspondente.

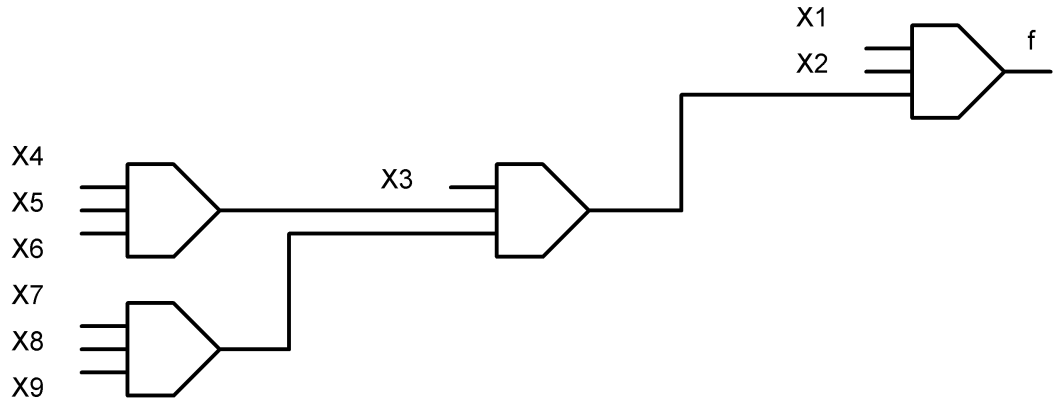
$$f = [\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,1\,0\,1\,] = M(0, A, M(C, M(B, D, 1), M(0, B, \bar{A}))) \quad (28)$$

Figura 15 – Topologia de circuito majoritário para $c = 5$.



Fonte: Produção do próprio autor.

Figura 16 – Topologia de circuito majoritário para $c = 6$.

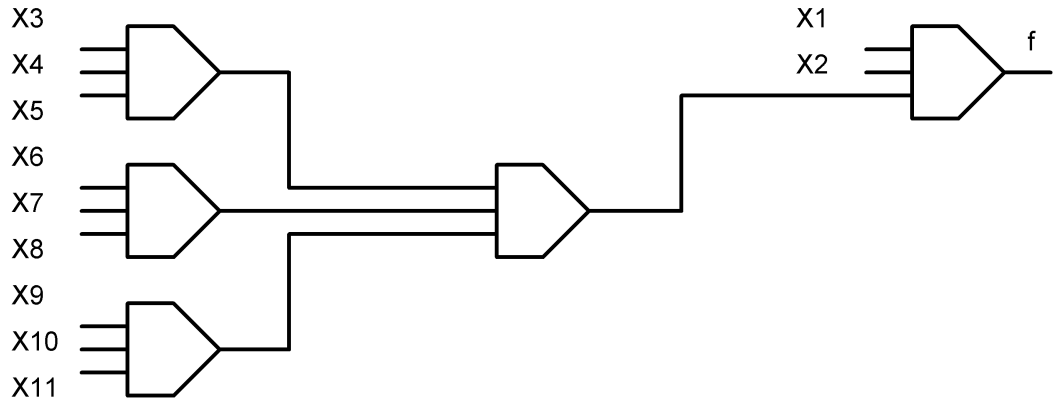


Fonte: Produção do próprio autor.

▪ $CCFM = 7$: São compostas por 5 portas lógicas majoritárias, com 1 no nível 1, 1 no nível 2 e 3 no nível 3. Essas funções possuem um formato do tipo $f = M(X_1, X_2, M(M(X_3, X_4, X_5), M(X_6, X_7, X_8), M(X_9, X_{10}, X_{11})))$. Há 11 entradas recebendo sinais de informação do barramento de dados. Em (29) apresenta-se um exemplo para esta classe de funções e na Figura 17 está sendo representado o circuito lógico combinacional correspondente.

$$f = [\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,] = M(A, B, M(M(0, C, D), M(B, \bar{A}, 1), M(A, B, \bar{D}))) \quad (29)$$

Figura 17 – Topologia de circuito majoritário para $c = 7$.

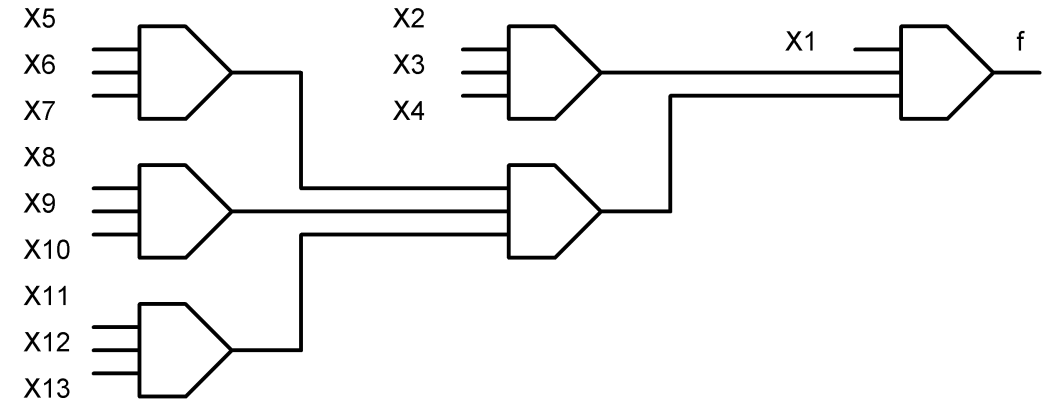


Fonte: Produção do próprio autor.

▪ $CCFM = 8$: São compostas por 6 portas lógicas majoritárias, com 1 no nível 1, 2 no nível 2 e 3 no nível 3. Essas funções possuem um formato do tipo $f = M(X_1, M(X_2, X_3, X_4), M(M(X_5, X_6, X_7), M(X_8, X_9, X_{10}), M(X_{11}, X_{12}, X_{13})))$. Há 13 entradas recebendo sinais de informação do barramento de dados. Em (30) apresenta-se um exemplo para esta classe de funções e na Figura 18 está sendo representado o circuito lógico combinacional correspondente.

$$f = [\,0\,0\,0\,0\,0\,1\,0\,0\,0\,1\,0\,0\,0\,1\,0\,0\,\,] =$$
$$= M(0, M(0, A, \bar{B}), M(M(C, D, 1), M(A, B, C), M(A, \bar{B}, \bar{D}))) \quad (30)$$

Figura 18 – Topologia de circuito majoritário para $c = 8$.

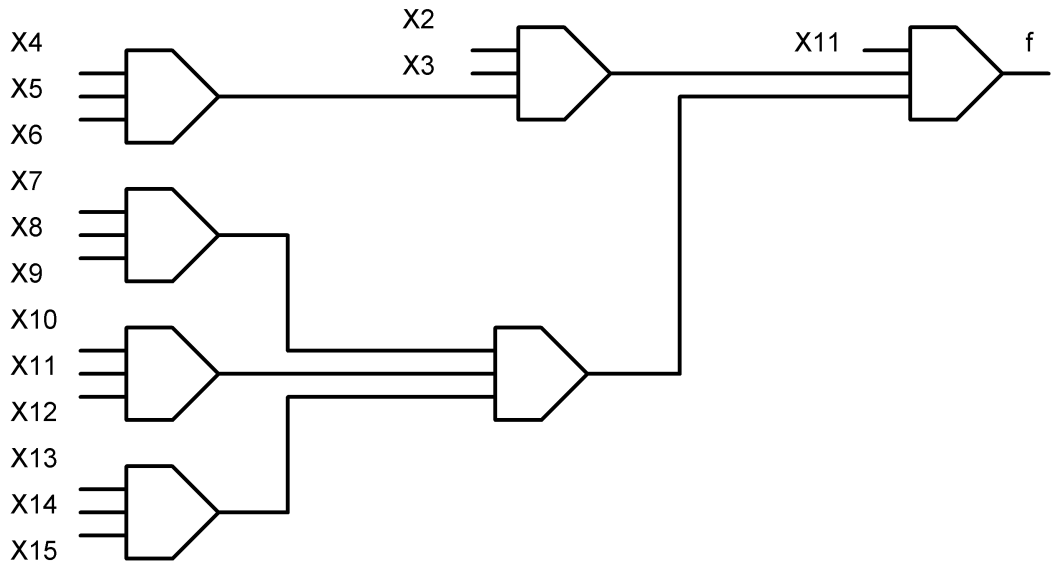


Fonte: Produção do próprio autor.

▪ $CCFM = 9$: São compostas por 7 portas lógicas majoritárias, com 1 no nível 1, 2 no nível 2 e 4 no nível 3. Essas funções possuem um formato do tipo $f = M(X_1, M(X_2, X_3, M(X_4, X_5, X_6)), M(M(X_7, X_8, X_9), M(X_{10}, X_{11}, X_{12}), M(X_{13}, X_{14}, X_{15})))$. São 15 entradas recebendo sinais de informação do barramento de dados. Em (31) apresenta-se um exemplo para esta classe de funções e na Figura 19 está sendo representado o circuito lógico combinacional correspondente.

$$f = [\,0\,1\,0\,1\,0\,1\,1\,1\,0\,1\,0\,1\,0\,1\,1\,1\,] =$$
$$= M(1, M(A, \bar{B}, M(0, A, C)), M(M(0, B, C), M(A, B, C), M(A, \bar{B}, 1))) \quad (31)$$

Figura 19 – Topologia de complexidade para $c = 9$.



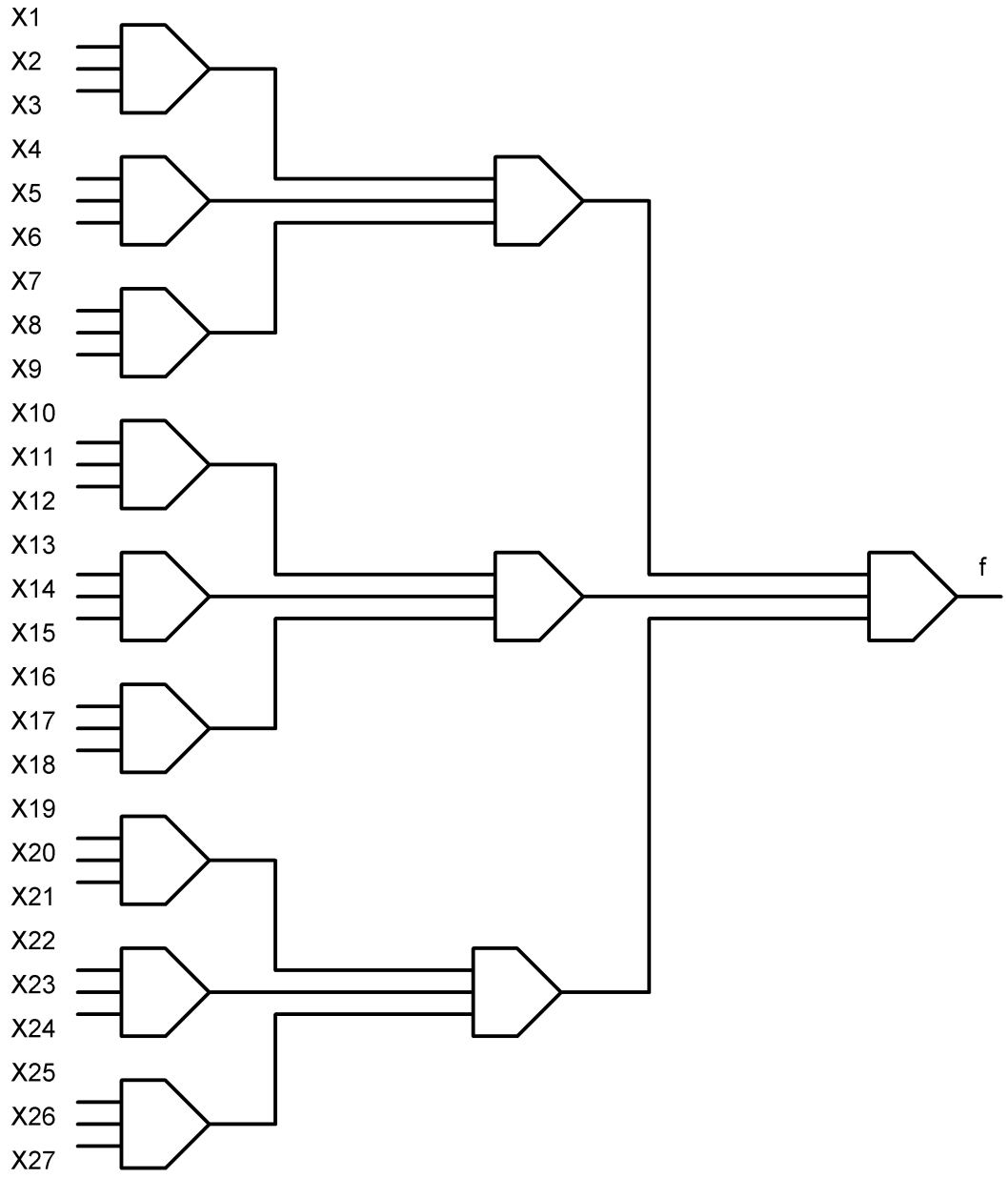
Fonte: Produção do próprio autor.

5.1.2 Funções com CCFM mais elevadas

Para avaliar as próximas classes de complexidade, basta seguir a metodologia de progressão e ir substituindo, gradativamente, as entradas livres por funções majoritárias. Deste modo, existem mais 6 classes com 3 níveis, em que $9 < c < 16$, antes de ser obtida a primeira classe de funções com 4 níveis, para a qual $c = 16$ e cuja estrutura é apresentada na Figura 20. Classes de maior complexidade podem ser obtidas, conforme o esquema descrito na Figura 21.

Nas funções genéricas que são utilizadas para representar as CCFM, X_i será chamada de *variável funcional*, uma vez que faz referência a uma FBF incógnita que é necessário descobrir no processo de minimização. Observe também que nas funções de 4 variáveis utilizadas como exemplos, aquelas em que $0 \leq c \leq 4$, representam suas formas mais minimizadas. Para as demais funções é possível obter formatos majoritários com minimização mais eficaz.

Figura 20 – Topologia de complexidade para $c = 15$.



Fonte: Produção do próprio autor.

5.1.3 Encontrando a classe adequada

Considerando o conjunto de funções majoritárias com m variáveis e analisando uma certa classe de complexidade, constata-se que é finita sua quantidade de funções. Isso ocorre porque que também é finita a quantidade de combinações possíveis envolvendo as $2m + 2$ FBF. Portanto, sempre que uma função não puder ser descrita em um formato de uma certa CCFM, deve haver um outro formato compatível, de modo que é plausível que se tente o formato da classe imediatamente posterior. Isso permite enunciar o Lema 2.

Figura 21 – Esquema para obtenção de uma CCFM.

<table>
<tr><th>NÍVEL 4</th><th>NÍVEL 3</th><th>NÍVEL 2</th><th>NÍVEL 1</th></tr>
<tr><td>PORTA 40</td><td rowspan="3">PORTA 13</td><td rowspan="9">PORTA 4</td><td rowspan="40">PORTA 1</td></tr>
<tr><td>PORTA 39</td></tr>
<tr><td>PORTA 38</td></tr>
<tr><td>PORTA 37</td><td rowspan="3">PORTA 12</td></tr>
<tr><td>PORTA 36</td></tr>
<tr><td>PORTA 35</td></tr>
<tr><td>PORTA 34</td><td rowspan="3">PORTA 11</td></tr>
<tr><td>PORTA 33</td></tr>
<tr><td>PORTA 32</td></tr>
<tr><td>PORTA 31</td><td rowspan="3">PORTA 10</td><td rowspan="12">PORTA 3</td></tr>
<tr><td>PORTA 30</td></tr>
<tr><td>PORTA 29</td></tr>
<tr><td>PORTA 28</td><td rowspan="3">PORTA 9</td></tr>
<tr><td>PORTA 27</td></tr>
<tr><td>PORTA 26</td></tr>
<tr><td>PORTA 25</td><td rowspan="3">PORTA 8</td></tr>
<tr><td>PORTA 24</td></tr>
<tr><td>PORTA 23</td></tr>
<tr><td>PORTA 22</td><td rowspan="3">PORTA 7</td><td rowspan="15">PORTA 2</td></tr>
<tr><td>PORTA 21</td></tr>
<tr><td>PORTA 20</td></tr>
<tr><td>PORTA 19</td><td rowspan="3">PORTA 6</td></tr>
<tr><td>PORTA 18</td></tr>
<tr><td>PORTA 17</td></tr>
<tr><td>PORTA 16</td><td rowspan="3">PORTA 5</td></tr>
<tr><td>PORTA 15</td></tr>
<tr><td>PORTA 14</td></tr>
</table>

Fonte: Produção do próprio autor.

▪ *Lema 2:* Se uma função majoritária f não pode ser expressa no formato de uma classe com $c = j$, então não pode ser expressa no formato de nenhuma outra classe com $c < j$.

Uma interpretação do Lema 2 permite concluir que, uma estratégia adequada de metodologia de solução do problema para a determinação da função majoritária minimizada, é iniciar tentando enquadrá-la na CCMJ mais baixa. Caso se inicie o processo numa classe superior e se encontre uma resposta, sempre haverá o risco de ter encontrado uma função majoritária contendo redundâncias. Deste modo, é uma boa prática metodológica avançar progressivamente, de modo a encontrar para a função f uma equivalência majoritária com o menor c possível. Ou seja, se já foi constatado que não há solução para $c = j$, avance para o universo de funções com $c = j + 1$.

5.2 OPERAÇÕES BOOLEANAS BIT A BIT

Considere o conjunto de todas as $2^{(2^m)}$ funções booleanas $f(B_1, B_2, \dots, B_m)$ possíveis de um sistema com m variáveis. Cada função f em sua forma vetorial é formada por $n = 2^m$ bits, ou seja, $f = [\, b_1, b_2, \dots, b_n \,]$. Assim como cada uma das $2m + 2$ FBF B_i também são formadas por n bits, ou seja, $B_i = [\, b_{i1}, b_{i2}, \dots, b_{in} \,]$. Cada um dos n bits b_i da forma vetorial de f pode ser obtido aplicando a mesma função das variáveis booleanas B_i aos bits b_{ij} correspondentes. Portanto, esta propriedade vale para todos os n bits de f, sendo descrita por (32).

$$b_i = f(b_{i1}, b_{i2}, \dots, b_{in}), \forall\, 1 \leq i \leq m \tag{32}$$

Para exemplificar esta propriedade, considere a função $f = AB + B\bar{C}$ de um sistema com 3 variáveis. Cada bit da forma vetorial de f pode ser obtido aplicando (32). A função vetorial obtida é apresentada em (33). Deste modo o resultado $f = [\, 0\ 0\ 1\ 1\ 0\ 0\ 0\ 1 \,]$, obtido pela aplicação bit a bit, representa exatamente a função inicial.

$$f = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \cdot 0 + 0 \cdot 1 \\ 1 \cdot 0 + 0 \cdot 1 \\ 0 \cdot 1 + 1 \cdot 1 \\ 1 \cdot 1 + 1 \cdot 1 \\ 0 \cdot 0 + 0 \cdot 0 \\ 1 \cdot 0 + 0 \cdot 0 \\ 0 \cdot 1 + 1 \cdot 0 \\ 1 \cdot 1 + 1 \cdot 0 \end{bmatrix} = \begin{bmatrix} 0 + 0 \\ 0 + 0 \\ 0 + 1 \\ 1 + 1 \\ 0 + 0 \\ 0 + 0 \\ 0 + 0 \\ 1 + 0 \end{bmatrix} \Longrightarrow f = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad (33)$$

Toda função dada em sua forma majoritária pode ser representada por uma função booleana algébrica equivalente com a aplicação do teorema fundamental da lógica majoritária, quantas vezes forem necessárias. Logo, a determinação de sua forma vetorial também pode ser feita bit a bit. Para exemplificar, em (34) é apresentada a obtenção da forma vetorial da função de 4 variáveis $f = M(0, B, \overline{D}) = 0 \cdot B + 0 \cdot \overline{D} + B \cdot \overline{D}$. Essa importante característica é enunciada no lema 3.

$$f = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \cdot 0 + 0 \cdot 1 + 0 \cdot 1 \\ 0 \cdot 0 + 0 \cdot 1 + 0 \cdot 1 \\ 0 \cdot 1 + 0 \cdot 1 + 1 \cdot 1 \\ 0 \cdot 1 + 0 \cdot 1 + 1 \cdot 1 \\ 0 \cdot 0 + 0 \cdot 1 + 0 \cdot 1 \\ 0 \cdot 0 + 0 \cdot 1 + 0 \cdot 1 \\ 0 \cdot 1 + 0 \cdot 1 + 1 \cdot 1 \\ 0 \cdot 1 + 0 \cdot 1 + 1 \cdot 1 \\ 0 \cdot 0 + 0 \cdot 0 + 0 \cdot 0 \\ 0 \cdot 0 + 0 \cdot 0 + 0 \cdot 0 \\ 0 \cdot 1 + 0 \cdot 0 + 1 \cdot 0 \\ 0 \cdot 1 + 0 \cdot 0 + 1 \cdot 0 \\ 0 \cdot 0 + 0 \cdot 0 + 0 \cdot 0 \\ 0 \cdot 0 + 0 \cdot 0 + 0 \cdot 0 \\ 0 \cdot 1 + 0 \cdot 0 + 1 \cdot 0 \\ 0 \cdot 1 + 0 \cdot 0 + 1 \cdot 0 \end{bmatrix} \Longrightarrow f = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (34)$$

▪ *Lema 3:* A forma vetorial de uma função na forma majoritária de 3 entradas pode ser obtida aplicando-se a própria função, bit a bit, em todos os bits relativos às respectivas posições.

5.3 SISTEMA DE EQUAÇÕES BOOLEANAS

Pretende-se encontrar a forma majoritária minimizada de uma função. Seguindo a diretriz do Lema 2, quando fornecida uma função f de m variáveis em seu formato vetorial, inicialmente deve ser feita uma busca dentre as $2m + 2$ funções de f^m, que contém as FBF. Se $f \in f^m$, o problema está solucionado, sendo um caso no qual tem-se $c = 0$. Qualquer tentativa de se obter um formato com $c > 0$, levaria a uma função contendo redundâncias e de mais alto custo. Para exemplificar, se a função for $f = [\,1\,1\,0\,0\,1\,1\,0\,0\,]$, a solução do problema é $f = f_7 = \bar{B} \in f^3$. Embora, por exemplo, tenha-se $M(0, 1, \bar{B}) = \bar{B}$, de modo que é dispensável o uso da função majoritária.

5.3.1 Equações booleanas nas variáveis vetoriais

Se $f \notin f^m$, avança-se para a próxima CCFM, com $c = 1$, de modo a verificar se f pode ser expressa pelo formato $f = M(X_1, X_2, X_3)$. Deste modo, é preciso determinar se existem $X_1, X_2, X_3 \in f^m$ que validam a igualdade. Um ponto de partida razoável é utilizar a relação fundamental, ou seja, determinar se existem $X_1, X_2, X_3 \in f^m$ que validam a igualdade $f = X_1X_2 + X_1X_3 + X_2X_3$. Assim, obtém-se um modelo de equação booleana formada por variáveis funcionais.

Utilizando o princípio descrito no Lema 3, propõe-se que, ao invés de identificar as variáveis funcionais, identifiquem-se os bits destas variáveis. Assim, define-se como *variável vetorial* qualquer bit de uma variável funcional, de modo que a cada variável funcional X_i corresponde n variáveis vetoriais $x_{i\,1}, x_{i\,2}, \ldots, x_{i\,n}$. Deste modo, no problema em questão há 3 variáveis funcionais e pode ser representado por um sistema de n equações booleanas e $3n$ variáveis vetoriais.

Para exemplificar esta proposta, considere o problema de encontrar a forma majoritária minimizada da função $f = [\,0\,1\,0\,1\,0\,1\,0\,1\,1\,1\,1\,1\,1\,1\,1\,]$. Como $f \notin f^4$, constrói-se o sistema representado por (35), composto por $n = 16$ equações booleanas e $3n = 48$ variáveis vetoriais. A forma majoritária minimizada para esta

função é $f = M(A, D, 1)$. A equivalência entre as formas vetorial e majoritária de f pode ser constatada em (36), onde também podem ser validadas todas as operações booleanas, bit a bit, decorrentes da relação fundamental.

$$\begin{cases} x_1x_{17} + x_1x_{33} + x_{17}x_{33} = 0 \\ x_2x_{18} + x_2x_{34} + x_{18}x_{34} = 1 \\ x_3x_{19} + x_3x_{35} + x_{19}x_{35} = 0 \\ x_4x_{20} + x_4x_{36} + x_{20}x_{36} = 1 \\ x_5x_{21} + x_5x_{37} + x_{21}x_{37} = 0 \\ x_6x_{22} + x_6x_{38} + x_{22}x_{38} = 1 \\ x_7x_{23} + x_7x_{39} + x_{23}x_{39} = 0 \\ x_8x_{24} + x_8x_{40} + x_{24}x_{40} = 1 \\ x_9x_{25} + x_9x_{41} + x_{25}x_{41} = 1 \\ x_{10}x_{26} + x_{10}x_{42} + x_{26}x_{42} = 1 \\ x_{11}x_{27} + x_{11}x_{43} + x_{27}x_{43} = 1 \\ x_{12}x_{28} + x_{12}x_{44} + x_{28}x_{44} = 1 \\ x_{13}x_{29} + x_{13}x_{45} + x_{29}x_{45} = 1 \\ x_{14}x_{30} + x_{14}x_{46} + x_{30}x_{46} = 1 \\ x_{15}x_{31} + x_{15}x_{47} + x_{31}x_{47} = 1 \\ x_{16}x_{32} + x_{16}x_{48} + x_{32}x_{48} = 1 \end{cases} \quad (35)$$

No exemplo, a função minimizada $f = M(A, D, 1)$ foi obtida encontrando-se a solução para todas as variáveis vetoriais x_{ij}, de modo que $x_{1\,1} = x_1, x_{1\,2} = x_2, \dots, x_{3\,16} = x_{48}$. Esta solução conduziu à solução das variáveis funcionais $X_1 = [\, x_1, x_2, \dots, x_{16}]$, $X_2 = [\, x_{17}, x_{18}, \dots, x_{32}]$ e $X_3 = [\, x_{33}, x_{34}, \dots, x_{48}]$, que permitiram a obtenção da solução final, conforme esquematizado em (37). Portanto, se a forma majoritária minimizada da função $f(X_1, X_2, \dots, X_m) = [\, b_1, b_2, \dots, b_n]$ tiver $c = 1$, será uma solução do modelo de sistema de equações booleanas apresentado em (38). Quanto maior o valor de m, maior será a quantidade de funções que podem ser minimizadas num formato com $c = 1$, em acordo com o disposto em (20).

$$f = \begin{bmatrix} 1\cdot 0 \\ 1\cdot 1 \\ 1\cdot 0 \\ 1\cdot 1 \\ 1\cdot 0 \\ 1\cdot 1 \\ 1\cdot 0 \\ 1\cdot 1 \\ 1\cdot 0 \\ 1\cdot 1 \\ 1\cdot 0 \\ 1\cdot 1 \\ 1\cdot 0 \\ 1\cdot 1 \\ 1\cdot 0 \\ 1\cdot 1 \end{bmatrix} + \begin{bmatrix} 1\cdot 0 \\ 1\cdot 0 \\ 1\cdot 0 \\ 1\cdot 0 \\ 1\cdot 0 \\ 1\cdot 0 \\ 1\cdot 0 \\ 1\cdot 0 \\ 1\cdot 1 \\ 1\cdot 1 \\ 1\cdot 1 \\ 1\cdot 1 \\ 1\cdot 1 \\ 1\cdot 1 \\ 1\cdot 1 \\ 1\cdot 1 \end{bmatrix} + \begin{bmatrix} 0\cdot 0 \\ 1\cdot 0 \\ 0\cdot 0 \\ 1\cdot 0 \\ 0\cdot 0 \\ 1\cdot 0 \\ 0\cdot 0 \\ 1\cdot 0 \\ 0\cdot 1 \\ 1\cdot 1 \\ 0\cdot 1 \\ 1\cdot 1 \\ 0\cdot 1 \\ 1\cdot 1 \\ 0\cdot 1 \\ 1\cdot 1 \end{bmatrix} = \begin{bmatrix} 0+0+0 \\ 1+0+0 \\ 0+0+0 \\ 1+0+0 \\ 0+0+0 \\ 1+0+0 \\ 0+0+0 \\ 1+0+0 \\ 0+1+0 \\ 1+1+1 \\ 0+1+0 \\ 1+1+1 \\ 0+1+0 \\ 1+1+1 \\ 0+1+0 \\ 1+1+1 \end{bmatrix} \Longrightarrow f = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \tag{36}$$

$$\begin{cases} x_1 = 0, x_2 = 1, \dots, x_{16} = 1 \Rightarrow X_1 = [\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,] \Rightarrow X_1 = A \\ x_{17} = 0, x_{18} = 0, \dots, x_{32} = 1 \Rightarrow X_2 = [\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,1\,1\,] \Rightarrow X_2 = D \\ x_{33} = 1, x_{34} = 1, \dots, x_{48} = 1 \Rightarrow X_3 = [\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,] \Rightarrow X_3 = 1 \end{cases} \tag{37}$$

$$\begin{cases} x_1 \cdot x_{n+1} + x_1 \cdot x_{2n+1} + x_{n+1} \cdot x_{2n+1} = b_1 \\ x_2 \cdot x_{n+2} + x_2 \cdot x_{2n+2} + x_{n+2} \cdot x_{2n+2} = b_2 \\ x_3 \cdot x_{n+3} + x_3 \cdot x_{2n+3} + x_{n+3} \cdot x_{2n+3} = b_3 \\ \vdots \\ x_n \cdot x_{2n} + x_{2n} \cdot x_{3n} + x_{2n} \cdot x_{3n} = b_n \end{cases} \tag{38}$$

5.3.2 Funções majoritárias com CCFM superior

Confirmado que uma função f não pode ser minimizada com $c = 1$, avança-se para a próxima classe de complexidade, inspecionando a possibilidade de possuir $c = 2$. Do mesmo modo como realizado para a classe anterior, expande-se a função majoritária correspondente utilizando apenas a relação fundamental, finalizando em

(39). Observe que as mesmas 5 variáveis funcionais presentes na forma majoritária compõem a expressão algébrica expandida, que é formada por uma SOP de 7 parcelas: 1 sendo o produto de 2 variáveis funcionais e 6 sendo o produto de 3 variáveis funcionais. Assim, o problema pode ser modelado como um sistema contendo n equações booleanas de $5n$ variáveis vetoriais com o aspecto representado por (40).

$$f = M(X_1, X_2, M(X_3, X_4, X_5)) = X_1X_2 + X_1M(X_3, X_4, X_5) + X_2M(X_3, X_4, X_5) = X_1X_2 + X_1 \cdot (X_3X_4 + X_3X_5 + X_4X_5) + X_2(X_3X_4 + X_3X_5 + X_4X_5) = X_1X_2 + X_1X_3X_4 + X_1X_3X_5 + X_1X_4X_5 + X_2X_3X_4 + X_2X_3X_5 + X_2X_4X_5 \quad (39)$$

$$x_ix_{i+n} + x_ix_{i+2n}x_{i+3n} + x_ix_{i+2n}x_{i+4n} + x_ix_{i+3n}x_{i+4n} + x_{i+n}x_{i+2n}x_{i+3n} + x_{i+n}x_{i+2n}x_{i+4n} + x_{i+n}x_{i+3n}x_{i+4n} = b_i \quad (40)$$

Para exemplificar esta aplicação, considere a função de $m = 4$ variáveis $f = [\,0\,1\,0\,1\,0\,0\,0\,1\,0\,1\,0\,1\,0\,0\,0\,1\,]$. O sistema de equações booleanas que representa o problema de obtenção da forma majoritária minimizada será composto por $n = 16$ equações booleanas e $m \cdot n = 4 \cdot 16 = 80$ variáveis vetoriais, 16 para cada uma das 5 variáveis funcionais. A solução adequada para o sistema está em (41). A confirmação da veracidade desta solução está em (42), onde todas as operações booleanas do sistema são realizadas, bit a bit, confirmando a função vetorial original. Portanto, $f = M(A, \bar{C}, M(0, B, C))$ é a solução do problema. Assim, qualquer outro formato majoritário com $c > 2$ contem redundâncias desnecessárias e não é sua forma minimizada.

5.3.3 Modelo de sistema para funções majoritárias com c = 3

Para modelar os próximos sistemas de equações booleanas correspondentes aos próximos valores de c, procede-se da mesma forma. Aplicando a relação fundamental, quantas vezes forem necessárias, obtém-se uma expansão algébrica para a forma majoritária contendo as variáveis funcionais. Sendo assim, para a classe de $c = 3$, (43) detalha a obtenção da expressão algébrica de f.

$$X_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} = A \quad X_2 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \bar{C} \quad X_3 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = 0 \quad X_4 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} = B \quad X_5 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = C \qquad (41)$$

$$f = \begin{bmatrix} 0\cdot1+0\cdot0\cdot0+0\cdot0\cdot0+0\cdot0\cdot0+1\cdot0\cdot0+1\cdot0\cdot0+1\cdot0\cdot0 \\ 1\cdot1+1\cdot0\cdot0+1\cdot0\cdot0+1\cdot0\cdot0+1\cdot0\cdot0+1\cdot0\cdot0+1\cdot0\cdot0 \\ 0\cdot1+0\cdot0\cdot1+0\cdot0\cdot0+0\cdot1\cdot0+1\cdot0\cdot1+1\cdot0\cdot0+1\cdot1\cdot0 \\ 1\cdot1+1\cdot0\cdot1+1\cdot0\cdot0+1\cdot1\cdot0+1\cdot0\cdot1+1\cdot0\cdot0+1\cdot1\cdot0 \\ 0\cdot0+0\cdot0\cdot0+0\cdot0\cdot1+0\cdot0\cdot1+0\cdot0\cdot0+0\cdot0\cdot1+0\cdot0\cdot1 \\ 1\cdot0+1\cdot0\cdot0+1\cdot0\cdot1+1\cdot0\cdot1+0\cdot0\cdot0+0\cdot0\cdot1+0\cdot0\cdot1 \\ 0\cdot0+0\cdot0\cdot1+0\cdot0\cdot1+0\cdot1\cdot1+0\cdot0\cdot1+0\cdot0\cdot1+0\cdot1\cdot1 \\ 1\cdot0+1\cdot0\cdot1+1\cdot0\cdot1+1\cdot1\cdot1+0\cdot0\cdot1+0\cdot0\cdot1+0\cdot1\cdot1 \\ 0\cdot1+0\cdot0\cdot0+0\cdot0\cdot0+0\cdot0\cdot0+1\cdot0\cdot0+1\cdot0\cdot0+1\cdot0\cdot0 \\ 1\cdot1+1\cdot0\cdot0+1\cdot0\cdot0+1\cdot0\cdot0+1\cdot0\cdot0+1\cdot0\cdot0+1\cdot0\cdot0 \\ 0\cdot1+0\cdot0\cdot1+0\cdot0\cdot0+0\cdot1\cdot0+1\cdot0\cdot1+1\cdot0\cdot0+1\cdot1\cdot0 \\ 1\cdot1+1\cdot0\cdot1+1\cdot0\cdot0+1\cdot1\cdot0+1\cdot0\cdot1+1\cdot0\cdot0+1\cdot1\cdot0 \\ 0\cdot0+0\cdot0\cdot0+0\cdot0\cdot1+0\cdot0\cdot1+0\cdot0\cdot0+0\cdot0\cdot1+0\cdot0\cdot1 \\ 1\cdot0+1\cdot0\cdot0+1\cdot0\cdot1+1\cdot0\cdot1+0\cdot0\cdot0+0\cdot0\cdot1+0\cdot0\cdot1 \\ 0\cdot0+0\cdot0\cdot1+0\cdot0\cdot1+0\cdot1\cdot1+0\cdot0\cdot1+0\cdot0\cdot1+0\cdot1\cdot1 \\ 1\cdot0+1\cdot0\cdot1+1\cdot0\cdot1+1\cdot1\cdot1+0\cdot0\cdot1+0\cdot0\cdot1+0\cdot1\cdot1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \qquad (42)$$

$$f = M(X_1, M(X_2, X_3, X_4), M(X_5, X_6, X_7)) = X_1 M(X_2, X_3, X_4) + X_1 M(X_5, X_6, X_7) + M(X_2, X_3, X_4) M(X_5, X_6, X_7) = X_1(X_2X_3 + X_2X_4 + X_3X_4) + X_1(X_5X_6 + X_5X_7 + X_6X_7) + (X_2X_3 + X_2 \cdot X_4 + X_3X_4)(X_5X_6 + X_5X_7 + X_6X_7) = X_1X_2X_3 + X_1X_2X_4 + X_1X_3X_4 + X_1X_5X_6 + X_1X_5X_7 + X_1X_6X_7 + X_2X_3X_5X_6 + X_2X_3X_5X_7 + X_2X_3X_6X_7 + X_2X_4X_5X_6 + X_2X_4X_5X_7 + X_2X_4X_6X_7 + X_3X_4X_5X_6 + X_3X_4X_5X_7 + X_3X_4X_6X_7 \quad (43)$$

A expansão algébrica obtida é composta por 7 variáveis funcionais e formada por uma SOP com 15 parcelas: 6 são produtos de 3 variáveis e 9 são produtos de 4 variáveis. Assim, o modelo do sistema para este problema é composto por n equações booleanas de $7n$ variáveis vetoriais.

5.3.4 Modelo de sistema para funções majoritárias com c = 4

Para funções majoritárias minimizadas com $c = 4$, a expansão algébrica é detalhada em (44), que foi obtida após algumas aplicações da relação fundamental. Como resultado, obtém-se uma SOP constituída por um total de 27 produtos com 4 variáveis. Esta representa a última estrutura de 2 níveis e contém 9 variáveis funcionais. O modelo do sistema para este problema é composto por n equações booleanas e $9n$ variáveis vetoriais.

$$f = M(M(X_1, X_2, X_3), M(X_4, X_5, X_6), M(X_7, X_8, X_9)) = M(X_1, X_2, X_3) \cdot M(X_4, X_5, X_6) + M(X_1, X_2, X_3) \cdot M(X_7, X_8, X_9) + M(X_4, X_5, X_6) \cdot M(X_7, X_8, X_9) = (X_1X_2 + X_1X_3 + X_2X_3)(X_4 \cdot X_5 + X_4X_6 + X_5X_6) + (X_1X_2 + X_1X_3 + X_2X_3)(X_7X_8 + X_7X_9 + X_8X_9) + (X_4X_5 + X_4X_6 + X_5 \cdot X_6)(X_7X_8 + X_7X_9 + X_8X_9) = X_1X_2X_4X_5 + X_1X_2X_4X_6 + X_1X_2X_5X_6 + X_1X_3X_4X_5 + X_1X_3X_4 X_6 + X_1X_3X_5X_6 + X_2X_3X_4X_5 + X_2X_3X_4X_6 + X_2X_3X_5X_6 + X_1X_2X_7X_8 + X_1X_2X_7X_9 + X_1X_2X_8 X_9 + X_1X_3X_7X_8 + X_1X_3X_7X_9 + X_1X_3X_8X_9 + X_2X_3X_7X_8 + X_2X_3X_7X_9 + X_2X_3X_8X_9 + X_4X_5X_7 X_8 + X_4X_5X_7X_9 + X_4X_5X_8X_9 + X_4X_6X_7X_8 + X_4X_6X_7X_9 + X_4X_6X_8X_9 + X_5X_6X_7X_8 + X_5X_6X_7 X_7X_9 + X_5X_6X_8X_9 \quad (44)$$

5.3.5 Modelo de sistema para funções majoritárias com c = 5

Para a primeira classe com 3 níveis, a expansão algébrica da forma majoritária com $c = 5$ está detalhada em (45), que também foi obtida após algumas aplicações

da relação fundamental. Como resultado, obtém-se uma SOP constituída por um total de 15 parcelas: 1 com o produto de 2 variáveis, 1 com o produto de 3 variáveis e 13 com o produto de 4 variáveis. Esta estrutura contém 7 variáveis funcionais e o modelo do sistema para este problema é composto por n equações booleanas e $7n$ variáveis vetoriais.

$$f = M(X_1, X_2, M(X_3, X_4, M(X_5, X_6, X_7))) = X_1X_2 + X_1M(X_3, X_4, M(X_5, X_6, X_7)) + X_2M(X_3, X_4, M(X_5, X_6, X_7)) = X_1X_2 + X_1(X_3X_4 + X_3M(X_5, X_6, X_7) + X_4M(X_5, X_6, X_7)) + X_2M(X_3X_4 + X_3M(X_5, X_6, X_7) + X_4M(X_5, X_6, X_7)) = X_1X_2 + X_1(X_3X_4 + X_3(X_5X_6 + X_5X_7 + X_6X_7) + X_4(X_5X_6 + X_5X_7 + X_6X_7)) + X_2(X_3X_4 + X_3(X_5X_6 + X_5X_7 + X_6X_7) + X_4(X_5X_6 + X_5X_7 + X_6X_7)) = X_1X_2 + X_1(X_3X_4 + X_3X_5X_6 + X_3X_5X_7 + X_3X_6X_7 + X_4X_5X_6 + X_4X_5X_7 + X_4X_6X_7) + X_2(X_3X_4 + X_3X_5X_6 + X_3X_5X_7 + X_3X_6X_7 + X_4X_5X_6 + X_4X_5X_7 + X_4X_6X_7) = X_1X_2 + X_1X_3X_4 + X_1X_3X_5X_6 + X_1X_3X_5X_7 + X_1X_3X_6X_7 + X_1X_4X_5X_6 + X_1X_4X_5X_7 + X_1X_4X_6X_7 + X_2X_3X_4 + X_2X_3X_5X_6 + X_2X_3X_5X_7 + X_2X_3X_6X_7 + X_2X_4X_5X_6 + X_2X_4X_5X_7 + X_2X_4X_6X_7 \quad (45)$$

5.3.6 Modelo de sistema para funções majoritárias com c superior

É notório que a complexidade do sistema de equações booleanas aumenta com o aumento da CCFM. A obtenção de expansões algébricas booleanas para valores de c mais elevados torna-se impraticável por execução manual, de modo que requer uso de aparato computacional. Isso é discutido no Capítulo 7. Como todas as variáveis funcionais X_i são incógnitas, alguns teoremas que poderiam simplificar a expressão não podem ser aplicados, de modo que as expressões algébricas destas expansões preservam todos os produtos e adições destas variáveis.

Todas as parcelas formadas pelos produtos das v variáveis funcionais X_k, que constituem as SOP das expansões, estão dispostos com seus índices k em ordem crescente e organizada, da esquerda para a direita. Em todos os modelos de sistemas de equações booleanas que representam estes problemas, estipulou-se que as variáveis vetoriais x_j sejam posicionadas de tal forma que x_1 represente o primeiro bit da forma vetorial da variável funcional X_1 e a última variável vetorial x_{vn} represente o último bit da forma vetorial da variável funcional X_v. Deste modo, no modelo de sistema de equações booleanas de um problema, a k-ésima variável funcional tem a forma vetorial incógnita $X_k = [\, x_{kn+1}, x_{kn+2}, \ldots, x_{kn+n} \,]$.

Conforme foi proposto, sempre haverá a possibilidade de se ter um sistema de equações booleanas atrelado ao problema da obtenção de uma função majoritária minimizada a partir de sua forma vetorial. O uso de um modelo de uma CCFM acima do necessário dará origem a um sistema superdimensionado. Isso permite encerrar este capítulo enunciando o próximo lema.

▪ *Lema 4*: O problema da determinação de uma função majoritária minimizada, a partir de sua forma vetorial, é representado por algum sistema de equações booleanas na forma SOP.

6 MODELAGEM DE UM PROBLEMA DE PROGRAMAÇÃO NÃO LINEAR

No Capítulo 5 foram descritos mecanismos que permitiram modelar um problema de minimização de uma função majoritária como um sistema de equações booleanas nas variáveis vetoriais. Como não há uma teoria regular para a solução destes sistemas, é natural que se questione a maneira como foram encontradas as soluções dos sistemas de equações booleanas propostos no capítulo anterior

Foram resolvidos utilizando o método MLP01, seguido de validação realizada nos próprios sistemas. Com um certo empenho, também poderiam ter sido resolvidos por tentativas, visto que foram exemplos simples. Para uma melhor compreensão, considere o problema de se obter a forma majoritária minimizada da função $f = [\,1\,0\,1\,0\,1\,0\,1\,1\,]$. Uma inspeção rápida constata que $f \notin f^3 \Rightarrow c > 0$. Para a próxima classe há 32 possibilidades e nenhuma delas é a solução, de modo que $c > 1$. Na classe seguinte há muito mais possibilidades e, a depender da habilidade teórica de quem realizar o procedimento, obtém-se a solução correta $f = M(1, \bar{A}, M(0, B, C))$.

Utilizar uma metodologia de tentativa e erro, mesmo de forma computacional, não constitui um método academicamente viável. Sendo assim, encontrar uma forma eficiente de resolver estes sistemas de equações booleanas foi um dos desafios deste trabalho e é discutido nas próximas seções.

6.1 ADIÇÃO E MULTIPLICAÇÃO BOOLEANAS COM MUITOS TERMOS

De acordo com o Capítulo 2, as Tabelas 1 e 2 apontam como proceder na adição e na multiplicação de quaisquer dois números booleanos. Aliadas às propriedades comutativa e associativa, fornecem o mecanismo para executar a adição e a multiplicação com quaisquer quantidades de parcelas e fatores. Como exemplo, em (46) é realizada a adição booleana $1 + 1 + 0 + 1 + 1 + 0 + 1 + 1$ executando-se associações e adições de números, dois a dois, da esquerda para a direita, até se obter o resultado 1. Estas associações podem ser executadas da maneira que se queira como, por exemplo, é indicado por (47), onde as associações agora são realizadas da esquerda para a direita, também obtendo como resultado da adição o número 1.

$$1+1+0+1+1+0+1+1=(1+1)+0+1+1+0+1+1=1+0+1+1+0+1+1=(1+0)+1+1+0+1+1=1+1+1+0+1+1=(1+1)+1+0+1+1=1+1+0+1+1=(1+1)+0+1+1=1+0+1+1=(1+0)+1+1=1+1+1=(1+1)+1=1+1=1 \quad (46)$$

$$1+1+0+1+1+0+1+1=1+1+0+1+1+0+(1+1)=1+1+0+1+1+0+1=1+1+0+1+1+(0+1)=1+1+0+1+1+1=1+1+0+1+(1+1)=1+1+0+1+1=1+1+0+(1+1)=1+1+0+1=1+1+(0+1)=1+1+1=1+(1+1)=1+1=1 \quad (47)$$

6.1.1 Adição booleana com muitas parcelas

Uma consequência direta destas propriedades é a de que qualquer adição algébrica booleana S contendo a parcelas de valor 1 e b parcelas de valor 0, devido à propriedade comutativa, poderá ser representada pela soma de todos os valores 1 com a soma de todos os valores 0, conforme é descrito em (48). Ou seja, uma adição booleana terá valor 1 sempre que ao menos uma das parcelas tenha valor 1 e terá valor 0 sempre que todas as parcelas forem 0.

Assim, a adição do exemplo anterior pode ser calculada diretamente da seguinte forma: $1+1+0+1+1+0+1+1=1$. A presença de uma única parcela de valor 1 é tão preponderante que a propriedade pode ser estendida a quantas parcelas se queira. Por exemplo, na adição de 1001 parcelas onde apenas 1 delas é o número booleano 1, o resultado é 1, conforme pode ser visto em (49). Portanto, a única possibilidade de uma adição algébrica booleana resultar em 0 é que todas as parcelas sejam 0.

$$S=1+1+\cdots+1+0+0+\cdots+0=\sum_{i=1}^{a}1+\sum_{i=1}^{b}0=1+0\Longrightarrow\Sigma=1 \quad (48)$$

$$S=1+\sum_{i=1}^{1000}0=1+0+0+0+\cdots+0+0+0=1 \quad (49)$$

6.1.2 Multiplicação booleana com muitos fatores

Em relação à multiplicação booleana, considere o problema de calcular o produto $1 \cdot 1 \cdot 0 \cdot 1 \cdot 0 \cdot 0 \cdot 1 \cdot 0$ executando associações e produtos de números, dois a dois, tomando sempre o segundo e terceiro fatores, até obter o resultado 0, conforme é apresentado em (50). Estas associações podem ser executadas de várias formas como, por exemplo, foi executado em (51), onde as associações agora são realizadas entre o penúltimo e o antepenúltimo fatores, também obtendo como resultado do produto o número 0.

$$1 \cdot 1 \cdot 0 \cdot 1 \cdot 0 \cdot 0 \cdot 1 \cdot 0 = 1 \cdot (1 \cdot 0) \cdot 1 \cdot 0 \cdot 0 \cdot 1 \cdot 0 = 1 \cdot 0 \cdot 1 \cdot 0 \cdot 0 \cdot 1 \cdot 0 = 1 \cdot (0 \cdot 1) \cdot 0 \cdot 0 \cdot 1 \cdot 0 = 1 \cdot 0 \cdot 0 \cdot 0 \cdot 1 \cdot 0 = 1 \cdot (0 \cdot 0) \cdot 0 \cdot 1 \cdot 0 = 1 \cdot 0 \cdot 0 \cdot 1 \cdot 0 = 1 \cdot (0 \cdot 0) \cdot 1 \cdot 0 = 1 \cdot 0 \cdot 1 \cdot 0 = 1 \cdot (0 \cdot 1) \cdot 0 = 1 \cdot 0 \cdot 0 = 1 \cdot (0 \cdot 0) = 1 = 1 \cdot 0 = 0 \quad (50)$$

$$1 \cdot 1 \cdot 0 \cdot 1 \cdot 0 \cdot 0 \cdot 1 \cdot 0 = 1 \cdot 1 \cdot 0 \cdot 1 \cdot 0 \cdot (0 \cdot 1) \cdot 0 = 1 \cdot 1 \cdot 0 \cdot 1 \cdot 0 \cdot 0 \cdot 0 = 1 \cdot 1 \cdot 0 \cdot 1 \cdot (0 \cdot 0) \cdot 0 = 1 \cdot 1 \cdot 0 \cdot 1 \cdot 0 \cdot 0 = 1 \cdot 1 \cdot 0 \cdot (1 \cdot 0) \cdot 0 = 1 \cdot 1 \cdot 0 \cdot 0 \cdot 0 = 1 \cdot 1 \cdot (0 \cdot 0) \cdot 0 = 1 \cdot 1 \cdot 0 \cdot 0 = 1 \cdot (1 \cdot 0) \cdot 0 = 1 \cdot 0 \cdot 0 = 1 \cdot (0 \cdot 0) = 1 \cdot 0 = 0 \quad (51)$$

Uma segunda consequência direta das propriedades da adição e multiplicação booleanas é a de que qualquer produto algébrico booleano P contendo a fatores de valor 1 e b fatores de valor 0, devido à comutatividade, poderá ser representada pelo produto de todos os valores 1 pelo produto de todos os valores 0, conforme é representado por (52). Ou seja, um produto booleano com muitos fatores terá valor 0 sempre que ao menos um dos fatores tenha valor 0, e terá valor 1 sempre que todos esses fatores sejam 1.

Assim, o produto do exemplo anterior pode ser calculado diretamente da seguinte forma: $1 \cdot 1 \cdot 0 \cdot 1 \cdot 0 \cdot 0 \cdot 1 \cdot 0 = 0$. A presença de um único fator de valor 0 é tão preponderante que a propriedade pode ser estendida a quantos fatores estiverem na operação. Por exemplo, em um produto de 1001 fatores, onde apenas 1 deles tem valor booleano 0, o resultado é 0, conforme descrito em (53). Portanto, dualmente à operação de adição, a única possibilidade de um produto algébrico booleano resultar em 1 é que todos os fatores sejam 1.

$$P = 1 \cdot 1 \cdot \cdots \cdot 1 \cdot 0 \cdot 0 \cdot \cdots \cdot 0 = \prod_{i=1}^{a} 1 \cdot \prod_{i=1}^{b} 0 = 1 \cdot 0 \Longrightarrow P = 0 \tag{52}$$

$$P = 0 \cdot \prod_{i=1}^{1000} 1 = 0 \cdot 1 \cdot 1 \cdot 1 \cdot \cdots \cdot 1 \cdot 1 \cdot 1 = 0 \tag{53}$$

6.2 ADAPTAÇÃO DO PROBLEMA AOS NÚMEROS NATURAIS

Retomando o problema da obtenção de uma função majoritária minimizada, considere um sistema de equações booleanas que o represente. Como na álgebra booleana que está sendo utilizada existem apenas os números 0 e 1, todas estas equações booleanas correspondem a uma igualdade com o número booleano 0 ou a uma igualdade com o número booleano 1. De fato, qualquer sequência de operações algébricas booleanas pode resultar apenas em 0 ou 1.

Como decorrência direta, qualquer equação booleana deste sistema representa uma soma de finitas parcelas, na qual cada uma é representada por um produto de, no mínimo, 2 fatores. Ou seja, toda equação deste sistema é formada pela soma de produtos das variáveis x_j. Matematicamente, se as equações do sistema contêm p parcelas e a i-ésima parcela contém q_i fatores x_j, então serão representadas, a depender da forma vetorial, por uma das duas igualdades descritas na Equação 54.

$$\sum_{i=1}^{p} \left[\left(\prod_{j=1}^{q_i} x_j \right)_i \right] = 0 \qquad \sum_{i=1}^{p} \left[\left(\prod_{j=1}^{q_i} x_j \right)_i \right] = 1 \tag{54}$$

6.2.1 Solução de equações booleanas

Com os resultados da seção anterior aliados à descrição da topologia das equações booleanas dos sistemas aqui tratados, é possível estabelecer mais dois lemas a respeito da solução destas equações. Os lemas 5 e 6 podem ser descritos matematicamente por (55) e (56), respectivamente.

▪ *Lema 5:* Para que uma equação algébrica booleana tenha como resultado o número booleano 0, todas as parcelas devem ter como valor o número booleano 0.

▪ *Lema 6:* Para que uma equação algébrica booleana tenha como resultado o número booleano 1, ao menos 1 parcela deve ter como valor o número booleano 1.

$$\sum_{i=1}^{p}\left[\left(\prod_{j=1}^{q_i} x_j\right)_i\right] = 0 \Leftrightarrow \left(\prod_{j=1}^{q_i} x_j\right)_i = 0, \forall\, i \tag{55}$$

$$\sum_{i=1}^{p}\left[\left(\prod_{j=1}^{q_i} x_j\right)_i\right] = 1 \Leftrightarrow \exists \left(\prod_{j=1}^{q_i} x_j\right)_i = 1 \tag{56}$$

Para compreender a aplicação dos Lemas 5 e 6 numa equação booleana de interesse, considere a primeira equação do sistema que representa o problema para $c = 1$, numa estrutura de m variáveis, composta por uma soma de 3 produtos de 2 fatores, relativas a 3 variáveis vetoriais. (57) e (58) representam a aplicação dos dois lemas à equação booleana. As equivalências são obtidas utilizando os Lemas 5 e 6, respectivamente. Avaliando as possíveis soluções para esta equação, elaborou-se a Tabela 17.

$$x_1x_{n+1} + x_1x_{2n+1} + x_{n+1}x_{2n+1} = 0 \Leftrightarrow x_1x_{n+1} = 0 \ \wedge\ x_1x_{2n+1} = 0 \ \wedge\ x_{n+1}x_{2n+1} = 0 \tag{57}$$

$$x_1x_{n+1} + x_1x_{2n+1} + x_{n+1}x_{2n+1} = 1 \Leftrightarrow x_1x_{n+1} = 1 \ \vee\ x_1x_{2n+1} = 1 \ \vee\ x_{n+1}x_{2n+1} = 1 \tag{58}$$

Analisando os dados dessa tabela, constata-se que há um total de 4 soluções para a equação nas variáveis vetoriais x_1, x_{n+1} e x_{2n+1} na igualdade com o número booleano 0 e um total de 4 soluções na igualdade com o número booleano 1. Considerando que são soluções para duas equações simples, tem-se um parâmetro sobre quão complexo pode se tornar um problema desta natureza. Deste modo, não havendo técnicas específicas consagradas para a resolução de equações booleanas

desta natureza e, consequentemente, para os sistemas destas equações, é preciso uma nova estratégia de solução.

Tabela 17 – Possibilidades para uma equação booleana de 3 variáveis vetoriais.

x_1	x_{n+1}	x_{2n+1}	x_1x_{n+1}	x_1x_{2n+1}	$x_{n+1}x_{2n+1}$	$x_1x_{n+1} + x_1x_{2n+1} + x_{n+1}x_{2n+1}$
0	0	0	0	0	0	0 + 0 + 0 = 0
0	0	1	0	0	0	0 + 0 + 0 = 0
0	1	0	0	0	0	0 + 0 + 0 = 0
0	1	1	0	0	1	0 + 0 + 1 = 1
1	0	0	0	0	0	0 + 0 + 0 = 0
1	0	1	0	1	0	0 + 1 + 0 = 1
1	1	0	1	0	0	1 + 0 + 0 = 1
1	1	1	1	1	1	1 + 1 + 1 = 1

Fonte: Produção do próprio autor.

6.2.2 Adaptação aos números naturais 0 e 1

A discussão até este estágio de apresentação ocorreu dentro do universo da álgebra booleana do conjunto $\mathbb{B}$, que abrange as constantes booleanas 0 e 1. Considere agora fazer uma migração de $\mathbb{B}$ para o conjunto $\mathbb{R}$ dos números reais. O conjunto dos números reais contempla uma série de axiomas e propriedades (Zorich, 2004). Sendo mais específico, considere o conjunto $\mathbb{N}$ dos números naturais, $\mathbb{N} \subset \mathbb{R}$, com atenção particular aos números naturais 0 e 1. Agora, considere avaliar o formato de uma equação booleana aplicada aos números naturais 0 e 1. Assim, é preciso estabelecer o que ocorre com a soma de produtos dentro do universo do conjunto $\mathbb{N}$.

Em decorrência das propriedades comutativa, associativa, elemento neutro da adição e elemento nulo da multiplicação de números naturais, um produto composto apenas por fatores 0 ou 1, é 0 quando ao menos 1 fator for 0 e é 1 quando todos os fatores forem 0 (Lafferriere; Nguyen, 2022). Uma consequência imediata deste fato é a de que um produto destes números pode ter como resultado apenas 0 ou 1. Sendo assim, um produto de q fatores x_j, com valor 1 ou 0, pode ser representado matematicamente por (59).

$$x_j \in \{0,1\} \subset \mathbb{N} \Longrightarrow \prod_{j=1}^{q} x_j \in \{0,1\} \subset \mathbb{N}$$

(59)

Como o produto de números naturais x_j só pode resultar em 0 ou 1, então uma equação, representada por uma soma de produtos destas variáveis só pode resultar em uma adição entre números naturais 0 e números naturais 1, equivalendo a um número natural múltiplo de 1. Deste modo, tal característica pode ser representada matematicamente por (60). Observe que o número natural b pode ser igual a 0, pois 0 também é um múltiplo de 1. Este caso particular ocorre quando todos os produtos tem valor 0, não ocorrendo parcela de valor 1.

$$\sum_{i=1}^{p}\left[\left(\prod_{j=1}^{q_i} x_j\right)_i\right] = \sum_{k=1}^{a} 0 + \sum_{k=1}^{b} 1 = a \cdot 0 + b \cdot 1 = 0 + b = b \in \mathbb{N}$$

(60)

Da mesma forma que os valores das variáveis vetoriais x_j puderam migrar de $\mathbb{B}$ para $\mathbb{N}$, os bits da forma vetorial da função f também podem. Voltando ao problema de se obter a função majoritária minimizada, seja b_k o k-ésimo bit da forma vetorial de f, de modo que $b_k \in \{0,1\} \subset \mathbb{N}$. Utilizando conjuntamente os resultados de (59) e (60), é possível agora reescrever qualquer equação booleana do sistema que modela o problema em uma forma equivalente do universo $\mathbb{N}$. Tal equivalência pode se enquadrar em uma das duas possibilidades descritas por (61). Note que para o caso de $b_k = 1$, o valor da soma dos produtos será um múltiplo de 1.

$$b_k = 0 \Leftrightarrow \sum_{i=1}^{p}\left[\left(\prod_{j=1}^{q_i} x_j\right)_i\right] = 0 \qquad b_k = 1 \Leftrightarrow \sum_{i=1}^{p}\left[\left(\prod_{j=1}^{q_i} x_j\right)_i\right] \geq 1$$

(61)

Para exemplificar a aplicação desta nova abordagem, considere o problema de obter a forma majoritária minimizada da função $f = [\,0\;1\;0\;1\;1\;1\;0\;1\,]$. A solução é $M(A, 1, M(A, C, \bar{B}))$, sendo f uma função com $c = 2$. Portanto o problema tem como

solução $X_1 = A,\ X_2 = 1,\ X_3 = A,\ X_4 = C$ e $X_5 = \bar{B}$. Para exemplificar e constatar a validade da abordagem do problema no universo de $\mathbb{N}$, em (62) estão representadas todas as operações entre os valores das variáveis vetoriais x_j, agora dentro do universo dos números naturais. Nos cálculos foi utilizada a forma majoritária característica de $c = 2$. Observe que para os três bits 0 de f, o resultado da soma de produtos das variáveis x_j foi 0. Para todos os outros 5 bits, a soma de produtos resultou em valores maiores ou iguais a 1, todos múltiplos de 1. Destaca-se o valor 7, relativo ao bit b_6, como o maior valor que se poderia obter, pois a adição de 7 parcelas iguais a 1 vale 7.

$$\begin{cases} 0\cdot 1 + 0\cdot 0\cdot 0 + 0\cdot 1\cdot 1 + 0\cdot 0\cdot 1 + 1\cdot 0\cdot 0 + 1\cdot 0\cdot 1 + 1\cdot 0\cdot 1 = 0 = 0 \\ 1\cdot 1 + 1\cdot 1\cdot 0 + 1\cdot 1\cdot 1 + 1\cdot 0\cdot 1 + 1\cdot 1\cdot 0 + 1\cdot 1\cdot 1 + 1\cdot 0\cdot 1 = 3 \geq 1 \\ 0\cdot 1 + 0\cdot 0\cdot 0 + 0\cdot 1\cdot 0 + 0\cdot 0\cdot 0 + 1\cdot 0\cdot 0 + 1\cdot 0\cdot 0 + 1\cdot 0\cdot 0 = 0 = 0 \\ 1\cdot 1 + 1\cdot 1\cdot 0 + 1\cdot 1\cdot 0 + 1\cdot 0\cdot 0 + 1\cdot 1\cdot 0 + 1\cdot 1\cdot 0 + 1\cdot 0\cdot 0 = 1 \geq 1 \\ 0\cdot 1 + 0\cdot 0\cdot 1 + 0\cdot 1\cdot 1 + 0\cdot 1\cdot 1 + 1\cdot 0\cdot 1 + 1\cdot 0\cdot 1 + 1\cdot 1\cdot 1 = 1 \geq 1 \\ 1\cdot 1 + 1\cdot 1\cdot 1 + 1\cdot 1\cdot 1 + 1\cdot 1\cdot 1 + 1\cdot 1\cdot 1 + 1\cdot 1\cdot 1 + 1\cdot 1\cdot 1 = 7 \geq 1 \\ 0\cdot 1 + 0\cdot 0\cdot 1 + 0\cdot 1\cdot 0 + 0\cdot 1\cdot 0 + 1\cdot 0\cdot 1 + 1\cdot 0\cdot 0 + 1\cdot 1\cdot 0 = 0 = 0 \\ 1\cdot 1 + 1\cdot 1\cdot 1 + 1\cdot 1\cdot 0 + 1\cdot 1\cdot 0 + 1\cdot 1\cdot 1 + 1\cdot 1\cdot 0 + 1\cdot 1\cdot 0 = 3 \geq 1 \end{cases} \tag{62}$$

6.3 PROBLEMA DE PROGRAMAÇÃO NÃO LINEAR

Como demonstrado na seção anterior, o problema de obtenção da função majoritária minimizada de um sistema com m variáveis, adaptado ao conjunto dos números naturais, pode assumir o formato apresentado em (63). Há $n = 2^m$ restrições do tipo soma de p produtos com q_i fatores das variáveis x_j e mais $n \cdot v$ restrições de pertinência ao conjunto $\mathbb{N}$, sendo v a quantidade de variáveis funcionais relativas a CCFM.

$$\begin{cases} b_k = 0 \Leftrightarrow \sum_{i=1}^{p} \left[\left(\prod_{j=1}^{q_i} x_{kj} \right)_i \right] = 0 \qquad b_k = 1 \Leftrightarrow \sum_{i=1}^{p} \left[\left(\prod_{j=1}^{q_i} x_{kj} \right)_i \right] \geq 1 \\ x_j \in \{0, 1\} \subset \mathbb{N} \end{cases} \tag{63}$$

Em um problema de programação linear inteira binária, as restrições devem ser tais que, além das variáveis só poderem assumir os valores inteiros 0 e 1, devem ser formadas pela soma de parcelas com a característica de serem constituídas pelo produto de um número real por uma única variável (Bazaraa; Jarvis; Sherali, 2010). Portanto, o problema aqui descrito não é de programação linear inteira binária pois, embora as variáveis sejam binárias, cada parcela das restrições é formada pelo produto de, ao menos, duas variáveis.

A modelagem descrita por (63) representa um Problema de Programação Não Linear Inteira Binária (PPNLIB) (Bazaraa; Sherali; Shetty, 2006). É adequado que se trabalhe com problemas de números inteiros para que se possam contemplar também os números negativos. Embora as análises tenham sido feitas no conjunto $\mathbb{N} \subset \mathbb{Z}$, todas as constatações continuam válidas para os números 0 e 1 pertencentes ao conjunto dos números inteiros $\mathbb{Z}$. Como conclusão, esta subseção é finalizada enunciando o Lema 7.

▪ *Lema 7*: O problema da determinação de uma função majoritária minimizada, a partir de sua forma vetorial, é representado por algum problema de programação não linear inteira binária.

6.3.1 Restringindo as variáveis do problema

Para consolidar o conjunto de restrições do PPNLIB é necessário que seja avaliado um outro aspecto de sua estrutura. Em todos os exemplos propostos foram apresentadas soluções que realmente condizem com os problemas, todas representando funções majoritárias minimizadas, correspondentes às respectivas formas vetoriais. Considere agora o problema de obter a função majoritária minimizada de $f = [\,0\,0\,0\,1\,0\,1\,1\,1\,]$. A solução adequada é a função $M(A,B,C)$, relativa ao formato $M(X_1,X_2,X_3) = X_1X_2 + X_1X_3 + X_2X_3$ de $c = 1$, que é apresentada em (64). Em (65) está a verificação da solução.

Ainda com relação ao problema anterior, observe (66), que apresenta um outro conjunto de valores para X_1, X_2 e X_3. Em (67) é verificado que essa nova terna de valores também é solução do PPNLIB proposto. Porém, o valor obtido para X_3 não é uma FBF, ou seja, $X_3 = [\,0\,0\,0\,1\,1\,1\,1\,1\,] \notin f^3$. A função obtida para X_3 representa a função $M(1,C,M(0,A,B))$, ou seja, obteve-se como solução a função $f =$

$M(A,B,M(1,C,M(0,A,B)))$, que corresponde a $c = 5$. Portanto, embora essa terna não represente a função majoritária simplificada, também representa uma solução do sistema de equações não lineares.

$$X_1 = A = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \qquad X_2 = B = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} \qquad X_3 = C = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \tag{64}$$

$$\begin{bmatrix} 0\cdot 0 + 0\cdot 0 + 0\cdot 0 \\ 1\cdot 0 + 1\cdot 0 + 0\cdot 0 \\ 0\cdot 1 + 0\cdot 0 + 1\cdot 0 \\ 1\cdot 1 + 1\cdot 0 + 1\cdot 0 \\ 0\cdot 0 + 0\cdot 1 + 0\cdot 1 \\ 1\cdot 0 + 1\cdot 1 + 0\cdot 1 \\ 0\cdot 1 + 0\cdot 1 + 1\cdot 1 \\ 1\cdot 1 + 1\cdot 1 + 1\cdot 1 \end{bmatrix} = \begin{bmatrix} 0+0+0 \\ 0+0+0 \\ 0+0+0 \\ 1+0+0 \\ 0+0+0 \\ 0+1+0 \\ 0+0+1 \\ 1+1+1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} \tag{65}$$

Da mesma forma, em (68) estão dispostas as formas vetoriais de uma terceira solução para o PPNLIB. A verificação desta solução pode ser comprovada por (69). Nesta terceira solução, a terceira variável funcional corresponde a $X_3 = [\,1\;0\;0\;0\;0\;1\;1\;1\,] = M(M(0,\bar{B},\bar{C}),M(1,C,\bar{A}),M(A,B,C))$, que também não representa uma FBF.

Em (70) são apresentadas as 3 soluções mencionadas, das quais a primeira é a correta, pois representa a forma minimizada. Note que para a implementação, seria necessária 1 única porta lógica majoritária de 3 entradas para a primeira função, 3 portas lógicas para a segunda função e 5 portas lógicas para a terceira função. Deste modo, até aqui, um solucionador não saberia distinguir estas soluções encontradas.

$$X_1 = A = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \qquad X_2 = B = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \qquad X_3 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \tag{66}$$

$$\begin{bmatrix} 0\cdot 0 + 0\cdot 0 + 0\cdot 0 \\ 1\cdot 0 + 1\cdot 0 + 0\cdot 0 \\ 0\cdot 1 + 0\cdot 0 + 1\cdot 0 \\ 1\cdot 1 + 1\cdot 1 + 1\cdot 1 \\ 0\cdot 0 + 0\cdot 1 + 0\cdot 1 \\ 1\cdot 0 + 1\cdot 1 + 0\cdot 1 \\ 0\cdot 1 + 0\cdot 1 + 1\cdot 1 \\ 1\cdot 1 + 1\cdot 1 + 1\cdot 1 \end{bmatrix} = \begin{bmatrix} 0+0+0 \\ 0+0+0 \\ 0+0+0 \\ 1+1+1 \\ 0+0+0 \\ 0+1+0 \\ 0+0+1 \\ 1+1+1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} \tag{67}$$

$$X_1 = A = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \qquad X_2 = B = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} \qquad X_3 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} \tag{68}$$

Portanto, é necessário estabelecer restrições que façam com que qualquer variável funcional $X_i = [\, x_{i1}\, x_{i2}\, \cdots\, x_{in}\,]$ seja uma FBF. Para que se possa descrever e compreender corretamente estas restrições, observe com atenção todas as 8 FBF para $m = 3$ variáveis em suas formas vetoriais e também todas as 10 FBF para $m = 4$ variáveis em suas formas vetoriais, que estão representadas nas Tabelas 18 e 19, respectivamente. A proposta é destacar características que tornem as $2m + 2$ FBF únicas dentre todas as $2^{(2^m)}$ funções possíveis.

$$\begin{bmatrix} 0\cdot 0+0\cdot 1+0\cdot 1 \\ 1\cdot 0+1\cdot 0+0\cdot 0 \\ 0\cdot 1+0\cdot 0+1\cdot 0 \\ 1\cdot 1+1\cdot 0+1\cdot 0 \\ 0\cdot 0+0\cdot 0+0\cdot 0 \\ 1\cdot 0+1\cdot 1+0\cdot 1 \\ 0\cdot 1+0\cdot 1+1\cdot 1 \\ 1\cdot 1+1\cdot 1+1\cdot 1 \end{bmatrix} = \begin{bmatrix} 0+0+0 \\ 0+0+0 \\ 0+0+0 \\ 1+0+0 \\ 0+0+0 \\ 0+1+0 \\ 0+0+1 \\ 1+1+1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} \tag{69}$$

$$\begin{cases} f = M(A,B,C) \Longrightarrow c = 1 \\ f = M(A,B,M(1,C,M(0,A,B))) \Longrightarrow c = 5 \\ f = M(A,B,M(M(0,\bar{B},\bar{C}),M(1,C,\bar{A}),M(A,B,C))) \Longrightarrow c = 7 \end{cases} \tag{70}$$

Tabela 18 – Funções booleanas fundamentais para $m = 3$.

0	A	B	C	1	$\bar{A}$	B	$\bar{C}$
0	0	0	0	1	1	1	1
0	1	0	0	1	0	1	1
0	0	1	0	1	1	0	1
0	1	1	0	1	0	0	1
0	0	0	1	1	1	1	0
0	1	0	1	1	0	1	0
0	0	1	1	1	1	0	0
0	1	1	1	1	0	0	0

Fonte: Produção do próprio autor.

6.3.2 Características das funções booleanas fundamentais

Nas Tabelas 18 e 19 as FBF estão dispostas, convenientemente, na forma vertical para que haja uma melhor visualização das propriedades que as diferenciam das demais funções. Estas propriedades serão chamadas de *características das FBF* e, quando avaliadas algebricamente, os números booleanos 0 e 1 estarão associados aos números inteiros 0 e 1, respectivamente. As características serão indexadas progressivamente com a notação C_i.

Tabela 19 – Funções booleanas fundamentais para $m = 4$.

0	A	B	C	D	1	$\bar{A}$	$\bar{B}$	$\bar{C}$	$\bar{D}$
0	0	0	0	0	1	1	1	1	1
0	1	0	0	0	1	0	1	1	1
0	0	1	0	0	1	1	0	1	1
0	1	1	0	0	1	0	0	1	1
0	0	0	1	0	1	1	1	0	1
0	1	0	1	0	1	0	1	0	1
0	0	1	1	0	1	1	0	0	1
0	1	1	1	0	1	0	0	0	1
0	0	0	0	1	1	1	1	1	0
0	1	0	0	1	1	0	1	1	0
0	0	1	0	1	1	1	0	1	0
0	1	1	0	1	1	0	0	1	0
0	0	0	1	1	1	1	1	0	0
0	1	0	1	1	1	0	1	0	0
0	0	1	1	1	1	1	0	0	0
0	1	1	1	1	1	0	0	0	0

Fonte: Produção do próprio autor.

▪ *Característica C₁*: Na ordem em que estão estabelecidas, as $m + 1$ últimas FBF são, ordenadamente, as inversas das $m + 1$ respectivas primeiras FBF. Dualmente, as $m + 1$ primeiras funções FBF são, ordenadamente, as inversas das $m + 1$ respectivas últimas FBF. Na apresentação das próximas características será mostrado que o que se aplica ao primeiro grupo de FBF também se aplica ao segundo grupo.

▪ *Característica C₂*: As quantidades de 0's, aqui representada por $Q_0(f_i)$ e, dualmente, a quantidade de 1's, aqui representada por $Q_1(f_i)$, presentes na forma vetorial da i-ésima FBF f_i, relativas a um sistema de m variáveis, têm valores dados por (71). Isso equivale a afirmar que para as $2m + 2$ FBF, sempre a primeira função $f_1 = 0$ é formada apenas por 0's e a $(m + 2)$-ésima função $f_{m+2} = 1$ é formada apenas por 1's. As outras $2m$ funções sempre têm quantidades iguais de 0's e 1's em sua forma vetorial.

Deve-se notar que as características C_1 e C_2 ainda não são suficientes para caracterizar com exclusividade as funções booleanas fundamentais. Na Tabela 20,

por exemplo, estão ordenadas 8 funções, relativas a $m = 3$, contendo essas duas características e que não representam f^3. Como pode ser observado, neste caso apenas a primeira e a última funções são FBF. Portanto é preciso refinar mais as características.

$$Q_0(f_i) = \begin{cases} n, se\ i = 1 \\ \frac{n}{2}, se\ m+2 < i < 2m+2 \\ 0, se\ i = m+2 \end{cases} \qquad Q_0(f_i) = \begin{cases} 0, se\ i = 1 \\ \frac{n}{2}, se\ m+2 < i < 2m+2 \\ n, se\ i = m+2 \end{cases} \tag{71}$$

Tabela 20 – Funções não fundamentais com as características C_1 e C_2.

F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8
0	0	0	0	1	1	1	1
0	1	0	1	0	1	0	1
0	1	0	1	0	1	0	1
0	1	1	0	0	0	1	1
0	0	1	1	1	0	0	1
0	0	1	1	1	0	0	1
0	0	1	0	1	0	1	1
0	1	0	0	0	1	1	1

Fonte: Produção do próprio autor.

▪ *Característica C_3*: Em qualquer FBF f_i, os n números 0 ou 1 que podem compor sua forma vetorial estão alternada e uniformemente distribuídos. Isso significa que, a partir de $f_1 = 0$, onde há uma sequência uniforme de números 0, todas as demais FBF seguem um padrão uniforme na distribuição dos números 0 e dos números 1 que as constituem.

Agora, com a característica C_3 há meios para serem obtidas todas as FBF para qualquer quantidade m de variáveis. Em (72) é apresentado um modelo de construção ordenada para todas estas $2m + 2$ funções. A seguir, são discutidas outras propriedades de caráter numérico que permitem transcrever matematicamente todas essas características.

$$0=\begin{bmatrix}0\\0\\0\\\vdots\\0\\0\\0\end{bmatrix}\ A=\begin{bmatrix}0\\1\\0\\\vdots\\1\\0\\1\end{bmatrix}\ B=\begin{bmatrix}0\\0\\1\\\vdots\\0\\1\\1\end{bmatrix}\ \cdots\ f_{m+1}=\begin{bmatrix}0\\0\\0\\\vdots\\1\\1\\1\end{bmatrix}\ 1=\begin{bmatrix}1\\1\\1\\\vdots\\1\\1\\1\end{bmatrix}\ \bar{A}=\begin{bmatrix}1\\0\\1\\\vdots\\0\\1\\0\end{bmatrix}\ \bar{B}=\begin{bmatrix}1\\1\\0\\\vdots\\1\\0\\0\end{bmatrix}\ \cdots\ f_{2m+2}=\begin{bmatrix}1\\1\\1\\\vdots\\0\\0\\0\end{bmatrix} \tag{72}$$

▪ *Característica C_4*: Na forma vetorial de toda FBF, a cada sequência de 4 números, a soma do primeiro número com o quarto é igual à soma do segundo número com o terceiro. Sendo $2^m = 2^2 \cdot 2^{m-2} = 4 \cdot 2^{m-2}$ a quantidade de números de cada função f_i, há 2^{m-2} sequências de 4 números para cada uma delas e, consequentemente, há $(2m+2) \cdot 2^{m-2} = (m+1) \cdot 2^{m-1}$ sequências relativas a todas as FBF. Logo, há também $(m+1) \cdot 2^{m-1}$ restrições deste tipo para o problema. Em (73) está formalizada a k-ésima restrição do problema relativa a esta característica e, para exemplificar, em (74) está representado o conjunto de todas as restrições relativas ao caso de $m = 3$.

$$x_{4k-3} + x_{4k} = x_{4k-2} + x_{4k-1} \Leftrightarrow x_{4k-3} - x_{4k-2} - x_{4k-1} + x_{4k} = 0 \tag{73}$$

$$\begin{cases} x_1 - x_2 - x_3 + x_4 = 0 \\ x_5 - x_6 - x_7 + x_8 = 0 \\ \vdots \\ x_{57} - x_{58} - x_{59} + x_{60} = 0 \\ x_{61} - x_{62} - x_{63} + x_{64} = 0 \end{cases} \tag{74}$$

Este conjunto de novas restrições não são suficientes para selecionar apenas as FBF. Considere, por exemplo, as 10 funções apresentadas na Tabela 21. Observe que a característica C4 pode ser observada em todas estas funções, mesmo que nenhuma seja uma FBF. Portanto, ainda é necessário estabelecer outras características restritivas para que sejam selecionadas apenas as funções booleanas fundamentais.

▪ *Característica C_5:* Na forma vetorial de toda FBF, a cada sequência de 8 números, a soma do primeiro número com o oitavo número é igual à soma do segundo número com o sétimo, igual à soma do terceiro número com o sexto e igual à soma do quarto número com o quinto. Sendo $2^m = 2^3 \cdot 2^{m-3} = 8 \cdot 2^{m-3}$ a quantidade de números de cada função f_i, há 2^{m-3} sequências para cada uma delas e, consequentemente, há $(2m+2) \cdot 2^{m-3} = (m+1) \cdot 2^{m-2}$ sequências relativas a todas as FBF. Logo, há também $3(m+1) \cdot 2^{m-2}$ restrições deste tipo para o problema. Em (75) estão formalizadas as 3 k-ésimas restrições do problema relativas a esta característica e em (76) está representado o conjunto de restrições relativas a $m = 3$.

Tabela 21 – Funções não fundamentais com a características C_4.

F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8
0	0	0	1	1	0	1	1
0	1	1	1	1	0	1	1
0	0	0	1	0	1	1	1
0	1	1	1	0	1	1	1
1	1	0	0	0	0	0	1
0	1	0	1	1	0	0	0
1	1	0	0	0	0	1	1
0	1	0	1	1	0	1	0

Fonte: Produção do próprio autor.

$$\begin{cases} x_{8k-7} + x_{8k} = x_{8k-6} + x_{8k-1} \Leftrightarrow x_{8k-7} - x_{8k-6} - x_{8k-1} + x_{8k} = 0 \\ x_{8k-7} + x_{8k} = x_{8k-5} + x_{8k-2} \Leftrightarrow x_{8k-7} - x_{8k-5} - x_{8k-2} + x_{8k} = 0 \\ x_{8k-7} + x_{8k} = x_{8k-4} + x_{8k-3} \Leftrightarrow x_{8k-7} - x_{8k-4} - x_{8k-3} + x_{8k} = 0 \end{cases} \quad (75)$$

$$\begin{cases} x_1 - x_2 - x_7 + x_8 = 0 \\ x_1 - x_3 - x_6 + x_8 = 0 \\ x_1 - x_4 - x_5 + x_8 = 0 \\ \vdots \\ x_{57} - x_{58} - x_{63} + x_{64} = 0 \\ x_{57} - x_{59} - x_{62} + x_{64} = 0 \\ x_{57} - x_{60} - x_{61} + x_{64} = 0 \end{cases} \quad (76)$$

6.3.3 Característica restritiva definitiva para as FBF

As características C_4 e C_5 são suficientes para selecionar apenas as FBF para um sistema de $m = 3$ variáveis. No entanto, para $m > 3$ estas restrições se tornam insuficientes, como pode ser constatado, por exemplo, nas 4 funções relativas a um sistema de 4 variáveis apresentadas em (77). Observe que as características C_4 e C_5 podem ser observadas em todas elas, mesmo não sendo FBF. Portanto, é necessário estabelecer uma característica mais efetiva.

$$f = \begin{cases} [\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,] \\ [\,1\,1\,1\,1\,1\,1\,1\,1\,0\,1\,0\,1\,0\,1\,0\,1\,] \\ [\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,0\,0\,1\,1\,1\,1\,] \\ [\,0\,1\,0\,1\,0\,1\,0\,1\,1\,0\,1\,0\,1\,0\,1\,0\,] \end{cases} \Longrightarrow f \notin f^4 \tag{77}$$

▪ *Característica C_6*: Na forma vetorial de toda FBF, em cada sequência de 2^i números, com $2 \leq i \leq m$ a soma do primeiro número com o 2^i-ésimo número é igual à soma do segundo número com o $\left(2^i - 1\right)$-ésimo e, assim por diante, até chegar à igualdade com a soma do 2^{i-1}-ésimo número com o $\left(2^{i-1} + 1\right)$-ésimo número. Sendo $2^m = 2^i \cdot 2^{m-i}$ a quantidade de números de cada função f_i, há 2^{m-i} sequências para cada uma delas e, consequentemente, também há $(2m+2) \cdot 2^{m-i} = (m+1) \cdot 2^{m-i+1}$ sequências relativas a todas as FBF. Logo, há um total de $\left(2^{i-1} - 1\right)(m+1) \cdot 2^{m-2}$ restrições deste tipo para o problema. Sendo $1 < k \leq 2^{i-1} - 1$, em (78) é apresentado o conjunto de restrições relativas a k-ésima sequência de 2^i números.

$$\begin{cases} x_{[(k-1)2^i]} + x_{k2^i} = x_{[(k-1)2^i+1]} + x_{(k2^i-1)} \\ x_{[(k-1)2^i]} + x_{k2^i} = x_{[(k-1)2^i+2]} + x_{(k2^i-2)} \\ \qquad\qquad\qquad \vdots \\ x_{[(k-1)2^i]} + x_{k2^i} = x_{[(k-1)2^i+2^{i-1}]} + x_{[(k-1)2^i+2^{i-1}+1]} \end{cases} \Leftrightarrow$$

$$\Leftrightarrow \begin{cases} x_{[(k-1)2^i]} - x_{[(k-1)2^i+1]} - x_{(k2^i-1)} + x_{k2^i} = 0 \\ x_{[(k-1)2^i]} - x_{[(k-1)2^i+2]} - x_{(k2^i-2)} + x_{k2^i} = 0 \\ \qquad\qquad\qquad \vdots \\ x_{[(k-1)2^i]} - x_{[(k-1)2^i+2^{i-1}]} - x_{[(k-1)2^i+2^{i-1}+1]} + x_{k2^i} = 0 \end{cases} \tag{78}$$

As características C_4 e C_5 representam os dois primeiros casos de uma característica universal. Nas funções que foram apresentadas em (77), as características C4 e C_5 ainda não podiam selecionar corretamente funções com $m > 3$. Atribuindo-lhes estas novas restrições, estas funções agora já não seriam selecionadas. As novas restrições estabelecidas na característica C_6 são auto suficientes para identificar, em um grupo de todas as $2^{(2^m)}$ funções, apenas as $2m + 2$ FBF. Assim, por exemplo, para $m = 3$ devem ser avaliadas as restrições para sequências de 4 e de 8 números da forma vetorial. Para $m = 4$ devem ser avaliadas as restrições para sequências de 4, 8 e 16 números. Quando $m = 5$ devem ser avaliadas as restrições para sequências de 4, 8, 16 e 32 números. De forma generalizada, para m variáveis, devem ser avaliadas as restrições para sequências de 4, 8, ..., 2^i, ... e 2^m números. Destaca-se que estas novas restrições estabelecidas pela característica C_6 são do tipo linear, também contendo apenas coeficientes inteiros 0 ou 1, de modo que continua sendo um problema não linear.

6.3.4 Função objetivo do PPNLIB

Uma razão de alto impacto para ser obtida uma função majoritária minimizada é a redução do custo material de produção, conjuntamente vinculada à otimização do espaço físico da estrutura do circuito. A quantidade de portas lógicas majoritárias em um circuito lógico combinacional está diretamente vinculada ao custo de produção e implementação do circuito físico associado. Paralelamente, a quantidade de níveis da função majoritária correspondente ao circuito também pode ser associada a este custo, podendo-se afirmar que a quantidade de portas lógicas e de níveis constituem um fator preponderante de custo.

Da maneira como foi definida a CCFM, fica estabelecido como o primeiro e segundo critérios de custo a quantidade de níveis e a quantidade de portas lógicas, respectivamente. Deste modo, o custo de implementação de uma função majoritária associada a c é menor que o de uma função majoritária associada a $c + 1$. Portanto, a própria estrutura proposta para o problema de minimização contempla estes dois critérios de custo.

6.3.5 Critério de inversores na forma majoritária

Embora as restrições decorrentes das características das FBF promovam uma enorme redução no espaço de soluções do problema, sempre é possível, devido às propriedades das funções majoritárias, a obtenção de mais de uma solução que atendam a elas. Como exemplo, para a função $f = [\,0\;0\;0\;0\;0\;0\;1\;1\,]$, em decorrência da comutatividade, há um total de 6 soluções, conforme apresentado em (79). No entanto, todas estas soluções correspondem a uma estrutura de 1 nível e 3 entradas, relativas à classe com $c = 1$. Deste modo, as 6 funções majoritárias deste exemplo são equivalentes.

$$M(0,B,C) = M(0,C,B) = M(B,0,C) = M(B,C,0) = M(C,0,B) = M(C,B,0) \quad (79)$$

Para que o problema da obtenção de uma função majoritária simplificada fique caracterizada como um PPNLIB típico, é necessário estipular sua função objetivo, que corresponde a um recurso de norteamento para que se encontre a solução mais adequada. Os parâmetros que podem selecionar qual função é mais efetiva dentro de uma CCFM dependem da tecnologia empregada e de sua aplicação. Um critério utilizado pela comunidade científica é a quantidade de inversores presentes na estrutura majoritária, de modo que se obtenham funções simplificadas contendo a menor quantidade de inversores, sejam nas entradas ou nas próprias portas majoritárias. Assim, embora as funções $M(A,B,\bar{C})$ e $M(A,\bar{B},\bar{C})$ pertençam à mesma CCFM, a segunda é composta por um inversor a mais que a primeira, o que a tornaria mais onerosa. Utilizando este critério de inversão é possível estabelecer uma função objetivo para o problema através de uma propriedade das formas vetoriais das FBF, descrita na Característica C_7.

▪ *Característica C_7*: O primeiro número da forma vetorial de qualquer FBF inversa é 1 e o primeiro número da forma vetorial de qualquer FBF não inversa é 0.

Assim, para reduzir a quantidade de inversores na solução do problema basta reduzir a quantidade de números 1 presentes na primeira variável vetorial das FBF presentes numa solução. Isso equivale a minimizar o valor algébrico do somatório destas variáveis. Portanto, a função objetivo para o PPNLIB é representada por (80).

A função objetivo F é uma função de v variáveis vetoriais, 1 para cada das v variáveis funcionais, representando assim um somatório de v parcelas.

$$F(x_{1\,1}, x_{n+1}, \dots, x_{(v-1)n+1}) = x_1 + x_{n+1} + \cdots + x_{(v-1)n+1} = \sum_{i=0}^{v-1} x_{in+1} \tag{80}$$

Para exemplificar a ação desta função objetivo considere a função f com $m = 5$ e forma vetorial descrita por (81). No processo de obtenção de sua forma majoritária simplificada, se não for considerada a função objetivo proposta, uma das soluções contém o inversor $\bar{A}$. No entanto, se for considerada a função objetivo, a melhor solução não contém inversores. Também deve-se observar que ambas soluções são caracterizadas por $c = 3$.

$$\begin{cases} f = [\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,0\,0\,0\,0\,0\,1\,1\,1\,0\,1\,1\,1\,0\,1\,1\,1\,] \\ f = M(A, M(0, B, \bar{A}), M(C, D, E)) \\ f = M(0, M(A, B, 1), M(C, D, E)) \end{cases} \tag{81}$$

6.3.6 Modelo geral para a solução do problema

Consolidadas a função objetivo de (80), as restrições de (78), que selecionam apenas as FBF, e as restrições de (63), que caracterizam a CCFM, é possível estabelecer um modelo geral para o PPNLIB que descreve a obtenção da função majoritária minimizada a partir de sua forma vetorial $f = [\,b_1\; b_2\; \cdots\; b_n\,]$. Um modelo generalizado para o problema está representado por (82).

A quantidade de variáveis funcionais X_i é v, com $1 \leq i \leq v$ e a quantidade de variáveis vetoriais x_j é $v \cdot 2^m = vn$, com $1 \leq j \leq vn$. Observe que o elemento da forma x_{ij} contida nas restrições refere-se a uma variável vetorial relativa ao número b_i e correspondente a um fator do produto presente numa das parcelas de um dos somatórios. No próximo capítulo é apresentada uma proposta de implementação deste modelo.

$$
\begin{cases}
min \sum_{i=0}^{v-1} x_{in+1} \\
sujeito\ a: \\
x_{[(k-1)2^i]} - x_{[(k-1)2^i+1]} - x_{(k2^i-1)} + x_{k2^i} = 0 \\
x_{[(k-1)2^i]} - x_{[(k-1)2^i+2]} - x_{(k2^i-2)} + x_{k2^i} = 0 \\
\vdots \\
x_{[(k-1)2^i]} - x_{[(k-1)2^i+2^{i-1}]} - x_{[(k-1)2^i+2^{i-1}+1]} + x_{k2^i} = 0;\ 1 < k \leq 2^{i-1} - 1 \\
se\ b_1 = 0, \sum_{i=1}^{p} \left[\left(\prod_{j=1}^{q_i} x_{1j} \right)_i \right] = 0 \qquad se\ b_1 = 1, \sum_{i=1}^{p} \left[\left(\prod_{j=1}^{q_i} x_{1j} \right)_i \right] \geq 1 \\
se\ b_2 = 0, \sum_{i=1}^{p} \left[\left(\prod_{j=1}^{q_i} x_{2j} \right)_i \right] = 0 \qquad se\ b_2 = 1, \sum_{i=1}^{p} \left[\left(\prod_{j=1}^{q_i} x_{2j} \right)_i \right] \geq 1 \\
\vdots \qquad\qquad\qquad\qquad \vdots \\
se\ b_n = 0, \sum_{i=1}^{p} \left[\left(\prod_{j=1}^{q_i} x_{nj} \right)_i \right] = 0 \qquad se\ b_n = 1, \sum_{i=1}^{p} \left[\left(\prod_{j=1}^{q_i} x_{nj} \right)_i \right] \geq 1 \\
x_j \in \{0,1\}; 1 \leq j \leq vn \\
n = 2^m
\end{cases}
\tag{82}
$$

7 USO DE LINGUAGEM COMPUTACIONAL E DE UM SOLUCIONADOR

Da maneira como foi apresentada, a teoria possui solidez por meio de constatações, lemas e demonstrações. Portanto, de maneira teórica representa um processo válido. No entanto, é fundamental que se possa constatar alguma aplicabilidade dessa teoria. Neste capítulo são apresentadas a metodologia computacional e o uso de um solucionador matemático como uma proposta de implementação

7.1 USO DA LINGUAGEM PYTHON

Um procedimento inicial no desenvolvimento da teoria proposta foi a observação do formato algébrico das funções majoritárias após sua expansão utilizando a relação fundamental. Para as funções com CCFM menores, o processo pode ser realizado manualmente, mas com o aumento no valor de c este procedimento passou a ser manualmente impraticável, tanto pela extensão dos cálculos quanto pela sua confiabilidade. Deste modo, foi imperativo o uso de uma linguagem computacional para viabilizar as análises matemáticas.

Foi escolhida a linguagem de programação *Python* em sua versão 3.10.11. Esta linguagem de alto nível permitiu fácil utilização e autonomia, além de contar com vasta biblioteca de *API's (Application Programming Interface)*. Esta linguagem permitiu o desenvolvimento de programas que atuaram como ferramentas no desenvolvimento e implementação da teoria proposta.

7.1.1 Programas de apoio desenvolvidos

Em (39), (43), (44) e (45) foram apresentadas as formas algébricas para funções com CCFM de valores 2, 3, 4 e 5, respectivamente. As quatro formas foram obtidas com aplicações sucessivas da relação fundamental. É evidente a dificuldade manual neste processo para valores maiores de c. Inicialmente, foi preciso desenvolver um programa que permitisse obter a distribuição das variáveis funcionais X_i na forma algébrica de uma função dada na forma majoritária. Dada uma função majoritária, este programa fornece o posicionamento ordenado destas variáveis numa SOP. Deste modo, este programa atua como uma ferramenta na obtenção de funções booleanas

algébricas a partir de sua forma majoritária. Em (84) é mostrado a utilização deste programa para a obtenção do padrão de formação correspondente a $c = 6$. O programa aplica a relação fundamental quantas vezes forem necessárias até decompor a função majoritária completamente. Deve-se notar que, para uma CCFM, o mesmo padrão de formação para a SOP das variáveis X_i se mantém para as nv SOP das variáveis vetoriais x_j.

$$M(X_1, M(X_2, X_3, X_4), M(X_5, X_6, M(X_7, X_8, X_9))) = X_1X_2X_3 + X_1X_2X_4 + X_1X_3X_4 + X_1X_5X_6 + X_1X_5X_7X_8 + X_1X_5X_7X_9 + X_1X_5X_8X_9 + X_1X_6X_7X_8 + X_1X_6X_7X_9 + X_1X_6X_8X_9 + X_2X_3X_5X_6 + X_2X_3X_5X_7X_8 + X_2X_3X_5X_7X_9 + X_2X_3X_5X_8X_9 + X_2X_3X_6X_7X_8 + X_2X_3X_6X_7X_9 + X_2X_3X_6X_8X_9 + X_2X_4X_5X_6 + X_2X_4X_5X_7X_8 + X_2X_4X_5X_7X_9 + X_2X_4X_5X_8X_9 + X_2X_4X_6X_7X_8 + X_2X_4X_6X_7X_9 + X_2X_4X_6X_8X_9 + X_3X_4X_5X_6 + X_3X_4X_5X_7X_8 + X_3X_4X_5X_7X_9 + X_3X_4X_5X_8X_9 + X_3X_4X_6X_7X_8 + X_3X_4X_6X_7X_9 + X_3X_4X_6X_8X_9 \quad (83)$$

Também foi desenvolvido um programa que atua como uma ferramenta de conversão de uma função na forma majoritária para sua forma vetorial. Assim, por exemplo, em (84) está representada uma função majoritária com $m = 4$ que, ao ser processada pelo programa, é convertida em sua forma vetorial. Esta ferramenta é muito útil na validação dos resultados obtidos pela metodologia proposta neste trabalho. Naturalmente, o desenvolvimento de um programa que pudesse realizar o caminho inverso é um dos objetivos da comunidade científica.

$$M(A, \bar{B}, M(B, C, M(0, B, \bar{A}))) = [\,0\;0\;0\;0\;0\;1\;1\;0\;0\;0\;0\;0\;0\;1\;1\;0\,] \quad (84)$$

Um terceiro programa foi desenvolvido para que se pudesse verificar de uma forma prática a validade da característica C_6 das FBF. Constitui uma ferramenta exaustiva para selecionar, dentre as $2^{(2^n)}$ funções vetoriais possíveis de um sistema de m variáveis, apenas aquelas que atendam a tal característica, comprovando serem as $2m + 2$ FBF. No capítulo 8 são apresentados os resultados obtidos com este programa.

Também foi criado um programa para produção de bancos de dados para testes da metodologia. Representa uma ferramenta que, a partir das formas vetoriais das

FBF, aplica a relação fundamental bit a bit, obtendo a forma vetorial equivalente. Em (85) é representada a obtenção de uma função vetorial de $m = 3$ variáveis com CCFM de valor 1 e que não pode ser representada por uma FBF. Estes resultados são separados em grupos e comparados com outros grupos de menor complexidade para serem retiradas as repetições, visto que sempre é possível representar uma função por meio de outra função de maior complexidade. Por comodidade, estas formas vetoriais são armazenadas na forma horizontal.

$$M(A,B,C) = \begin{bmatrix}0\\1\\0\\1\\0\\1\\0\\1\end{bmatrix} \cdot \begin{bmatrix}0\\0\\1\\1\\0\\0\\1\\1\end{bmatrix} + \begin{bmatrix}0\\1\\0\\1\\0\\1\\0\\1\end{bmatrix} \cdot \begin{bmatrix}0\\0\\0\\0\\1\\1\\1\\1\end{bmatrix} + \begin{bmatrix}0\\0\\1\\1\\0\\0\\1\\1\end{bmatrix} \cdot \begin{bmatrix}0\\0\\0\\0\\1\\1\\1\\1\end{bmatrix} = \begin{bmatrix}0\cdot 0+0\cdot 0+0\cdot 0\\1\cdot 0+1\cdot 0+0\cdot 0\\0\cdot 1+0\cdot 0+1\cdot 0\\1\cdot 1+1\cdot 0+1\cdot 0\\0\cdot 0+0\cdot 1+0\cdot 1\\1\cdot 0+1\cdot 1+0\cdot 1\\0\cdot 1+0\cdot 1+1\cdot 1\\1\cdot 1+1\cdot 1+1\cdot 1\end{bmatrix} = \begin{bmatrix}0\\0\\0\\1\\0\\1\\1\\1\end{bmatrix} \tag{85}$$

7.2 EMPREGO DE UM SOFTWARE SOLUCIONADOR

Para implementar e validar a metodologia proposta neste trabalho utilizou-se o software *IBM ILOG CPLEX Optimization Studio*, popularmente conhecido por *CPLEX*, em sua versão 22.1.0.0. O CPLEX é um poderoso programa para a otimização de tomadas de decisões, que se utiliza de programação matemática. Para realizar a modelagem do problema na linguagem Python e sua intercomunicação com o CPLEX utilizou-se a *DOcplex Python Modeling API*, em sua versão 2.28.240. O DOcplex é uma API que representa uma biblioteca de modelagem nativa em Python, composta por dois módulos: modelagem de programação matemática e modelagem de programação com restrições. O uso desta API permite a modelagem e a resolução de problemas de programação matemática com o CPLEX via código em Python, sem a necessidade do uso de janelas do programa.

7.2.1 Recurso de variáveis vetoriais auxiliares

A partir da CCFM de valor 1, todas as demais classes geram um PPNLIB. Como visto em (80) a função objetivo é linear, assim como as restrições relativas à caracterização das FBF (característica C_6), conforme descrito por (78). Todas as restrições relativas ao padrão de formação da função majoritária das classes são não lineares. Para $c = 1$ as restrições são formadas apenas por produtos de 2 variáveis vetoriais, decorrentes de uma única aplicação da relação fundamental na obtenção do padrão da restrição. Como apresentado em (39), para $c = 2$ as restrições são formadas por produtos de 2 e 3 variáveis vetoriais. Em (43) e (44) pode-se observar, respectivamente, que as restrições para $c = 3$ e $c = 4$ são constituídas por produtos de 2, 3 e 4 variáveis vetoriais.

Deste modo, a implementação de qualquer problema relativo a $c > 1$ deparou-se com uma limitação do CPLEX, visto que este opera com parcelas formadas por produtos de apenas 2 variáveis. A simples tentativa de simular um problema com apenas uma restrição formada por um único produto de 3 variáveis retorna em um erro de operação. Para contornar essa limitação, inseriu-se no programa de modelagem o princípio de variável auxiliar, de modo que para cada produto com mais de 2 variáveis vetoriais, é criada uma variável vetorial auxiliar correspondente ao produto de outras 2. Este procedimento é realizado quantas vezes forem necessárias até que se obtenham apenas restrições formadas por produtos de 2 variáveis.

Para exemplificar este procedimento, (86) mostra como se transforma um produto de 3 variáveis vetoriais em um produto de 2, utilizando 1 variável vetorial auxiliar. Em (87) aplica-se o mesmo princípio para transformar um produto de 4 variáveis vetoriais em um produto de 2, utilizando 2 variáveis vetoriais auxiliares. Já em (88) utilizam-se 3 variáveis vetoriais auxiliares para transformar um produto de 5 variáveis vetoriais em um produto de 2. Deste modo, qualquer conjunto de restrições formadas por produtos com mais de 2 variáveis auxiliares podem ser representados, equivalentemente, em produtos de apenas 2 variáveis. Este recurso foi amplamente utilizado no programa desenvolvido em Python para a implementação do modelo do PPNLIB.

$$x_r x_s x_t = x_u x_t \Leftrightarrow x_r x_s = x_u \Leftrightarrow x_r x_s - x_u = 0 \tag{86}$$

$$x_r x_s x_t x_u = x_v x_w \Leftrightarrow \begin{cases} x_r x_s = x_v \\ x_t x_u = x_w \end{cases} \Leftrightarrow \begin{cases} x_r x_s - x_v = 0 \\ x_t x_u - x_w = 0 \end{cases} \quad (87)$$

$$x_r x_s x_t x_u x_v = x_w x_y x_v = x_z x_v \Leftrightarrow \begin{cases} x_r x_s = x_w \\ x_t x_u = x_y \\ x_w x_y = x_z \end{cases} \Leftrightarrow \begin{cases} x_r x_s - x_w = 0 \\ x_t x_u - x_y = 0 \\ x_w x_y - x_z = 0 \end{cases} \quad (88)$$

7.3 MODELAGEM PARA A PRESENÇA DE DON'T CARE STATE

Quando se pretende obter um circuito lógico combinacional para representar um sistema, nem sempre se está interessado em todas as possibilidades que podem ser obtidas com as entradas. Assim, podem ocorrer situações em que não se tem um valor decisivo de saída para certas combinações de entradas. A cada situação em que este estado ocorre, dá-se o nome de *don't care state*, que corresponde a um estado de irrelevância da saída (Tocci; Widmer; Moss, 2019). Deste modo, alguns circuitos lógicos podem ser projetados para que em certas condições de entrada não haja níveis de saída especificados.

De maneira geral, o *don't care state* está vinculado à existência de três situações particulares: a condição não ocorre, a condição é irrelevante ou não se conhece a resposta a ela (Ercegovac; Lang; Moreno, 1999). Tanto na tabela-verdade quanto na forma vetorial, estas condições são comumente indicadas por um X. Assim, por exemplo, na função $f = [\,0\;1\;X\;0\;0\;0\;0\;X\,]$, relativa a um sistema de $m = 3$ variáveis há 2 condições de *don't care state*, de modo que se tem $b_3 = b_8 = X$.

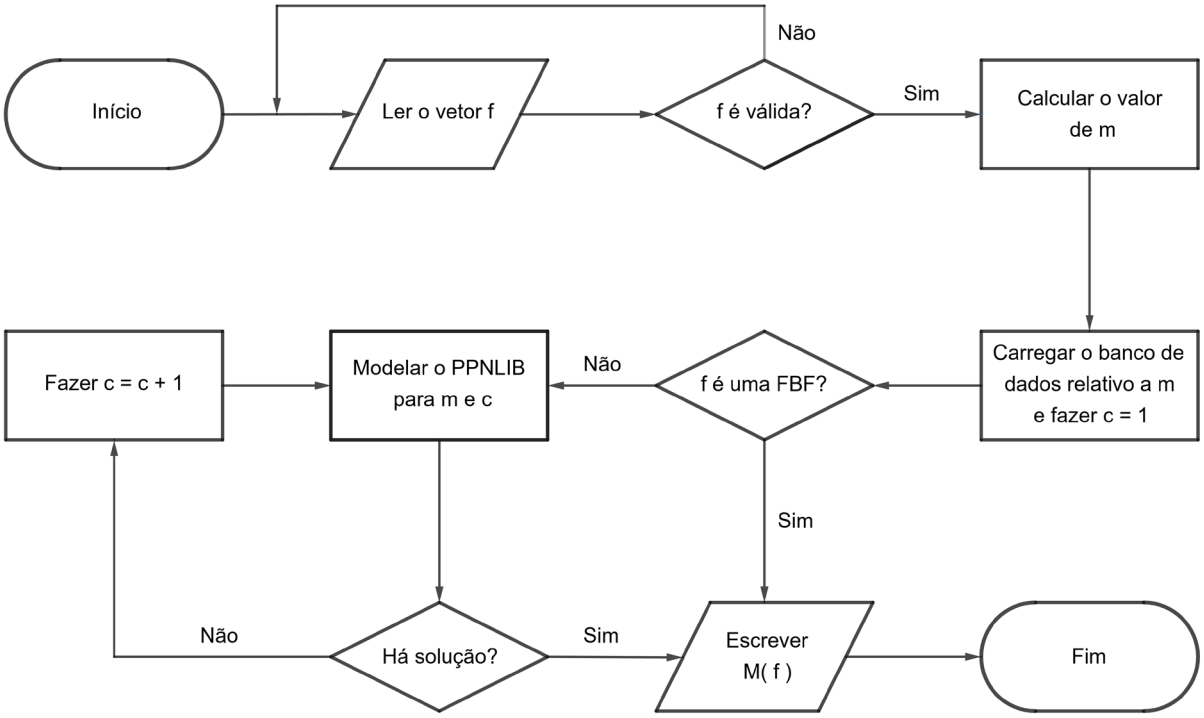
Para lidar com situações da obtenção de uma função majoritária minimizada a partir de uma forma vetorial contendo don't care states parte-se do raciocínio de que, uma vez que o valor 0 ou 1 da saída não é previamente estabelecido, esta condição não deve ser imperativa na resolução do problema. Assim, procede-se eliminando a restrição correspondente a estes estados indefinidos. Portanto, no modelo do PPNLIB, se um elemento b_i da forma vetorial for X, a esta posição i não corresponde uma restrição r_i, conforme descrito em (89). No próximo capítulo é relatado um exemplo de aplicação desta modelagem.

$$b_i = X \Longrightarrow \nexists\, r_i; 1 \leq i \leq n = 2^m \quad (89)$$

7.4 ALGORITMO DE IMPLEMENTAÇÃO

No processo de obtenção da forma majoritária minimizada a partir de uma forma vetorial o trabalho mais extenso ocorre na construção da arquitetura das restrições do PPNLIB. Uma vez modelado o problema, a metodologia é simples e pode ser representada pelo fluxograma esquematizado na Figura 22. Todo o processo pode ser resumido nas quatro etapas descritas a seguir.

Figura 22 – Fluxograma para o algoritmo de solução do PPNLIB.



Fonte: Produção do próprio autor.

▪ Etapa 1: O usuário do programa informa uma sequência formada pelos caracteres 0, 1 ou X, dispostos de maneira justaposta com quantidade tal que $n = 2^m$, que correspondem à forma vetorial da função f. Se for informado algo diferente disso, o programa irá relatar um erro e pedir que seja digitada uma entrada válida. Uma vez que tenha sido digitada uma sequência válida, a partir da quantidade n de caracteres, é determinado o valor de $m = log_2 n$.

▪ Etapa 2: Conhecido o valor de m, são armazenadas no programa as $2m + 2$ FBF, já previamente arquivadas no sistema em sua forma vetorial. É feita uma comparação

de f com estas funções. Se f for uma FBF, o problema está resolvido e é exibida a função do tipo $f_i \in f^m$ correspondente. Este é o único caso em que f não precisa de uma porta lógica majoritária para ser representada.

▪ Etapa 3: Se f não for uma FBF, a partir dos dados já arquivados no sistema, são carregadas as informações para uma modelagem do PPNLIB relativo a $c = 1$ e o solucionador CPLEX é acionado. Se for encontrada uma solução, esta é convertida para a forma majoritária, o problema está encerrado e o programa retorna ao início, agora pronto para receber uma nova função problema.

▪ Etapa 4: Quando o programa não encontrar uma solução para um problema modelado para um certo valor de c, então são carregadas as informações para a modelagem do problema relativo a $c + 1$. Este processo pode se repetir até que o programa informe que não foi capaz de encontrar uma solução, retornando ao início, pronto para receber um novo problema.

8 EXEMPLOS DE APLICAÇÃO E RESULTADOS

Neste capítulo são apresentados alguns exemplos de aplicação da metodologia desenvolvida e um conjunto de resultados para todos os testes realizados. Os resultados obtidos referem-se ao uso de um microcomputador desktop utilizando um sistema operacional *Windows 10 Pro* de 64 bits com processador *Intel Core i3* com frequência de 3,07 GHz. A memória RAM instalada é do tipo DDR-3 com 8 GB.

8.1 VALIDAÇÃO DAS CARACTERÍSTICAS RESTRITIVAS DAS FBF

No desenvolvimento desta teoria foi essencial o estabelecimento das restrições que permitem diferenciar as FBF das demais funções. Isso foi estabelecido de forma generalizada pela característica C_6. Tal característica foi proposta e descrita matematicamente com solidez em sua validade. No entanto, como uma forma de comprovação prática, utilizou-se um dos programas desenvolvidos para testar todas as funções de 3, 4 e 5 variáveis. O programa aplica a característica C_6 em todas as funções na forma vetorial, separando aquelas que atendem às restrições que foram estabelecidas.

Para $m = 3$ variáveis, o programa comprovou rapidamente que apenas as $2m + 2 = 8$ FBF atendiam à característica C_6. Rapidamente, o programa também verificou que, para $m = 4$ variáveis, apenas as $2m + 2 = 10$ FBF atendiam à característica C_6. Após vários dias de processamento, o programa testou todas as funções para $m = 5$ variáveis e comprovou que apenas as $2m + 2 = 12$ FBF atendiam à característica C_6. Estes resultados estão descritos na Tabela 22. Esta validação concedeu credibilidade à sequência da pesquisa que conduziu à modelagem final do problema em questão.

Tabela 22 – Validação das características das FBF.

VARIÁVEIS	TOTAL DE FUNÇÕES	FBF
3	256	8
4	65.536	10
5	4.294.967.296	12

Fonte: Produção do próprio autor.

8.2 EXEMPLO DE APLICAÇÃO PARA UMA FBF

Como primeiro exemplo de aplicação prática do programa desenvolvido para implementar a teoria proposta, considere uma função f em sua forma vetorial, tal que $f = [\,1\,1\,1\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,]$. Não havendo erros na digitação da sequência, o programa identifica que há $n = 32$ bits, correspondendo ao sistema de $m = 5$ variáveis. A seguir, são carregadas e armazenadas em uma lista as $2m + 2 = 12$ FBF. A cada forma vetorial da lista de FBF está associada a função correspondentemente ordenada da lista $\{\,0, A, B, C, D, E, 1, \bar{A}, \bar{B}, \bar{C}, \bar{D}, \bar{E}\,\}$. É verificado que f pertence ao 11º vetor da lista de FBF e é fornecida sua forma algébrica, conforme mostrado em (90). Observe que para casos como este não há necessidade da modelagem de um problema de programação, nem a necessidade de uma porta lógica majoritária para sua implementação física.

$$f = [\,1\,1\,1\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,] = \bar{D} \tag{90}$$

8.3 EXEMPLO DE APLICAÇÃO SEM USO DE VARIÁVEL AUXILIAR

Considere agora o problema de obter a forma majoritária minimizada da função $f = [\,0\,0\,0\,0\,0\,1\,0\,1\,]$. O programa identifica que há $n = 8$ bits na forma vetorial, correspondendo a um sistema de $m = 3$ variáveis. São armazenadas as $2m + 2 = 8$ FBF e constatado que f não é nenhuma delas. Assim, é modelado o problema de programação referente à CCFM com $c = 1$, que está representado por (91). O problema contém 8 restrições relativas ao padrão de funções $f = M(X_1, X_2, X_3)$ e 15 restrições reativas à característica C_6, sendo 6 referentes às sequências de 4 números e 9 referentes às sequências de 8 números.

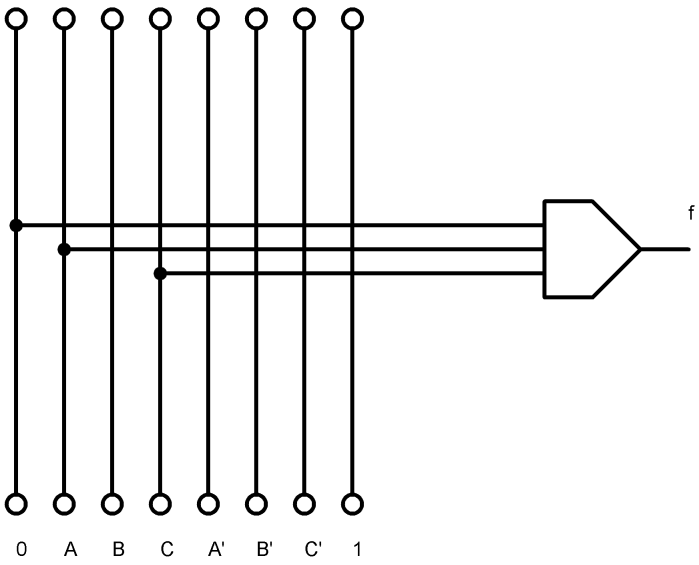
Portanto, o problema é formado por 3 variáveis funcionais X_i e 24 variáveis vetoriais x_j. O solucionador CPLEX encontra a solução para o problema, confirmando que se trata de uma função com $c = 1$. A solução do problema e a forma majoritária minimizada são apresentadas em (92). Em (93) está demonstrada a validação da solução obtida. Na Figura 23 está representado o circuito lógico combinacional majoritário minimizado para a função f.

$$
\begin{cases}
min\ x_1 + x_9 + x_{17} \\
sujeito\ a: \\
x_1x_9 + x_1x_{17} + x_9x_{17} = 0 \\
x_2x_{10} + x_2x_{18} + x_{10}x_{18} = 0 \\
x_3x_{11} + x_3x_{19} + x_{11}x_{19} = 0 \\
x_4x_{12} + x_4x_{20} + x_{12}x_{20} = 0 \\
x_5x_{13} + x_5x_{21} + x_{13}x_{21} = 0 \\
x_6x_{14} + x_6x_{22} + x_{14}x_{22} \geq 1 \\
x_7x_{15} + x_7x_{23} + x_{15}x_{23} = 0 \\
x_8x_{16} + x_8x_{24} + x_{16}x_{24} \geq 1 \\
x_1 - x_2 - x_3 + x_4 = 0 \\
x_5 - x_6 - x_7 + x_8 = 0 \\
x_9 - x_{10} - x_{11} + x_{12} = 0 \\
x_{13} - x_{14} - x_{15} + x_{16} = 0 \\
x_{17} - x_{18} - x_{19} + x_{20} = 0 \\
x_{21} - x_{22} - x_{23} + x_{24} = 0 \\
x_1 - x_2 - x_7 + x_8 = 0 \\
x_1 - x_3 - x_6 + x_8 = 0 \\
x_1 - x_4 - x_5 + x_8 = 0 \\
x_9 - x_{10} - x_{15} + x_{16} = 0 \\
x_9 - x_{11} - x_{14} + x_{16} = 0 \\
x_9 - x_{12} - x_{13} + x_{16} = 0 \\
x_{17} - x_{18} - x_{23} + x_{24} = 0 \\
x_{17} - x_{19} - x_{22} + x_{24} = 0 \\
x_{17} - x_{20} - x_{21} + x_{24} = 0 \\
x_j \in \{0,1\}; 1 \leq j \leq 24
\end{cases}
\tag{91}
$$

$$
f = [\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ ] \Longrightarrow \begin{cases} X_1 = [\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ ] = 0 \\ X_2 = [\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ ] = A \\ X_3 = [\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ ] = C \end{cases} \Longrightarrow f = M(0, A, C) \tag{92}
$$

$$\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix}\cdot\begin{bmatrix}0\\1\\0\\1\\0\\1\\0\\1\end{bmatrix}+\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix}\cdot\begin{bmatrix}0\\0\\0\\0\\1\\1\\1\\1\end{bmatrix}+\begin{bmatrix}0\\1\\0\\1\\0\\1\\0\\1\end{bmatrix}\cdot\begin{bmatrix}0\\0\\0\\0\\1\\1\\1\\1\end{bmatrix}=\begin{bmatrix}0\cdot0+0\cdot0+0\cdot0\\0\cdot1+0\cdot0+1\cdot0\\0\cdot0+0\cdot0+0\cdot0\\0\cdot1+0\cdot0+1\cdot0\\0\cdot0+0\cdot1+0\cdot1\\0\cdot1+0\cdot1+1\cdot1\\0\cdot0+0\cdot1+0\cdot1\\0\cdot1+0\cdot1+1\cdot1\end{bmatrix}=\begin{bmatrix}0+0+0\\0+0+0\\0+0+0\\0+0+0\\0+0+0\\0+0+1\\0+0+0\\0+0+1\end{bmatrix}=\begin{bmatrix}0\\0\\0\\0\\0\\1\\0\\1\end{bmatrix} \quad (93)$$

Figura 23 – Circuito lógico para a função $f = [\,0\,0\,0\,0\,0\,1\,0\,1\,]$.



Fonte: Produção do próprio autor.

8.4 EXEMPLO DE APLICAÇÃO COM USO DE VARIÁVEL AUXILIAR

Considere o problema de obter a forma majoritária minimizada de uma função $f = [\,0\,1\,0\,0\,0\,1\,0\,0\,0\,1\,0\,0\,1\,1\,0\,1\,]$. Ao entrar no programa com a forma vetorial de f, são identificados $n = 16$ bits e $m = 4$ variáveis. São carregadas as $2m + 2 = 10$ FBF e é verificado que f não é uma delas. A seguir é modelado o problema de programação para $c = 1$, que não retorna uma solução. Portanto, agora é modelado o problema de programação para $c = 2$, que está representado por (94) e corresponde ao padrão $f = M(X_1, X_2, M(X_3, X_4, X_5))$.

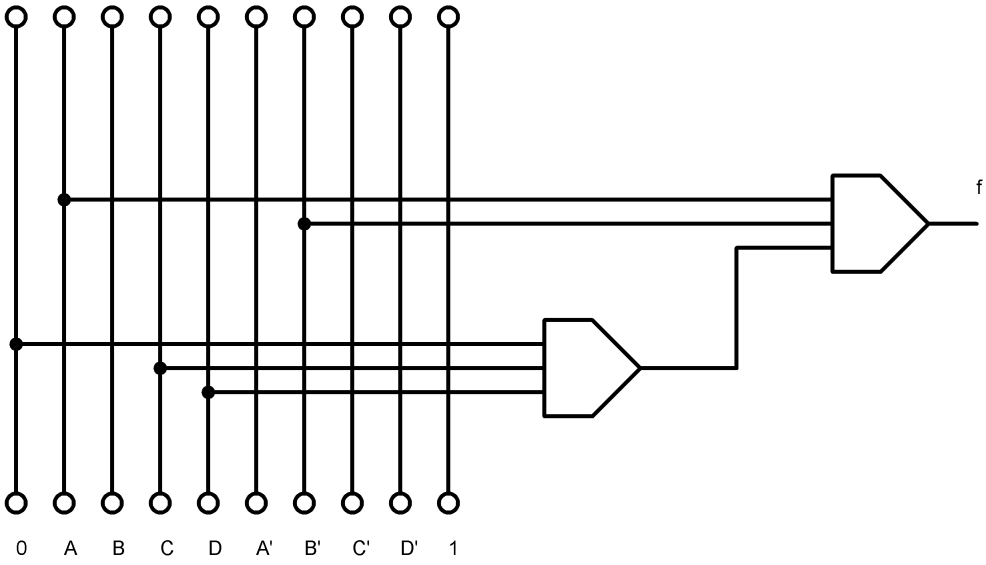
$$
\begin{cases}
min\, x_1 + x_{17} + x_{33} + x_{49} + x_{65} \\
sujeito\ a: \\
x_{33}x_{49} - x_{81} = 0 \\
\quad\vdots \\
x_{48}x_{64} - x_{96} = 0 \\
x_{33}x_{65} - x_{97} = 0 \\
\quad\vdots \\
x_{48}x_{80} - x_{112} = 0 \\
x_{48}x_{65} - x_{113} = 0 \\
\quad\vdots \\
x_{64}x_{80} - x_{128} = 0 \\
x_1x_{17} + x_1x_{81} + x_1x_{97} + x_1x_{113} + x_{17}x_{81} + x_{17}x_{97} + x_{17}x_{113} = 0 \\
\qquad\qquad\qquad\qquad\qquad\qquad\vdots \\
x_{16}x_{32} + x_{16}x_{96} + x_{16}x_{112} + x_{16}x_{128} + x_{32}x_{96} + x_{32}x_{112} + x_{32}x_{128} \geq 1 \\
x_1 - x_2 - x_3 + x_4 = 0 \\
\quad\vdots \\
x_{77} - x_{78} - x_{79} + x_{80} = 0 \\
x_1 - x_2 - x_7 + x_8 = 0 \\
\quad\vdots \\
x_{73} - x_{76} - x_{77} + x_{80} = 0 \\
x_1 - x_2 - x_{15} + x_{16} = 0 \\
\quad\vdots \\
x_{65} - x_{72} - x_{73} + x_{80} = 0 \\
x_j \in \{0,1\}; 1 \leq j \leq 128
\end{cases}
\tag{94}
$$

O problema é formado por 5 variáveis funcionais e 3 variáveis funcionais auxiliares, 80 variáveis vetoriais e 48 variáveis vetoriais auxiliares. Há 48 restrições para criar as variáveis auxiliares, 16 restrições relativas ao modelo de $c = 2$, 20 restrições relativas à característica das sequências com 4 números, 40 restrições relativas à característica das sequências com 8 números e 40 restrições relativas à característica das sequências com 16 números. A solução é apresentada em (95) e na Figura 24 é mostrado o circuito lógico majoritário correspondente.

$$f = [\,0\,1\,0\,0\,0\,1\,0\,0\,0\,1\,0\,0\,1\,1\,0\,1\,] \Longrightarrow \begin{cases} X_1 = [\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,] = A \\ X_2 = [\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,] = \bar{B} \\ X_3 = [\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,] = 0 \\ X_4 = [\,0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,1\,1\,1\,1\,] = C \\ X_5 = [\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,1\,1\,] = D \end{cases} \Longrightarrow$$

$$\Longrightarrow f = M(A, \bar{B}, M(0, C, D)) \tag{95}$$

Figura 24 – Circuito lógico para a função $f = [\,0\,1\,0\,0\,0\,1\,0\,0\,0\,1\,0\,0\,1\,1\,0\,1\,]$.



Fonte: Produção do próprio autor.

8.5 EXEMPLO DE APLICAÇÃO COM A PRESENÇA DE DON'T CARE STATE

Considere o problema de obter a função $f = [\,X\,0\,X\,1\,0\,1\,0\,1\,]$ em sua forma majoritária minimizada. A função apresenta 2 *don't care states*, um no bit b_1 e outro no bit b_3. Ao ler essa entrada vetorial, o programa identifica $n = 8$ bits e $m = 3$ variáveis. Ao modelar o problema de programação para o padrão de formação $f = M(X_1, X_2, X_3)$, correspondente a $c = 1$, o solucionador não encontra solução. Então é modelado o problema de programação para o padrão $M(X_1, X_2, M(X_3, X_4, X_5))$, referente a $c = 2$, que está representado por (96).

Este problema é formado por 5 variáveis funcionais e 3 variáveis funcionais auxiliares, correspondendo a 40 variáveis vetoriais e 24 variáveis vetoriais auxiliares. Há 24 restrições relativas à criação das variáveis vetoriais auxiliares. Como ocorrem

2 *don't care states*, há 6 restrições relativas à CCFM ao invés de 8. Há 10 restrições relativas às sequências de 4 números e 20 relativas às sequências de 8 números.

$$\begin{cases} min\, x_1 + x_9 + x_{17} \\ sujeito\ a: \\ x_{17}x_{25} - x_{41} = 0 \\ x_{18}x_{26} - x_{42} = 0 \\ \quad \vdots \\ x_{31}x_{39} - x_{63} = 0 \\ x_{32}x_{40} - x_{64} = 0 \\ x_2x_{10} + x_2x_{42} + x_2x_{50} + x_2x_{58} + x_{10}x_{42} + x_{10}x_{50} + x_{10}x_{58} = 0 \\ x_4x_{12} + x_4x_{44} + x_4x_{52} + x_4x_{60} + x_{12}x_{44} + x_{12}x_{52} + x_{12}x_{60} \geq 1 \\ x_5x_{13} + x_5x_{45} + x_5x_{53} + x_5x_{61} + x_{13}x_{45} + x_{13}x_{53} + x_{13}x_{61} = 0 \\ x_6x_{14} + x_6x_{46} + x_6x_{54} + x_6x_{62} + x_{14}x_{46} + x_{14}x_{54} + x_{14}x_{62} \geq 1 \\ x_7x_{15} + x_7x_{47} + x_7x_{55} + x_7x_{63} + x_{15}x_{47} + x_{15}x_{55} + x_{15}x_{63} = 0 \\ x_8x_{16} + x_8x_{48} + x_8x_{56} + x_8x_{64} + x_{16}x_{48} + x_{16}x_{56} + x_{16}x_{64} \geq 1 \\ x_1 - x_2 - x_3 + x_4 = 0 \\ x_5 - x_6 - x_7 + x_8 = 0 \\ \quad \vdots \\ x_{57} - x_{58} - x_{59} + x_{60} = 0 \\ x_{61} - x_{62} - x_{63} + x_{64} = 0 \\ x_1 - x_2 - x_7 + x_8 = 0 \\ x_1 - x_3 - x_4 + x_8 = 0 \\ \quad \vdots \\ x_{57} - x_{59} - x_{62} + x_{64} = 0 \\ x_{57} - x_{60} - x_{61} + x_{64} = 0 \\ x_j \in \{0,1\}; 1 \leq j \leq 64 \end{cases} \tag{96}$$

Para esta CCFM o programa encontrou a solução apresentada em (97), que é confirmada por (98), utilizando a expansão algébrica de (39). Observe que, para obter a forma majoritária mais otimizada, obteve-se $b_1 = 0$ e $b_3 = 0$, correspondendo ao circuito lógico combinacional apresentado na Figura 25. Outras soluções atenderiam

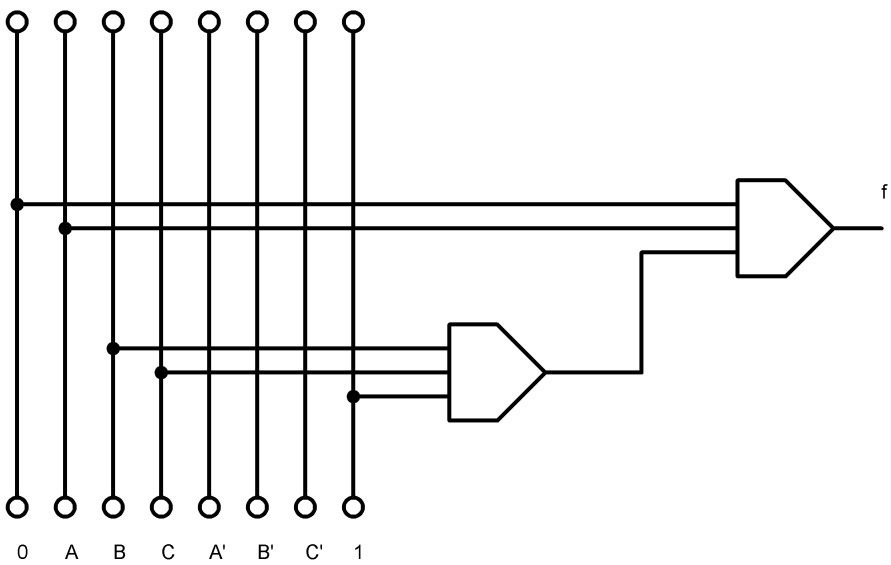
à função de forma não otimizada, como por exemplo a solução proposta por (99), em que $b_1 = 1$ e $b_3 = 0$, referente a $c = 4$ e que possui o circuito lógico correspondente representado na Figura 26.

$$f = [\, X\ 0\ X\ 1\ 0\ 1\ 0\ 1\,] \Longrightarrow \begin{cases} X_1 = [\,0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\,] = 0 \\ X_2 = [\,0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\,] = A \\ X_3 = [\,0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\,] = B \\ X_4 = [\,0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\,] = C \\ X_5 = [\,1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\,] = 1 \end{cases} \Longrightarrow f = M(0, A, M(B, C, 1)) \tag{97}$$

$$\begin{bmatrix} 0\cdot 0+0\cdot 0\cdot 0+0\cdot 0\cdot 1+0\cdot 0\cdot 1+0\cdot 0\cdot 0+0\cdot 0\cdot 1+0\cdot 0\cdot 1 \\ 0\cdot 1+0\cdot 0\cdot 0+0\cdot 0\cdot 1+0\cdot 0\cdot 1+1\cdot 0\cdot 0+1\cdot 0\cdot 1+1\cdot 0\cdot 1 \\ 0\cdot 0+0\cdot 1\cdot 0+0\cdot 1\cdot 1+0\cdot 0\cdot 1+0\cdot 1\cdot 0+0\cdot 1\cdot 1+0\cdot 0\cdot 1 \\ 0\cdot 1+0\cdot 1\cdot 0+0\cdot 1\cdot 1+0\cdot 0\cdot 1+1\cdot 1\cdot 0+1\cdot 1\cdot 1+1\cdot 0\cdot 1 \\ 0\cdot 0+0\cdot 0\cdot 1+0\cdot 0\cdot 1+0\cdot 1\cdot 1+0\cdot 0\cdot 1+0\cdot 0\cdot 1+0\cdot 1\cdot 1 \\ 0\cdot 1+0\cdot 0\cdot 1+0\cdot 0\cdot 1+0\cdot 1\cdot 1+1\cdot 0\cdot 1+1\cdot 0\cdot 1+1\cdot 1\cdot 1 \\ 0\cdot 0+0\cdot 1\cdot 1+0\cdot 1\cdot 1+0\cdot 1\cdot 1+0\cdot 1\cdot 1+0\cdot 1\cdot 1+0\cdot 1\cdot 1 \\ 0\cdot 1+0\cdot 1\cdot 1+0\cdot 1\cdot 1+0\cdot 1\cdot 1+1\cdot 1\cdot 1+1\cdot 1\cdot 1+1\cdot 1\cdot 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \tag{98}$$

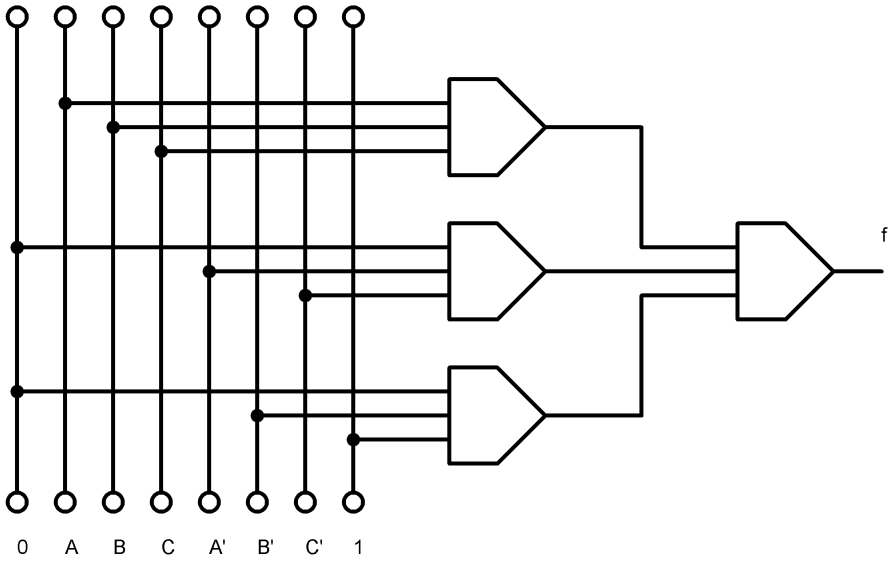
$$f = [\, X\ 0\ X\ 1\ 0\ 1\ 0\ 1\,] = [\,1\ 0\ 0\ 1\ 0\ 1\ 0\ 1\,] = M(M(A, B, C), M(0, \bar{A}, \bar{C}), M(A, \bar{B}, 1)) \tag{99}$$

Figura 25 – Circuito lógico otimizado para a função $f = [\, X\ 0\ X\ 1\ 0\ 1\ 0\ 1\,]$.



Fonte: Produção do próprio autor.

Figura 26 – Circuito lógico não otimizado para a função $f = [\, X\ 0\ X\ 1\ 0\ 1\ 0\ 1\,]$.



Fonte: Produção do próprio autor.

8.6 RESULTADOS GERAIS

Para avaliar a aplicabilidade da teoria desenvolvida na pesquisa foram realizados testes com funções para vários valores m de variáveis e para várias CCFM, de modo a observar o desempenho com relação ao aumento de variáveis, níveis e complexidade das funções.

8.6.1 Resultados para funções de 3 variáveis

Foram testadas todas as 256 funções para $m = 3$ variáveis, de modo que foram encontradas soluções ótimas para todas elas. Na Tabela 23 são apresentados os resultados obtidos, mostrando a classe de complexidade, a quantidade de funções para cada valor de c e o tempo médio para a obtenção da função majoritária minimizada. Deve-se ressaltar que nenhuma delas requer $c > 4$, de acordo com outras metodologias, como as propostas por Ferraz (2018), Oliveira Júnior (2021) e Ferraz (2022). Naturalmente, devido à forma como foi desenvolvida a metodologia, sempre é possível representar uma destas funções através de outras, de formato redundante e não minimizado, com CCFM superior.

Tabela 23 – Testes com funções de 3 variáveis.

CCFM	QUANTIDADE	TEMPO MÉDIO
0	8	0,01 s
1	32	1,77 s
2	64	4,36 s
3	56	22,81 s
4	96	127,53 s

Fonte: Produção do próprio autor.

8.6.2 Resultados para funções de 4 variáveis

Dentre as 65.536 funções possíveis com $m = 4$ variáveis, foram testadas 4.270, incluindo todas as funções possíveis até $c = 3$. Na Tabela 24 estão os resultados obtidos na implementação do solucionador CPLEX. Como pode ser observado, a partir de $c = 4$ os tempos de solução passam a ser elevados e seus valores médios não possuem grande precisão, visto que foram avaliadas poucas funções dentre as possíveis e observou-se que os tempos podem variar muito para um mesmo valor de c, a depender da função.

Tabela 24 – Testes com funções de 4 variáveis.

CCFM	QUANTIDADE	TEMPO MÉDIO
0	10	0 s
1	80	3 s
2	640	5 s
3	3300	2 min 46 s
4	200	14 min 02 s
5	10	28 min 24 s
6	10	1 h 44 min 17 s
7	10	3 h 58 min 44 s
8	10	9 h 24 min 32 s

Fonte: Produção do próprio autor.

8.6.3 Resultados para funções de 5 variáveis

Na Tabela 25 são apresentados os resultados obtidos na implementação do programa na solução de funções com $m = 5$ variáveis. Foram avaliadas todas as funções para $c = 0$ e $c = 1$. A partir de $m > 4$ variáveis, a quantidade de $2^{(2^n)}$ funções possíveis torna-se extremamente elevada, de modo que os valores médios para $c > 1$ podem variar muito de acordo com os lotes de funções selecionados para os testes.

Tabela 25 – Testes com funções de 5 variáveis.

CCFM	QUANTIDADE	TEMPO MÉDIO
0	12	0 s
1	160	5 s
2	100	22 s
3	100	5 min 47 s
4	10	17 min 39 s
5	5	1 h 43 min 25 s
6	5	5 h 50 min 25 s
7	5	10 h 09 min 01 s
8	5	14 h 32 min 06 s

Fonte: Produção do próprio autor.

8.6.4 Comparando soluções variando a quantidade de variáveis

Para realizar uma análise horizontal de profundidade do método, foram realizados testes com funções de até 10 variáveis para CCFM de até quinta ordem. Como os tempos computacionais na obtenção das soluções são muito altos para m elevado, para estes casos foram testadas poucas funções. Os resultados estão apresentados na Tabela 26. Note que para $m = 3$ não há funções com $c = 5$.

8.6.5 Explorando o aumento de variáveis

Para explorar a capacidade do método também foram realizados alguns testes isolados e puntuais para $c = 16$ e até para padrões de 5 níveis. Para explorar o método com muitas variáveis, foram realizados testes para m variando de 3 a 15 na solução

de funções com $c = 1$. Estes resultados são apresentados na Tabela 27 e comprovam que, embora os tempos de solução aumentem muito, o método permite uma exploração vertical com o aumento das variáveis.

Tabela 26 – Comparação entre soluções para vários valores de m.

m	c = 2	c = 3	c = 4	c = 5
3	4 s	23 s	2 min 08 s	-
4	5 s	2 min 46 s	14 min 02 s	28 min 24 s
5	23 s	5 min 47 s	17 min 39 s	1 h 43 min 25 s
6	1 min 10 s	9 min 58 s	42 min 43 s	2 h 44 min 35 s
7	4 min 49 s	13 min 09 s	1 h 12 min 45 s	3 h 26 min 05 s
8	1 h 30 min 26 s	2 h 25 min 45 s	2 h 50 min 37 s	5 h 33 min 57 s
9	2 h 36 min 06 s	3 h 26 min 02 s	5 h 10 min 28 s	7 h 37 min 36 s
10	6 h 36 min 44 s	10 h 53 min 56 s	13 h 12 min 49 s	17 h 01 min 00 s

Fonte: Produção do próprio autor.

Tabela 27 – Soluções para $c = 1$ com até 15 variáveis.

m	QUANTIDADE	TEMPO MÉDIO
3	32	2 s
4	80	3 s
5	160	5 s
6	280	10 s
7	448	18 s
8	672	54 s
9	960	2 min 27 s
10	1320	4 min 36 s
11	100	18 min 45 s
12	100	1 h 49 min 47 s
13	10	7 h 40 min 12 s
14	10	12 h 41 min 38 s
15	10	20 h 56 min 04 s

Fonte: Produção do próprio autor.

9 CONCLUSÃO

Essa pesquisa proporcionou o desenvolvimento de uma teoria matemática consistente e promissora, baseada em proposições e demonstrações, que dentro de certos limites pode ser validada com o uso dos programas desenvolvidos e do solucionador CPLEX. Inicialmente, demonstrou-se que é possível romper com o paradigma de uma quase obrigatoriedade do emprego das funções majoritárias primitivas nos métodos de minimização. Porém, deve-se observar que, independentemente de sua aplicabilidade prática imediata, a teoria possui solidez considerável. Teórica e matematicamente, acredita-se que a teoria desenvolvida permite modelar qualquer problema de obtenção da forma majoritária minimizada de uma função booleana, desde que haja a disponibilidade para desenvolver toda a extensa estrutura. Considera-se que seja uma base de referência para outras metodologias a serem desenvolvidas.

Foi possível implementar a teoria dentro das limitações de mapeamento e processamento computacional. Foram obtidos resultados consistentes que validaram a solidez da metodologia desenvolvida. Destacou-se por explorar sistemas com até 15 variáveis, dos quais alguns com até 5 níveis, pouco abrangidos pela comunidade científica. Tais limitações abrem uma visão expansiva sobre resultados muito mais poderosos que podem ser obtidos se aplicada, por exemplo, em conjunto com a implementação com supercomputação.

REFERÊNCIAS

BAZARAA, M. S.; JARVIS, J. J.; SHERALI, H. D. **Linear programming and network flows.** 4. ed. New Jersey: John Wiley & Sons, 2010. 748 p.

BAZARRA, M. S.; SHERALI, H. D.; SHETTY C. M. **Nonlinear Programming – Theory and Algorithms.** 3. ed. New York: John Wiley & Sons, 2006. 872 p.

BOOLE, G. **An investigation of the laws of thought: on which are founded the mathematical theories of logic and probabilities.** Cork: Dover Publications, 1854. p. 24-38.

CHATTOPADHYAY, A.; AMARÚ, L.; SOEKEN, M.; GAILLARDON, P. E.; MICHELI, G. Notes on majority boolean algebra. In: MULTIPLEVALUED LOGIC - ISMVL, 46., 2016, Sapporo. **Proceedings...** Sapporo: IEEE. 2016. p. 50-55.

CHU, Z. et al. Advanced functional decomposition using majority and its applications. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems**, Lausanne, v. 39, n. 8, p. 1621-1634, 2019.

DAGHLIAN, J. **Lógica e álgebra de boole.** 4. ed. São Paulo: Atlas, 1986. p. 26-28.

ERCEGOVAC, M. D.; LANG, T.; MORENO, J. H. **Introduction to digital systems.** 1. ed. New york: John Wiley & Sons, 1999. p. 25-27.

FERRAZ, E. C. **Combinação de funções primitivas para síntese de funções majoritárias.** 2018. 66 f. Dissertação (Mestrado em Engenharia Elétrica) - Faculdade de Engenharia de Ilha Solteira, Universidade Estadual Paulista, Ilha Solteira, 2018.

FERRAZ, E. C. **Optimization models for majority logic synthesis.** 2022. 76 f. Tese (Doutorado em Engenharia Elétrica) - Faculdade de Engenharia de Ilha Solteira, Universidade Estadual Paulista, Ilha Solteira, 2022.

FRANCO, S. **Analog circuit design: discrete and integrated.** New York: McGraw-Hill Education, 2015. p. 10-14.

KARNAUGH, M. The map method for synthesis of combinational logic circuits. **Transactions of the American Institute of Electrical Engineers, Part I: Communication and Electronics**, New York, v. 72, n. 5, p. 593-599, 1953.

KOHAVI, Z.; JHA, N. K. **Switching and finite automata theory. 3. ed. New York: Cambridge University Press, 2010. p. 46-50.**

LAFFERRIERE, B.; LAFFERRIERE, G.; NGUYEN, M. N. Introduction to Mathematical Analysis I. 3. ed. Portland: Portland State University Library, 2022. p. 19-29.

LAGIDI et al. **Design of 16-bit and 32-bit approximate full adder using majority logic.** 2021 2nd Global Conference for Advancement in Technology (GCAT), Bangalore, p. 1-5, 2021.

LAKSHMI, V.; REUBEN, J.; PUDI, V. A novel in-memory wallace tree multiplier architecture using majority logic. **IEEE Transactions on Circuits and Systems I: Regular Papers**, v. 69, n. 3, p. 1148-1158, 2022.

LEITHOLD, L. **O cálculo com geometria analítica - volume 1**. 3. ed. São Paulo: Harbra, 1994. p. 31-36.

LINDAMAN, R. A theorem for deriving majority-logic networks within an augmented boolean algebra. **IRE Transactions on electronic computers**, New York, v. 9, n. 3, p.338-342, 1960.

MAINI, A. K. **Digital electronics: principles, devices and applications.** West Sussex: John Wiley & Sons, 2007. p. 189-193.

MCCLUSKEY, E. J. Minimization of boolean functions. **Bell Labs Technical Journal, Kansas**, v. 35, n. 6, p. 1417-1444, 1956.

MICHAELIS. **Michaelis: moderno dicionário da língua portuguesa.** 10. ed. São Paulo: Melhoramentos, 2002. p 1299-1300.

MOSS, G. L.; TOCCI, R. J.; WIDMER, N. S. **Digital systems: principles and applications.** 12. ed. Harlow: Pearson Education, 2018.p. 24-27.

MOTOROLA. **Semiconductor technical data: dual 5-input majority gate.** Phoenix: Motorola, 1995. 7 p.

OLIVEIRA JÚNIOR, J. V. **Otimização de funções lógicas majoritárias utilizando programação linear inteira binária e quantificação de primitivas.** 2021. 82 f. Dissertação (Mestrado em Engenharia Elétrica) - Faculdade de Engenharia de Ilha Solteira, Universidade Estadual Paulista, Ilha Solteira, 2021.

RAFIQ, M., CHAUHAN, Y. S., SAHAY, S. Exploiting single ferroelectric FET for efficient implementation of majority gate function for approximate computing. 2024 **8th IEEE Electron Devices Technology & Manufacturing Conference (EDTM)**, Bangalore, 2024. p. 1-3

RAJ, M.; SEKAR, K. R.; LAKSHMINARAYANAN, G. Majority-logic-based self-checking adder in quantum-dot cellular automata. **IEEE Design & Test**, v. 39, n. 5, p. 88-97, 2022.

RIENER, H. et al. Logic optimization of majority-inverter graphs. In: MBMV 2019: **22ND WORKSHOP-METHODS AND DESCRIPTION LANGUAGES FOR MODELLING AND VERIFICATION OF CIRCUITS AND SYSTEMS**, 22., 2019, Kaiserslautern. Proceedings... Kaiserslautern: VDE. 2019. p. 1-4.

ROSEN, K. H. **Discrete mathematics and its applications.** 8. ed. New York: McGraw-Hill Education, 2019. p. 847-853.

SEDRA, A. S.; SMITH, K. C.; CARUSONE, T. C.; GAUDET, V. **Microelectronic circuits.** 8. ed. New York: Oxford University Press, 2020. p. 1110-1119.

SOEKEN, M.; AMARU, L. G.; GAILLARDON, P.; DE MICHELI, G. Exact synthesis of majority-inverter graphs and its applications. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems**, La Jolla, v. 36, n. 11, p. 1842-1855, 2017.

SHANNON, C. E. A symbolic analysis of relay and switching circuits. **Transactions american institute of electrical engineers**, vol. 57, p. 471-495, 1938.

STEWART, J. **Cálculo, volume 2**. 7. ed. São Paulo: Cengage Learning, 2013. p. 792-800.

THOMAS, G. B.; WEIR, M. D.; HASS, J. **Cálculo, volume 1.** 12. ed. São Paulo: Pearson Education do Brasil, 2012. p. 13-17.

TOCCI, R. J.; WIDMER, N. S.; MOSS, G. L. **Sistemas digitais: princípios e aplicações.** 12. ed. São Paulo: Pearson Education do Brasil, 2019. p. 164-166.

TOSHIBA. **C2mos integrated circuits: technical data.** Toshiba America, 1985. p. 389-391.

WAKERLY, J. F. **Digital design: principles and practices.** 4. ed. New Jersey: Pearson Prentice Hall, 2006. p. 18-22.

WANG, P.; NIAMAT, M. Y.; VEMERU, S. R.; ALAM, M; KILLIAN, T. Synthesis of majority/minority logic networks. **IEEE Transactions on Nanotechnology**, Boston, v. 14, n. 3, p. 473-483, 2015.

WIGINGTON, R. L. A new concept in computing. **Proceedings of the IRE**, v. 47, n. 4. pp. 516-523, 1959.

ZHANG, T. et al. A survey of majority logic designs in emerging nanotechnologies for computing. **IEEE Transactions on Nanotechnology**, v. 22, p. 732-739, 2023.

ZORICH, V. A. **Mathematical analysis I**. 4. ed. New York: Springer-Verlag, 2004. p. 35-48.