**UNIVERSIDADE FEDERAL DE ITAJUBÁ - UNIFEI**
**PROGRAMA DE PÓS-GRADUAÇÃO EM**
**CIÊNCIA E TECNOLOGIAS DA INFORMAÇÃO**

**Douglas Sandy Bonafé**

# Utilização de Algorítmos Evolutivos para a Síntese de Circuitos CMOS: o Estado da Arte.

Dissertação submetida ao Programa de Pós-Graduação como parte dos requisitos para obtenção do Título de Mestre em Ciências e Tecnologias da Informação.

**Área de Concentração: Inteligência Artificial e Software e Hardware Básico**

**Orientador: Prof. Dr. Paulo César Crepaldi**

**7 de julho de 2018**
**Itajubá**

**UNIVERSIDADE FEDERAL DE ITAJUBÁ - UNIFEI**
**PROGRAMA DE PÓS-GRADUAÇÃO EM**
**CIÊNCIA E TECNOLOGIAS DA INFORMAÇÃO**

**Douglas Sandy Bonafé**

# Utilização de Algorítmos Evolutivos para a Síntese de Circuitos CMOS: o Estado da Arte.

Dissertação submetida ao Programa de Pós-Graduação como parte dos requisitos para obtenção do Título de Mestre em Ciências e Tecnologias da Informação.

**Área de Concentração: Inteligência Artificial e Software e Hardware Básico**

**Orientador: Prof. Dr. Paulo César Crepaldi**

**7 de julho de 2018**
**Itajubá**

**UNIVERSIDADE FEDERAL DE ITAJUBÁ - UNIFEI**
**PROGRAMA DE PÓS-GRADUAÇÃO EM**
**CIÊNCIA E TECNOLOGIAS DA INFORMAÇÃO**

# Utilização de Algorítmos Evolutivos para a Síntese de Circuitos CMOS: o Estado da Arte.

**Douglas Sandy Bonafé**

Dissertação aprovada por banca examinadora em 22 de Junho de 2018, conferindo ao autor o título de **Mestre em Ciências e Tecnologias da Informação.**

***Banca Examinadora:***
Prof. Dr. José Feliciano Adami
(UNESP - Guaratinguetá)
Prof. Dr. Robson Luiz Moreno.
(UNIFEI - Itajubá)

**Itajubá**
**2018**

Douglas Sandy Bonafé
Utilização de Algorítmos Evolutivos para a Síntese de Circuitos CMOS: o Estado da Arte/ Douglas Sandy Bonafé. – Itajubá, 7 de julho de 2018-
147 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Dr. Paulo César Crepaldi

Dissertação (Mestrado)
Universidade Federal de Itajubá - UNIFEI
Programa de pós-graduação em ciência e tecnologias da informação, 7 de julho de 2018.

1. Palavra-chave1. 2. Palavra-chave2. I. Orientador. II. Universidade xxx. III. Faculdade de xxx. IV. Título

CDU 07:181:009.3

Douglas Sandy Bonafé

# Utilização de Algorítmos Evolutivos para a Síntese de Circuitos CMOS: o Estado da Arte

Dissertação submetida ao Programa de Pós-Graduação como parte dos requisitos para obtenção do Título de Mestre em Ciências e Tecnologias da Informação.

Trabalho aprovado. Itajubá, 22 de Junho de 2018:

---

**Prof. Dr. Paulo César Crepaldi**
Orientador

---

**Prof. Dr. José Feliciano Adami**
**(UNESP - Guaratinguetá)**

---

**Prof. Dr. Robson Luiz Moreno.**
**(UNIFEI - Itajubá)**

Itajubá

7 de julho de 2018

# Agradecimentos

Primeiramente agradeço a Deus, que me deu o intelecto como faculdade da alma, me inspirou por seu Espírito a utilizá-lo bem, me assiste a cada instante de minha vida e, de forma especialíssima, neste trabalho. É Ele o autor da própria realidade, e em cuja infinita Misericórdia, compartilha conosco, criaturas, algo de Si, para que possamos conhecer a Verdade, e compreender a realidade. Agradeço à Igreja Católica, por ter me acolhido no Santo Batismo e me dado a Sã Doutrina a qual me ensinou, dentre tudo o mais, a humildade que me manda aprender de todos e de boa vontade aquilo que desconheço. Sem Ela, os dons de Deus seriam apenas mistérios insondáveis. Agradeço a Hugo de São Vitor, brilhante homem, filósofo, teólogo, Cardeal e autor místico da Idade Média, cuja sabedoria e, em minha devoção particular, santidade, irradiam pelos séculos em suas obras como o *Didascalicon* a qual, dentre muitas outras coisas, ensinaram-me as três coisas necessárias ao estudante: a natureza, o exercício e a disciplina. Agradeço também aos meus pais e irmãos, em especial à minha mãe Iracilda. Ela, juntamente com meu pai Fernando, além de terem-me concedido a vida dando seu corajoso sim a Deus, em cumprimento da missão mais difícil e nobre da vida Matrimonial (a paternidade), também muito me incentivou e não me deixou desistir nas muitas vezes que havia decidido fazê-lo. Agradeço também à minha amadíssima esposa Fernanda, cujo amor e compreensão, as saudosas noites distantes que triste teve que dar-me boa-noite e aqueles momentos que, mesmo fisicamente perto, abdicou de minha atenção e trabalhou por nós dois no seio de nosso lar para que eu pudesse dedicar-me a este trabalho foram de imensurável importância. Sem menos importância agradeço ao meu orientador Paulo César Crepaldi que não apenas foi paciente com minhas dificuldades, mas sempre solícito a resolvê-las, generoso ao ajudar-me e presente em todos os momentos deste trabalho. Sem ele este trabalho jamais seria feito. Também gostaria de agradecer ao professor Edmilson Marmo Moreira que, sendo na época coordenador do POSCOMP, me convidou primeiramente ao programa de pós-graduação da UNIFEI e tantas vezes me ajudou; e ao professor Guilherme Sousa Bastos, atual coordenador, que, acreditando no meu trabalho, permitiu que eu continuasse no curso quando já tinha passado do tempo de conclusão e que tantas vezes auxiliou-me com aspectos burocráticos. Sem os senhores, eu não estaria aqui. Também aos professores Carlos Henrique Valério, Robson Moreno e Tales Cleber Pimenta que me ajudaram em alguns momentos deste projeto. Não por menos, deixo também o agradecimento a tantos membros da comunidade de software, às empresas Concrete Solutions, Datablink Ltda e Wipro e a todos que me auxiliaram e cuja lista não caberia neste pequeno espaço. Por fim, agradeço a cada trabalhador brasileiro cujos altíssimos impostos e taxas mantém, apesar dos altíssimos índices de corrupção, dos desperdícios próprios de um Estado socialista e das políticas globalistas, as escolas e universidades, dentre as quais se encontra Universidade Federal de Itajubá.

*"Omnia legentes, quae bona sunt tenentes."*
*(I Tess. 5)*

**Resumo**

Dimensionar transistores não é uma tarefa trivial, empregando-se diversas técnicas, geralmente através de projeto por partes (blocos), com o auxílio de simulação. Este trabalho tem por objetivo apresentar o estado da arte em matéria de síntese topológica e otimização de circuitos CMOS, assim como utilizar o conhecimento adquirido para elaborar um algorítimo evolutivo capaz de encontrar as dimensões específicas de um transistor CMOS para encontrar os valores ótimos de comprimento e largura do canal do transistor de forma a conseguir-se as dimensões para uma porta lógica com M entradas a fim de que o circuito apresente tempo de subida igual ao tempo de descida e responda no menor tempo possível. Para tanto, optou-se por utilizar algorítimos genéticos, sendo desenvolvido na linguagem Python versão 2.7 do interpretador, e o software CADENCE/Orcad para a validação do circuito. Como resultados deste trabalho obtiveram-se um resumo do estado da arte na área de algorítmos genéticos e uma ampla pesquisa em todas as áreas de contorno a do problema em questão e um prospecto de todas as implicações que as observações feitas pelo estudo dos algoritmos genéticos exercem nas áreas de contorno. Através da validação feita pela simulação com Orcad, provou-se ser perfeitamente possível projetar-se um circuito de alta precisão, otimizado para determinadas condições de contorno, sem a utilização do simulador neste processo (utilizando-o apenas para validação). Não obstante, apresentou-se um comparatativo entre os métodos evolucionários a fim de ajudar o projetista a encontrar o algoritmo que melhor o ajude a solucionar seu problema espefífico. Feito isso, desenvolveu-se um *framework open source*, em linguagem Python, para auxiliar o projetista que deseje implementar estas soluções. Ao têrmo deste, refletiu-se nas implicações filosóficas das observações feitas durante todo a pesquisa e levantou-se questionamentos às teorias evolucionárias (das pressuposições de Darwin às proposições de Orr), apresentando-se assim como uma contribuição importante para as teorias do *Intelligent Design*.

**Palavras-chaves**: ALGORITMO EVOLUTIVO, CIRCUITO CMOS, PORTAS LÓGICAS CMOS, ALGORITMO GENÉTICO, PYTHON.

# Abstract

The sizing of transistors is not a trivial task. To do so, engineers use a lot of techniques, generally doing the project by blocks, with simulation help for adjustments. This work have as objective to present the state of art regarding to topological synthesis and CMOS circuitry optimization, as so as using the acquired knowledge to elaborate an evolutionary algorithm that is able to find the optimal values of length and and width of transistors channel to build with them a logical gate with low-to-high time equals to high-to-low time and the minimum response time when we have a logical circuitry with M gates. For this, was opted to use genetic algorithms, implemented with Python 2.7 and simulated with Cadence Orcad. As result of this research, was attained a complete resume about the state of art in a huge amount of sciences that surrounds the Genetic Algorithm. Through the Orcad validation, was proved that is perfectly possible to develop high precision circuits for certain boundary conditions just by the use of Mathematics probabilities and evolutionary process: using different algorithms solutions, each one having their pros and cons. With these observations, an open-source framework was constructed and made available on the internet. None the less, at the end of the work, the philosophical implications of the necessary assumptions done during the process was reflected and questions raised against the traditional academic approach of the *darwinist* evolutionary approach, positioning this one as one more contribution for the conclusions of the Intelligent Design and its implications.

**Key-words**: EVOLUTIONARY ALGORITHMS, CMOS CIRCUITRY, CMOS LOGICAL GATES, GENETIC ALGORITHM, PYTHON.

# Lista de ilustrações


Figura 1 – Problema Teórico . . . . . 26
Figura 2 – Problema Prático . . . . . 27
Figura 3 – Hipóteses . . . . . 36
Figura 4 – Metodologia Técnica . . . . . 37
Figura 5 – Inversor NMOS com resistor pull-up . . . . . 40
Figura 6 – Fonte de Corrente com PMOS em Pull-up . . . . . 41
Figura 7 – PMOS funcionando como fonte de corrente pull-up no circuito . . . . . 42
Figura 8 – Topologia do Inversor CMOS . . . . . 42
Figura 9 – Curva característica do CMOS . . . . . 44
Figura 10 – Ilustração do tempo de propagação usando uma onda quadrada na entrada do inversor (SODINI, 2017). . . . . 45
Figura 11 – Circuitos inversores acoplados em cascata. . . . . 48
Figura 12 – Seção transversal do inversor . . . . . 49
Figura 13 – Seção transversal do inversor . . . . . 49
Figura 14 – Seção superficial do *Gate* de um iversor . . . . . 50
Figura 15 – Lógica *Pull-Up* . . . . . 53
Figura 16 – Lógica *Pull-Down* . . . . . 53
Figura 17 – Lógica inversora *Pull-Up,Pull-Down* . . . . . 54
Figura 18 – Generalização para circuitos CMOS estáticos complementares (RABAEY; CHANDRAKASAN; NIKOLIC, 2004) . . . . . 55
Figura 19 – Atraso em função do padrão das entradas (FERREIRA, 2004) . . . . . 56
Figura 20 – Resumo dos Tipos de Problemas e onde se encontra o do estudo de caso . . . . . 62
Figura 21 – Método da Roleta Viciada . . . . . 68
Figura 22 – Método de Amostragem Estocástica Universal . . . . . 69
Figura 23 – Seleção por Torneio . . . . . 71
Figura 24 – Crossover de um ponto . . . . . 75
Figura 25 – Crossover de múltiplos pontos . . . . . 76
Figura 26 – Crossover uniforme . . . . . 76
Figura 27 – Evolução dos indivíduos ao longo das gerações . . . . . 98
Figura 28 – Esquemático da porta NAND-5 . . . . . 100
Figura 29 – Simulação para os valores apresentados no esquemático da figura 28. . . . . 101
Figura 30 – Processo Construtivo CMOS (NICOLETT, 2017). . . . . 124

# Lista de tabelas

Tabela 1 – Resultados mais relevantes das execuções do GA com 50% de mutação . . . . . 97
Tabela 2 – Comparativo dos melhores resultados obtidos nas execuções do AG variando-se os parâmetros de entrada . . . . . 98
Tabela 3 – Comparação entre métodos evolutivos . . . . . 103
Tabela 4 – Resultados mais relevantes das execuções do PSO. . . . . 106
Tabela 5 – Resultados mais relevantes das execuções do PSO . . . . . 107

# Sumário


**1 INTRODUÇÃO . . . . . 21**
**1.1 Objetivo . . . . . 21**
**1.2 Breve Apresentação do Trabalho . . . . . 21**
**1.3 Soluções para Síntese de Circuitos CMOS . . . . . 22**
**1.4 O estado da arte . . . . . 23**
1.4.1 O trabalho de Ivan Silva . . . . . 23
1.4.2 O trabalho de Zebulum . . . . . 24
**1.5 A escolha do problema . . . . . 25**
**1.6 Os problemas de otimização na história do Ocidente . . . . . 26**
**1.7 Motivação do Trabalho . . . . . 31**
**1.8 As escolhas efetuadas neste trabalho . . . . . 32**
1.8.1 A escolha do Problema . . . . . 32
1.8.2 A escolha do Algoritmo Genético . . . . . 32
1.8.3 A escolha do CMOS . . . . . 35
1.8.4 A escolha da porta NAND . . . . . 35
1.8.5 Síntese Hipotética e Metodológica . . . . . 36

**2 REVISÃO CONCEITUAL DE MICROELETRÔNICA . . . . . 39**
**2.1 Introdução . . . . . 39**
**2.2 A importância do inversor e o tempo de propagação . . . . . 39**
2.2.1 O Surgimento do Inversor CMOS . . . . . 39
2.2.2 Operação Básica do Inversor . . . . . 43
2.2.3 Atraso de propagação . . . . . 44
**2.3 Cálculo de $C_L$ . . . . . 46**
2.3.1 Capacitância de entrada dos próximos estágios . . . . . 46
2.3.2 Capacitância dos *gates* dos próximos estágios . . . . . 47
2.3.3 Capacitância parasita no dreno e no *bulk* . . . . . 47
2.3.4 Capacitância parasita nas trilhas . . . . . 50
2.3.5 Capacitância de Carga Total: $C_L$ . . . . . 50
2.3.6 Esforço lógico . . . . . 51
2.3.7 As portas lógicas com CMOS . . . . . 52

**3 CIÊNCIA DA COMPUTAÇÃO E A INTELIGÊNCIA ARTIFICIAL . . . . . 57**

**3.1 Apresentação das principais técnicas evolucionárias . . . 57**
3.1.1 Passos Genéricos de um Algoritmo Evolucionário . . . 57
3.1.2 PSO: O Enxame de partículas . . . 58
3.1.3 DE - Evolução Diferencial . . . 59
**3.2 Os problemas intratáveis . . . 61**
**3.3 Teoria dos Algoritmos Genéticos . . . 61**
**3.4 Teorema da Inexistência de Almoço Grátis . . . 62**
**3.5 Terminologias e Características dos Algoritmos Genéticos . . . 64**
**3.6 A estrutura de um Algoritmo Genético e os Principais Operadores . . . 66**
**3.7 Exploração das possibilidades dos operadores Genéticos . . . 67**
3.7.1 As diferentes formas de Seleção . . . 67
3.7.2 As diferentes formas de Recombinação . . . 75
3.7.3 As diferentes formas de mutação . . . 78
**3.8 Múltiplos objetivos e Restrições . . . 80**
3.8.1 O conceito de Pareto . . . 82
3.8.2 Método da Pena de Morte . . . 83
3.8.3 Penalização dos cromossomos inadequados . . . 83
3.8.4 Método da soma ponderada . . . 83
3.8.5 Pareto com $\varepsilon-$restrito . . . 84
3.8.6 A questão de dominância múltipla . . . 84

**4 DESENVOLVIMENTO DO CASO DE ESTUDO . . . 85**
**4.1 Organização do trabalho . . . 85**
4.1.1 Metodologia . . . 86
4.1.2 Equacionamento do problema . . . 86
4.1.3 Cálculo do *score* da função de avaliação . . . 88
4.1.4 Aplicação do Método do Cálculo Aritimético do Ponto do Crossover . . . 89
4.1.5 Desenvolvimento do Algorítimo . . . 90
4.1.6 Inicialização da População . . . 91
4.1.7 Avaliação dos indivíduos da população . . . 91
4.1.8 A reprodução dos indivíduos: recombinação genética . . . 94
4.1.9 A mutação . . . 95
4.1.10 Execução do algoritmo . . . 96
4.1.11 A disponibilidade do Framework e do código-fonte . . . 99
4.1.12 Validação do Circuito . . . 99

**5 COMPARATIVO ENTRE DIVERSAS IMPLEMENTAÇÕES . . . 103**
**5.1 Comparação Qualitativa . . . 103**

5.2 Comparação Quantitativa . . . . . 104

6 CONCLUSÃO DO TRABALHO . . . . . 109
6.1 Conclusões Gerais . . . . . 109
6.2 Comparação com outros trabalhos . . . . . 110
6.3 Aplicabilidade do trabalho . . . . . 111
6.4 Sugestão de trabalhos futuros . . . . . 111

APÊNDICES 113

APÊNDICE A – NOÇÕES INTRODUTÓRIAS DE MICROELETRÔNICA . . 115
A.1 Transistor MOS . . . . . 115
A.2 Análise da operação do CMOS . . . . . 117
A.3 Região de Triodo e Modelo linear . . . . . 117
A.4 Região de Saturação e Modelo Quadrático . . . . . 119
A.5 Processo de fabricação de um circuito CMOS . . . . . 120

APÊNDICE B – MODELOS NÃO-EVOLUCIONÁRIOS . . . . . 125
B.1 O Teorema de Pierre Fermat . . . . . 125

APÊNDICE C – ELEMENTOS HISTÓRICOS DA COMPUTAÇÃO . . . . . 127
C.1 A Máquina Universal de Turing . . . . . 127
C.2 O Teste de Turing . . . . . 128
C.3 Otimização Global . . . . . 128

APÊNDICE D – DISCUSSÃO FILOSÓFICA A RESPEITO DA FUNDAMENTAÇÃO DOS ALGORITMOS GENÉTICOS E A ORIGEM DAS ESPÉCIES . . . . . 129
D.1 Introdução à discussão . . . . . 129
D.2 Apresentação do Design Inteligente . . . . . 132
D.3 Questões importantes e questões não-importantes . . . . . 132
D.4 Em relação às observações feitas nos modelos algorítimicos . . . . . 134
D.5 A refutação de Orr ao Intelligent Desing . . . . . 138
D.6 Brevíssimas objeções metafísicas a respeito da evolução darwinista . . . . . 139
D.7 Algumas Conclusões . . . . . 141

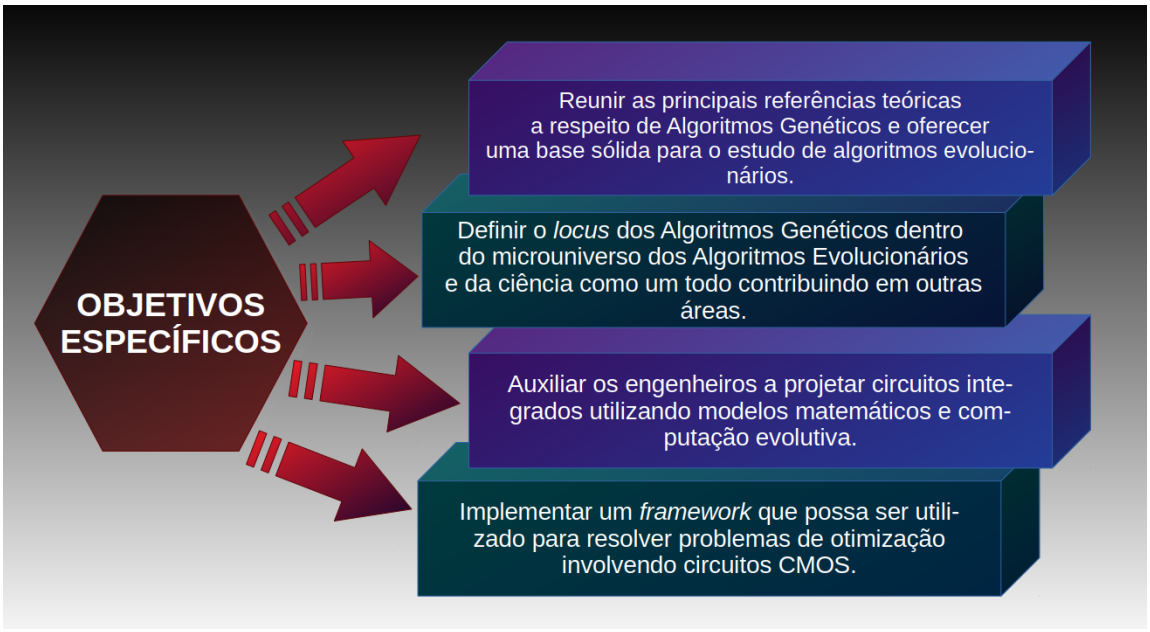
REFERÊNCIAS . . . . . 143

# 1 Introdução

Neste capítulo trata-se da revisão bibliográfica. Aqui expõe-se também os objetivos deste trabalho e a justificativa para a escolha do tema.

## 1.1 Objetivo

O objetivo geral deste trabalho é fazer um levantamento do estado da arte a respeito dos aspectos teóricos sobre algoritmos genéticos e sua aplicação para solução de problemas de otimização do processo de síntese de circuitos CMOS averiguando se é possível encontrar um modelo matemático que auxilie os engenheiros projetistas no processo de encontrar os parâmetros dimensionais de projeto de um transistor CMOS de forma a obter condições apropriadas para cada tipo de projeto. Os objetivos específicos são melhor condensados na figura 1.1.




## 1.2 Breve Apresentação do Trabalho

A fim de atingir tais objetivos fez-se ampla pesquisa bibliográfica apresentando breves resenhas de alguns trabalhos cujas técnicas apresentadas são representações distintas das diversas

pesquisas, elaborou uma apresentação que crê-se completa, embora suscinta, a respeito do tema de Algoritmos Genéticos, e implementou-se um caso de estudo específico no qual se modela matemáticamente a implementação de uma porta NAND-5 a partir das chamadas técnicas evolucionárias, em cima do qual foram feitos alguns experimentos a fim de se apontar conclusões que sejam uma contribuição a mais ao estado da arte, além da própria pesquisa. A validação da implementação se faz utilizando Simulação no SPICE.

Por fim, nos Apêndices apresenta-se um guia completo de Microeletrônica, onde o leitor pode encontrar toda fundamentação teórica na área utilizada para modelar o caso de estudo deste trabalho e compreender mais profundamente suas justificativas; um estudo comparativo da implementação principal feita neste trabalho (utilizando Algoritmos Genéticos) com implementações de outros algoritmos evolutivos ou pseudo-evolutivos; e, por fim, um estudo filosófico [1] a respeito das Teorias que envolvem o estudo de Algoritmos Genéticos e como este trabalho pode servir de evidência para objetar contra a Teoria Darwiniana - utilizando um modelo computacional da própria teoria.

## 1.3 Soluções para Síntese de Circuitos CMOS

A síntese de circuitos integrados analógicos é uma tarefa complexa devido ao grande número de parâmetros e de objetivos envolvidos no processo. Como parâmetros, tem-se largura (W) e comprimento (L) do canal dos transistores, a corrente de polarização do circuito, a tensão de entrada em modo comum, as capacitâncias e resistências envolvidas, o comprimento das linhas, a capacitância das linhas, etc. Como objetivos, vê-se o ganho de tensão em malha aberta, a frequência de ganho de tensão unitário, a margem de fase, as condições de polarização por tensão e corrente contínuas, a potência total dissipada pelo circuito, a área de todos os transistores, os regimes de operação e inverso dos transistores, o *Slew Rate* (SR), a razão de rejeição em modo comum (CMRR), entre outros. Para tanto, apresentam-se diversas soluções para esta síntese, sendo a mais recomendada, quando possível, a analítica. Todavia, nem sempre as soluções analíticas são possíveis sendo necessárias heurísticas [2] e simulações para encontrar os valores e fazer ajustes. O problema desta solução é que, embora seja a mais segura, é também a mais onerosa, tanto em tempo, qu. Modelos computacionais, então, são utilizados codificando a topologia do circuito, em conjunto com os parâmetros do transistor, em estruturas binárias. Outras soluções fazem buscas simples entre possíveis valores de parâmetros dos transistores a fim de encontrar um subconjunto que atenda ao problema específico. Todavia, se o espaço amostral não contiver as soluções para o problema em questão, a busca resultará vazia ou retornará valores que de fato não são resposta. Outras soluções são basea-

[1] O qual pode muito bem ser traduzido por científico, dado ser um erro grosseiro a equivalência entre ciência e o método cartesiano, quando **ciência** por excelência é a própria Filosofia. (NOUGUÉ, 2017))

[2] Cuja definição pode ser encontrada na Seção 3.5.

das na combinação das dimensões dos transistores e condições de regime de inversão que podem ser identificadas para alcançar um objetivo específico do projeto. Circuitos, porém, ainda que de baixa complexidade, já podem necessitar de uma quantidade tão grande de bits para serem representados que o espaço de soluções poderia exceder o número estimado de átomos existentes no universo. A solução proposta utiliza uma combinação destas técnicas a fim de analiticamente modelar o circuito para filtrar e limitar, dentre uma aleatoriedade de possíveis parâmetros, aqueles que podem resolver problema de otimização, sem entretanto, pretender encontrar todos os parâmetros envolvidos [3] e efetuar a busca da solução para o circuito dentro de um espaço amostral, o que também ajuda a reduzir drasticamente a quantidade de bits necessários. Essa abordagem está de acordo com os princípios Lean (INSTITUTE, 2018) e Ágil (FOWLER et al., 2001) para os projetos de software e busca seguir a filosofia de manter todo projeto tão simples quanto se possa; princípios estes tão difundidos na indústria de software, também para o projeto de hardware.

## 1.4 O estado da arte

### 1.4.1 O trabalho de Ivan Silva

Na última década um não grande número de trabalhos têm se destacado neste campo de estudos. No Brasil, trabalhos como o de Ivan (FILGUEIRAS, 2010) surgem emergentes nos mecanismos de busca de artigos acadêmicos e teve por objetivo dimensionar um circuito CMOS utilizando algorítimos genéticos. Ele utiliza para isso a linguagem **SCALA**, que se vale dos paradigmas orientado a objeto e funcional, o que é um fator interessante desta linguagem que pretende ligar o melhor dos dois paradigmas. O estudo de caso deste trabalho foi a construção de uma fonte de corrente. Como critérios para o Algoritmo Genético, Ivan utilizou os seguintes parâmetros derivados do circuito em questão:

| **Propriedade** | **Valor Mínimo** | **Valor Máximo** |
|---|---|---|
| **L** | $0.35\mu m$ | $5\mu m$ |
| **W** | $1\mu m$ | $100\mu m$ |
| **Rs** | $5K\Omega$ | $50K\Omega$ |
| ***threshold*** | – | 33% |
| **tolerância de variação** $\zeta$ | – | 5% |
| **taxa de mutação** $\alpha$ | – | 50% |

Para testar o algorítimo, especificou-se uma fonte com $I_{out} = 0,5\mu A$ com tolerância de variação $\pm 0,03\mu A$ e que a fonte deveria operar com boa estabilidade de tensão e com tensão de alimentação menor que $2,1V$. O resultado do trabalho foi um algorítimo de baixa complexidade numa linguagem "*scriptada*". Como ponto positivo do trabalho, encontra-se o fato dele ter conse-

[3] Dado que alguns deles podem ser encontrados analiticamente.

guido mostrar ser possível gerar-se a síntese de circuitos CMOS a partir de algorítimos evolutivos, obtendo resultados práticos além de ter feito a integração com simuladores *open source*. Com essa abordagem, vê-se claramente ser possível reduzir significativamente o custo de projetos desta categoria de circuitos ao passo que alguns elementos apresentam-se como complicadores. O primeiro deles é a linguagem utilizada para a programação. O **SCALA** é uma linguagem de sintaxe confusa, pouco aplicada no mercado (no site de empregos Indeed, menos de $0,02\%$ dos empregos relacionados à tecnologia da informação colocam Scala como requisito) e, embora *scriptada*, é extremamente verbosa [4]. O trabalho abre o leque para várias melhorias, dentre elas:

- Estudar se as otimizações funcionam com um número maior de restrições, como gerar fontes com as menores dimensões possíveis de transistores;
- Forçar os transistores em fraca ou forte inversão;
- Integrar simuladores comerciais mais confiáveis;
- Aplicar técnicas mais sofisticadas de algorítimos genéticos;
- Implementar uma interface gráfica amigável para que projetistas experientes possam utilizar esta ferramenta sem a necessidade de conhecimentos aprofundados no ambiente de programação utilizado;
- Testar outros circuitos;
- Fazer outros tipos de simulação, como transiente e AC, e verificar se resultados semelhantes serão obtidos.

### 1.4.2 O trabalho de Zebulum

Sabe-se que os transistores MOS podem operar em inversão alta, baixa ou moderada. Neste trabalho de ZEBULUM; PACHECO; VELLASCO, talvez um dos mais antigos e significantes artigos, os autores introduziram o uso dos algoritimos genéticos, numa abordagem multi-objetiva, para solucionar o problema de dimensionamento de circuitos MOS, pretendendo que sua metodologia pudesse ser aplicada tanto no desenvolvimento de circuitos analógicos e digitais. No caso particular do desenvolvimento de um algorítimo para otimizar o processo de construção de amplificadores operacionais, verifica-se que um vasto número de especificações técnicas precisam ser consideradas. O cenário ideal incluiria todas as especificações, todavia isso faria com que o processamento tornasse quaisquer técnicas inviáveis. Dentre o que fora desenvolvido neste trabalho o que mais será aproveitável para o nosso projeto são:

[4] Isto é, com um grande número de palavras reservadas e formas específicas de programar.

- As condições que justificam o uso de algorítimos genéticos para resolver o nosso problema.
- Os critérios que utilizamos para escolher uma abordagem específica em detrimento às demais.
- A prevenção de alguns problemas já tratados.
- Critérios de comparação de resultados.

Além do já conhecido problema de não se poder utilizar todas as especificações técnicas para representar os componentes eletrônicos e encontrar a função de transferência correta, outras questões são importantes serem levantadas. Uma dessas questões, levantadas pelos autores, foi: "Por que precisamos de um algorítimo genético?" [5]. O projeto feito por seres humanos baseia-se na solução de um sistema de equações. Entretanto, não há uma solução exata para o sistema quando múltiplos objetivos são tomados. Neste caso o problema é melhor solucionado numa busca dentro de um espaço de soluções. Devido ao tamanho do espaço de busca e à natureza multi-objetiva do problema, os autores decidiram usar algorítimos genéticos para resolverem o problema.

## 1.5 A escolha do problema

O problema deste trabalho é encontrar uma referência bibliográfica completa que reflita o Estado da Arte em relação às teorias de Algoritmos Evolucionários (em especial os Algoritmos Genéticos), colocadas na ordem das Ciências [6]: desde as aplicações práticas, passando pela fundamentação matemática e explorando as fronteiras filosóficas sobre os temas que envolvem este assunto. Este problema necessita, pois, ser abordado em dois vieses: o teórico como o apresentado na figura 1 e o prático (figura 2).

Em relação a questão prática, após desenvolver uma série de circuitos digitais e estudar os processos construtivos de circuitos CMOS, percebeu-se que por mais que as ferramentas de simulação ajudem e muito neste processo ainda há um fator de *felling* do engenheiro projetista em boa parte baseado no processo de tentativa e erro. Geralmente os projetistas, através de modelos simplificados de blocos de transistores, calculam o ponto de partida das dimensões dos transistores. Posteriormente, ele aplica esses valores nos simuladores e inicia-se o processo de tentativa e erro, baseado na experiência do próprio projetista e nos resultados que este vai encontrando nas sucessivas simulações. Quanto mais experiente o engenheiro projetista, melhor tende a ser o circuito e

[5] Ou seja, a questão *propter quid sit*.

[6] Pressupõe-se aqui duas coisas a saber: que a Ciência por excelência é a Filosofia - em especial a Tomista - e não o Cartesianismo, sendo a argumentação técnica parte do processo científico e não todo ele; mais que isso, pressupõe-se a existêcia de uma clara hierarquia nas Ciências, tendo a Lógica como uma arte-ciência propedêutica (NOUGUÉ, 2017), passando-se pelas ciências naturais ou Física (em cujo entremeio encontra-se a Matemática como uma das linguagens), aurindo à Economia, à Ética, à Metafísica e, por fim, a Teologia Sagrada. É neste universo hierárquico que se pretende ordenar este trabalho, colocando a arte tecnológica em seu *locus*.

mais rápido o processo. Se o projetista não é tão experiente, para atingir resultados semelhantes, o processo tende a ser mais oneroso. Neste sentido, por mais que seja previamente sabido ser impossível substituir por completo a riqueza da experiência humana em toda sua complexidade [7], encontrar um processo automático, fundamentado num modelo matemático, que auxilie os projetistas - especialmente os menos experientes - a reduzir o tempo de projeto é, não apenas desejável, como de grande contribuição. Tenta-se, portanto, investigando a bibliografia do estado da arte desta tecnologia, e das ciências que a circunscrevem, verificar se tal modelo é possível e se ele ajuda a elucidar as questões abertas nestas ciências.

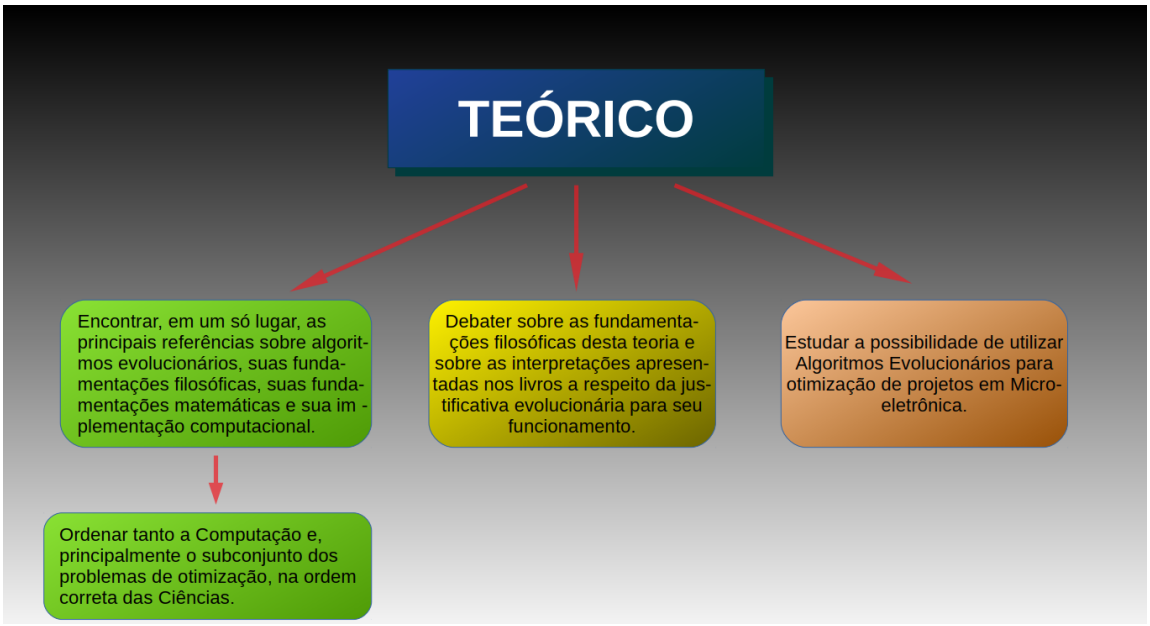



Figura 1 – Problema Teórico

## 1.6 Os problemas de otimização na história do Ocidente

Os problemas de otimização são antigos e históricos: encontrar a menor distância a ser percorrida por um cacheiro viajante, a menor quantidade de sementes a ser utilizada para se obter uma quantidade desejada na colheita, a maior quantidade de habitações que se pode colocar numa determinada área, etc. Na história da matemática, o primeiro relato que se tem de um problema de otimização foi documentado por Euclides, em 300 d.C. Ele considerou a mínima distância entre um ponto e uma linha reta e provou em seu livro "Os Elementos" (SC.D., 1908). Em 200 a.C.,

[7] Cuja mera pretensão já previamente tornaria falacioso este trabalho.


Figura 2 – Problema Prático

Zenodorus estudou o famoso Problema de Dido (WEISSTEIN, 2017), o qual fora descoberto por Virgílio em 19 a.C., em sua obra Enéada (VIRGILIUS, 19 a.C.). Há 100 a.C. Heron provou, em "Catoprica", que a luz viaja entre dois pontos através do menor caminho quando refletida por um espelho. Neste período os problemas de otimização eram sempre relacionados aos problemas geométricos encontrados pelos gregos. Antes da descoberta do Cálculo, apenas alguns problemas de otimização eram profundamente investigados e razoavelmente explicados matematicamente. Em 1615, J. Kepler, pensou no problema de otimizar as dimensões de um barril de vinho. Ele também formulou a versão mais primitiva do chamado Problema da Secretária (WIKIPEDIA, 2017) quando ele começou a olhar para o vinho. Nas décadas de 1660 e 1670, Gottfried von Leibniz e Sir. Isaac Newton criaram a análise matemática que formou a base do Cálculo das Variações. Para tanto, utilizaram de problemas de otimização. Fato curioso não pode deixar de ser que o Cálculo das Variações surge, e com uma grande deficiência, após o conhecimento de vários conceitos envolvendo as tais "derivadas". Um dos mais importantes trabalhos neste âmbito é o famoso "Problema da Mínima Resistência de um Corpo" de Newton (PELETIER, 2010).

Em 1696, Johann e Jacob Bernoulli estudou o "Problema de Brachistochronus" e nasce propriamente o cálculo diferencial. Em 1740, a publicação de Euler iniciou a pesquisa na teoria geral do cálculo das variações. Sucederam-nos os grandes Lagrange, dentre muitas coisas formulando o

"Problema do Platau" (problema das superfícies mínimas), o qual fora solucionado posteriormente por J. Douglas. Em 1784, G. Monge investigou um problema de otimização combinacional conhecido como Problema do Transporte Ótimo.

Todas estas abordagens eram até então analíticas. O primeiro algorítimo de otimização foi apresentado no século IXX por Weierstrass, Steiner, Hamilton e Jacobi. Em 1808, Legendre apresenta o "Método dos Mínimos Quadrados", cuja invenção também fora clamada por Gauss. Este é ainda o método mais amplamente utilizado para minimizar a soma dos resíduos de uma regressão, de forma a maximizar o grau de ajuste do modelo aos dados observados.

Em 1847 apresentou o Método Gradiente.

O século XX trouxe inúmeras descobertas em realação a estes problemas de otimização.

A Teoria dos Mínimos e Máximos surge em 1917 com Harris Hancock (HANCOCK, 1917).

No ano de 1944, John Von Neuman e O. Morgenstern resolveram os problemas das decisões sequenciais utilizando pela primeira vez a idéia de programação dinâmica. Um trabalho relacionado surgiu em seguida (1947) pela pesquisa de A. Wald. No mesmo ano, as Forças Armadas dos Estados Unidos usaram, através do trabalho de G. Dantzig, as terorias de Von Neuman para resolver os problemas LP. Fora descoberto aí o Método do Simplex.

Em 1936 surgira a Máquina Universal de Turing, criada pelo brilhante matemático Alan Mathison Turing após concluir seu Mestrado no "Colégio do Rei" (*King's College*)em 1935.

Até o final da década de 1970 os computadores *mainframes* ainda eram reinantes e, sendo eles muito caros e complexos, reinava ainda nas pesquisas na área de otimização, os modelos analíticos, por mais que agora publicasse mais modelos analíticos programáticos.

No âmbito da filosofia, o século XX fora o auge da inversão de pensamento que culminaria no relativismo tecnicista, no matematicismo e que implicaria, por um lado, na decadência da razão humana (tendo como expoente as doutrinas materialistas, e auge o relavismo) e, por outro, em grandes descobertas pragmáticas. Dentre muito fatores, isso levou a dar-se mais importância a um processo de exploração dos fenômenos em detrimento da busca da verdade e conformação à realidade pelo princípio das causas primeiras. Assim, teorias em muitos aspectos falhas como a Relatividade Geral de Albert Einstein e a Teoria da Evolução das Espécies de Charles Darwin deixaram de ser tão questionadas em relação a serem verdadeiras e bons modelos da realidade e passaram a ser exploradas. Uma vez que toda teoria que respeite os princípios lógicos, ainda que falaciosa, muito provavelmente contenha em si algum elemento de verdade (ou muitos), este processo exploratório permitiu descobertas que, sem ele, levariam mais tempo para serem encontradas.

É neste mundo de novidades filosóficas no qual também a chamada "comunidade acadêmica" encontra-se inserida e, nas décadas de 1940 e 1950, muitos acadêmicos cartesianos (matemáticos,

psicologistas, engenheiros, economistas, sociólogos e políticos) começaram a discutir a possibilidade de se criar um "cérebro artificial". Dessas discussões surge, em 1956, o campo da chamada "Inteligência Artificial".

Era de se esperar dadas as discussões que ocorriam na época que o início da área girava em torno de se construir o tal "cérebro artificial".

A Cibernética de Nobert Wiener (1948) descreveu o controle e a estabilidade de redes eletétricas. As pesquisas na Neurologia dos anos de 1950 mostraram que o cérebro humano funciona como uma rede de pulsos [8] elétricos. A junção destas descobertas com a Teoria da Informação de Shannon, que descreveu os sistemas digitais com pulsos binários, e a Teoria da Computação de Turing que mostrou que toda forma de computador pode ser resolvida num computador de pulsos binários, levou os cientistas a concluirem que seria possível a criação de uma rede neural artificial.

Em 1951, Mavin Minsky lança o primeiro computador neural, denominado Snark. Embora não tivesse feito nenhum processamento de dados, mostrou ser possível uma máquina operar de forma neural.

Surgem então duas formas inteligência artificial: a simbólica e a conexiosta. Na simbólica, o comportamento inteligente humano é simulado, desconsiderando os mecanismos responsáveis por ela. Na conexionista, busca-se criar um modelo da estrutura do cérebro a fim de que este sistema seja capaz de aprender, assimilar, errar e aprender com os próprios erros.

Contrariando as teorias filosóficas de sua época (e também da nossa), o livro de Ashby - "Design for a Brain: The Origin of Adaptative Behavior" (1952) - observa que o conhecimento real de um indivíduo humano (o único ser vivo capaz de raciocinar) não lhe é intrínseco, mas sim aprendido de outro indivíduo, diferente do restante dos animais que são instintivos. Ashby considera o indivíduo animal uma máquina e o avalia em termos de estabilidade.

Seu estudo foi muito interessante e fundamental para o progresso das redes neurais, pois a melhoria do Snark necessitaria que ele tivesse um aprendizado, ou seja, que lhe fosse ensinado.

Outro fator interessante é que isso faz com que o modelo artificial criado pela Técnica, condiz de certa forma com a realidade conhecida da Biologia e da Antropologia: os indivíduos possuem duas formas de conhecimento - o institivo, que lhe é parte intrínseco e parte aprendido do meio; e o específico, aprendido de outros indivíduos. Também os "indivíduos" do modelo artificial apresentaram esta característica, dado que tornou-se necessário que o sistema computacional neural fosse "ensinado".

[8] O conceito de pulso usado aqui é o utilizado na eletrônica digital. Em sistemas digitais, pulsos contrapõem breves picos de tensão DC (corrente contínua) com cada explosão tendo um início (ou aumento) abrupto e um final (ou decadência) abrupto (a).

Surgem, na década de 1958, a rede Perceptron e, em 1960 a ADALINE a MADALINE (*multiple ADALINE*). Estas redes fizeram uso desta descoberta de forma que o *bias factor* recebe informação do meio. Além disso, estas redes devem ser "ensinadas". Logo depois, Hebb publica um trabalho que dá origem aos sistemas adaptativos.

Uma vez consolidadas as redes neurais, procurou-se utilizá-las para auxiliar na solução de problemas complexos. Dentre esta categoria, provavelmente o mais difícil problema seja o de otimização. Embora as redes neurais sejam eficazes na solução de problemas deste tipo, elas apresentam uma grande lacuna principalmente para problemas de solução NP [9]. Nesta classe de problemas, a rede oferece dificuldades de convergência ou reverbera [10] (ZENG; CHENG, 2009)

Para se resolver esse tipo de problema, é, também, nos meados da década de 1950, que Lawrence J. Fogel e John Henry Holland começaram a pensar se seria possível, ao invés de se criar um modelo do cérebro humano, utilizar o modelo de comportamento de uma população para "ensinar" as redes neurais. Daí surgem os Algorítimos Genéticos.

Os Algorítimos Genéticos iniciaram uma área da computação denominada Computação Evolucionária [11] a qual se desenvolveu numa vastidão de diferentes técnicas nos anos seguintes.

A Computação Evolucionária é uma família de algorítimos de otimização global C.3.

Na Computação Evolucionária, um conjunto inicial de "soluções candidatas" (chamada indivíduos) é gerado e depois é atualizado em sucessivas iterações. Cada iteração é denominada geração e, a cada geração, os indivíduos passam por uma série de operações que simulam os processos de seleção natural. Novas gerações são produzidas de forma estocástica, ou seja, de uma forma cujo estado é indeterminado, com origemm em eventos aleatórios. Por exemplo, o lançamento de um dado.

Obviamente, os algorítimos evolucionários não foram a primeira forma computacional que surgiu para se resolver problemas de otimização e nem sempre são a melhor alternativa.

Os métodos numéricos são, por excelência, aqueles que ocupam o trono da metodologia ótima para se resolver problemas de otimização. Isso se dá porque visam encontrar soluções ótimas, exatas. Os algorítimos evolucionários, por outro lado, são modelos heurísticos os quais, por sua vez, não garantem a solução ótima sempre e, se o fizer nalgum caso, dificilmente tornarão a repetir a façanha. Isso implica que os algorítimos genéticos não oferecem garantia alguma sobre a qualidade da solução encontra apenas que esta solução é uma solução aceitável e num grau aceitável de

[9] Na teoria da complexidade computacional, NP é acrônimo para *non-deterministic polynomial time problems*, que denota um conjunto de problemas que são decidíveis em tempo polinomial por uma máquina de Turing não determinística. A importância desta classe de problemas de decisão é que ela contém muitos problemas de busca e de otimização, como o problema do caixeiro viajante.

[10] Reverberar é ficar muito facilmente presa num mínimo ou máximo local

[11] Ou "Programação Evolucionária"

otimização para o problema.

## 1.7 Motivação do Trabalho

O universo da Inteligência Artificial estabelece uma inegável proximidade num abismo criado desde o Iluminismo [12] e, pricipalmente após as inversões do Discurso do Método [13](cartesianismo) entre a chamada Ciência Moderna e a Filosofia e por este motivo, muito mais que pelo técnico, particularmente [14], rendem-lhe um fascínio ímpar.

Muito mais que do âmbito técnico, de icomensurável fascínio, dado o universo de possibilidades que daí tem surgido, a complexidade matemática envolta, os diferentes paradigmas, o comportamento dos computadores e o futuro que disso decorre, motivo de muitos trabalhos científicos como de uma variedade de trabalhos em outras artes [15], as questões que fazem fronteira aos temas deste trabalho nunca foram tão atuais.

Do lado da Inteligência Artificial, a questão da teoria da Evolução das Espécies frente a outras teorias, a própria questão sobre a "inteligência", e a tentativa de refutar tais teorias sem negar os fatos observados por suas aplicações e que até hoje as sustentam frente a comunidade acadêmica são questões a serem discutidas.

Do lado da eletrônica, a possibilidade de criar circuitos em dimensões cada vez menores, otimizadas por um processamento baseado em uma "inteligência" [16] artificial e a questão de se

[12] " "As Três Revoluções na Arte" (MONTFORT, 2018a), "Iluminismo, Trevas na época das Luzes" (MONTFORT, 2018b)

[13] O *"Cogito ergo sum" pressupõe que o pensamento do homem antecede seu predecessor, a idéia antecederia ao ente que a idealizou, o que é absurdo. A dúvida metódica pressupõe que deva-se duvidar de tudo, ao passo que exclui deste "tudo" a própria dúvida. Em outras palavras a dúvida para Descartes é certeza no momento em que é pensada, e não a realidade. Real, para Descartes, é a dúvida e não o sujeito que duvidou, ainda que este mesmo sujeito tenha dúvidas concomitantes e diferentes e seja, pela obviedade, levado a concluir que o real é ele próprio e não a dúvida, ao contrário do que afirma Descartes. Em decorrência desta inversão surge posteriormente o experimentalismo racionalista, materialista e panteísta de Voltaire, Diderot e D'Alambert, que busca adequar a realidade às teorias ao invés do contrário - como era feito na Idade Média. O Método Catesiano (denominado equivocadamente científico) assume verdadeiro o absurdo lógico "Tudo é duvidável, exceto a própria dúvida!" - que alimentará o espírito irracionalista, misticista e gnóstico do pietismo protestante alemão, do quietismo francês de Rousseau e que posteriormente levará ao relativismo contemporâneo, ainda mais absurdo, que afirma que "É absolutamente verdadeiro que não existe verdade absoluta". (POPPER, 1874) (MACHADO, 2018)* No cartesianismo, pela primeira vez, a Filosofia (Ciência) deixa de partir do "Ser" e passa a partir do "Eu". Com efeito, mesmo um modernista que não pode ser elogiado (Karl Popper), mostrou o caráter dialético do racionalismo ao mostrar que *"O racionalismo é uma fé irracional na razão"*.

[14] Por mais que seja uma concepção subjetiva, geralmente imprópria aos trabalhos acadêmicos, crê o autor que as motivações sinceras despertarão maior interesse e serão de maior contribuição para a comunidade acadêmica e toma-se por base para tal extrapolação cientistas como Sir. Isaac Newton, Von Newman e o próprio Turing que de forma semelhante assim se expressaram em suas obras.

[15] Utiliza-se neste trabalho o termo "arte" no sentido empregado pelas Artes Liberais

[16] O termo inteligência virá entre aspas quando referir-se às máquinas dado que uma máquina não pode conhecer "aquilo que é", logo não pode ter intelecto e, por fim não pode ser inteligente (*"Obiectum autem intellectus est quod quid est, idest essentia rei, ut dicitur in III de anima. Unde intantum procedit perfectio intellectus, inquantum cognocit essentiam alicuius rei".) (Thomas Aquinas, )*

tornar possível que as máquinas auxiliem o projetista não apenas na simulação dos circuitos como também no próprio cálculo e projeto do mesmo, de forma dinâmica, evolutiva.

Este universo fronteiriço que aproxima a chamada Ciência Moderna da Ciência Antiga, e que tenta aproximar a máquina do homem, e expõe como nenhuma outra área, a Matemática como uma **linguagem** [17] levantando questões nos mais variados campos do conhecimento humano é o que motivou o autor a enveredar-se por este caminho.

## 1.8 As escolhas efetuadas neste trabalho

Nesta seção são abordadas as principais escolhas efetuadas neste trabalho.

### 1.8.1 A escolha do Problema

As portas lógicas estão presentes em todos os circuitos microprocessados. São elas as células constituintes de todo sistema digital e não poucas vezes são construídas e dimensionadas de forma ineficiente ou demasiada onerosa. Certamente que dimensionar uma única porta de lógica NAND ou NOR [18], de duas entradas é uma tarefa trivial. Todavia, e quando se pretende implementar a operação $\neg(ABC)$ sendo $A, B$ e $C$ entradas? Neste caso, a solução mais óbvia é utilizar-se duas portas NAND de duas entradas. Entretanto, ao se observar tal implementação a nível de construção dos transistores, percebe-se que tal solução desperdiçaria uma valiosa área de silício.

Neste sentido, calcular as dimensões exatas dos transistores de forma a otimizar a área de silício e, simultaneamente, responder em M entradas como responderia um circuito com apenas duas entradas, torna-se uma tarefa bem mais complexa.

As atuais maneiras de se implementar são onerosas envolvendo aproximações por modelo de equações, simulação e experiência do projetista.

Otimizar este processo obtendo dimensões corretas que satisfaçam as exigências do projeto poupará precioso tempo de projeto gasto em tentativa e erro.

### 1.8.2 A escolha do Algoritmo Genético

A primeira pergunta a se fazer num trabalho de otimização é "Por que não utilizar um modelo analítico derivado do Teorema de Fermat?", como por exemplo o método gradiente *hill climbing*. Soluções analíticas são sempre preferíveis a soluções probabilísticas dado que encontram

[17] Há uma série de pesquisadores e estudiosos, dentre os quais o autor deste trabalho, que não consideram a Matemática como Ciência, nem tampouco como Arte. A Matemática enquadra-se no limiar entre Arte e Ciência sendo mais perfeitamente definida como uma linguagem. (KENNEY et al., 2005)

[18] As lógicas NÃO-E (*NAND*) e NÃO-OU (*NOR*) são implementações eletrônicas das operações equivalentes na Álgebra Booleana.

o valor exato e não valores aproximados. Todavia há uma gama de problemas que não podem ser resolvidos de forma analítica ou não é viável que se resolva de forma analítica devido ao alto custo computacional. Nosso problema depende de encontrar valores de $W_n$, $W_p$, $L_n$ e $L_p$ que promovam simultaneamente:

- Tempo de subida igual ao tempo de descida;
- A menor corrente possível no dreno, implicando na menor potência;
- A melhor condição de performance transiente.
- A menor área de silício considerando a área consumida pelas trilhas.

Como se viu até então, descobriu-se ao longo dos séculos várias formas de se resolver problemas de otimização, mas foi apenas após a segunda metade do século XX, com o advento dos computadores transistorados, que uma gama de problemas particularmente complexos puderam encontrar um horizonte de solução. Os chamados problemas NP completos não podem ser solucionados de maneira exata e uma categoria particular de algorítmos é inventada como solução: os algorítimos evolutivos, ou evolucionários. Estes algorítimos modelam uma gama de mecanismos da Genética e da Teoria da Evolução das Espécies.

Aceite-se ou não a Teoria Darwinista, um fato que a precede e para o qual ela surge, é que a natureza vê-se munida de diversos recursos para resolver uma série de problemas de sobrevivência.

Valendo-se de uma abstração destes recursos, os cientistas do século XX criaram uma série de técnicas metaheurísticas e enxergaram o espaço de busca das soluções como uma população. Cada possível solução é um indivíduo na população. Estes indivíduos passam por processos de reprodução, mutação, recombinação e seleção. A "inteligência" do algorítimo reside justamente na função de adptação (*fitness function*) que julga cada indivíduo da população e dá a adaptabilidade deste indivíduo. A cada geração, o conjunto de soluções da mesma tende a ser mais adaptado à função *fitness* que a anterior até se atingir um conjunto de soluções considerado aceitável para o problema. É interessante perceber três elementos: a população originária pressupõe-se ser gerada aleatoriamente; todavia, os valores ditos aleatórios já pressupõem uma "inteligência" que lhe seja superior [19] e que define os limites entre os quais estes valores "aleatórios" podem ser gerados. [20] O segundo fator interessante é justamente o próprio julgamento de adaptabilidade, o qual é dado não de maneira aleatória, mas pela inteligência do projetista. O terceiro, e talvez mais importante, é que os algorítimos evolutivos não encontram a solução ótima, mas a que se adapta de forma melhor

[19] Ou seja, que não é aprendida pelo algorítimo, mas que já está intrínseca nos próprios dados.
[20] Logo, os valores não são assim tão aleatórios, por mais que neste modelo, se use uma heurística para gerá-los. Daí o processo ser chamado de **Meta**heurítica.

no tempo mais hábil, com os recursos disponíveis. Nesta gama de algorítimos o primeiro que surgiu historicamente foram os **Algorítimos Genéticos**. Além do mais antigo é também o mais popular e genérico deles. Ele é recomendado para todo processo que ofereça restrições no espaço de busca sobre o qual uma função tende a assumir o melhor valor[21] após o processo de otimização. Outra categoria de algorítimos, chamados *algorítimos de enxame*, são descritos a seguir:

- **Colônia de formigas:** baseado no comportamento das colônias de formigas, este algorítimo é melhor utilizado em otimizações combinatórias e problemas gráficos, como reconhecimento de imagens.
- **Enxame de abelhas:** é baseado no comportamento de um enxame de abelhas e é recomendado para otimizações de problemas numéricos, combinatoriais e multi-objetivos.
- **Colônia de abelhas:** diferente do exame de abelhas, a colônia de abelhas, é um algorítimo baseado no comportamento de uma colméia e é recomendado para problemas de caminho mínimo como roteamento e agendamentos.
- **Enxame de partículas:** é uma generalização do processo de comportamento de um conjunto de animais reunidos. Também recomendado para problemas de otimização puramente numéricos.
- **Cuckoo search:** é baseado no parasitismo da espécie *cuckoo* [22].
- **Algorítimo do vagalume:** baseado no comportamento dos vagalumes, no qual os indivíduos atacam uns aos outros com *flashes* de luz. Ele é muito usado para otimizações multitudinais.

Estes são os principais algorítimos evolutivos e concorrem com o algorítimo genético para a solução de nosso problema. Entretanto, embora seja numérico e multi-objetivos, não se trata de um problema combinatorial, nem de caminho mínimo.

Por este motivo, escolheu-se os Algorítimos Genéticos, que são o caso mais geral, em detrimento dos algorítimos de enxames.

No Capítulo 5 elegeu-se o enxame de partículas como modelo para esta categoria a fim de comparação. Nele, foi feita uma comparação qualitativa e outra quantitativa entre o Algoritmo Genético (GA) e o Enxame de Partículas (PSO).

[21] Entenda-se por "“melhor”" aqui um valor de máximo local ou global.
[22] Os cuckoos são pássaros da família *Cuculidæ*.

### 1.8.3 A escolha do CMOS

De todas as escolhas, a escolha do CMOS foi a mais fácil. O Grupo de Microeletrônica da Universidade Federal de Itajubá tem mais de uma década de destaque com projetos voltados ao desenvolvimento de circuitos integrados em tecnologia CMOS, para operar com frequências RF, com especial ênfase em dispositivos de aplicações biomédicas. Além da vasta experiência e especial interesse na própria Universidade da qual se pretende obter o título de Mestre em Ciências e Tecnologias da Informação com este trabalho, os circuitos integrados constituídos por transistores de efeito de campo (*Field-Effect Transistors* [FETs]) "são o apogeu de uma história de desenvolvimento tecnológico que teve início em 1925. Naquele ano, J. Lilienfeld, da Universidade de Leipzig, propõe um dispositivo com condutividade modulada por campos elétricos. Em 1935, o alemão O. Heil solicita na Inglaterra a patente de uma estrutura que muito se assemelha aos transistores MOS modernos".(KOFUJI; ZUFFO; SOARES, 2004) Em 1958, Jack Kilby construiu o primeiro circuito integrado de um flip-flop com dois transistores da Texas Instruments. Em 2008, o microprocessador Itanium da Intel já continha mais de 2 bilhões de transistores; e 16Gb de memória Flash contém 4 bilhões de transistores. Isso corresponde a um crescimento anual de 53% ao longo dos últimos 50 anos, o que confere aos transistores CMOS a característica de ser a tecnologia que sustenta a mais alta taxa de crescimento da história. Este crescimento deve-se a um processo estável de miniaturização dos transistores e a uma melhora no processo de fabricação. Muitas outras áreas da engenharia, em especial da eletrônica e da própria computação involvem um balanço entre desempenho, potência e preço. Entretanto, os transistores, em especial os CMOS, também se tornaram extremamente performáticos [23], de baixíssimo consumo [24] e baratos [25]. (WESTE; HARRIS, 2011) Por estes motivos, acredita-se neste trabalho serem os transistores CMOS a tecnologia ideal para compor o caso de estudo específico.

### 1.8.4 A escolha da porta NAND

Dentre as diversas possibilidades de ciruitos, tanto digitais quanto analógicos a escolha das portas lógicas CMOS deu-se principalmente pela ampla utilização destes circuitos no mundo contemporâneo. Dentro deste micro-universo escolheu-se as portas NAND devido ao fato de se poder escrever qualquer circuito lógico através da combinação destas portas. Assim sendo, otimizando-se o processo construtivo de portas NAND com M entradas, considera-se ser possível otimizar, com esta técnica, o processo construtivo de quaisquer circuitos lógicos em tecnologia CMOS.

[23] Que neste ponto significa que fornecem respostas rápidas a um estímulo feito à entrada.
[24] Dado que dissipam uma potência cada vez menor, devido à pouca área que ocupam.
[25] Se comparados a outras tecnologias, num processo em escala.

### 1.8.5 Síntese Hipotética e Metodológica

Em síntese, este trabalho terá duas abordagens: uma propriamente bibliográfica e teorética, característica dos Estudos de Estado da Arte; outra pragmática e técnica, na implementação de um Caso de Estudo Específico que ilustre, alimente e valide seus pressupostos teóricos. Para tanto, destacam-se inicialmente as hipóteses levantadas (figura 3) e resume-se a metodologia técnica (figura 4).


Figura 3 – Hipóteses


Figura 4 – Metodologia Técnica

# 2 Revisão Conceitual de Microeletrônica

Neste capítulo são introduzidos os principais elementos de Microeletrônica que servem de base para a compreensão do problema em questão, especialmente para o caso de estudo no qual aplica-se a técnica dos Algoritmos Genéticos. [1]

## 2.1 Introdução

Um requisito primário de um circuito digital é, obviamente, sua capacidade de realizar a função para a qual fora projetado. Outro requisito fundamental é que desempenhe essa função num tempo aceitável (definido pelo projetista) e cada vez mais rápido. O comportamento mensurado de um circuito manufaturado geralmente desvia da resposta esperada devido às variações no processo de fabricação: dimensões, tensão "*threshold*" [2], variações de corrente. Além disso, a presença de ruídos também é inerente ao processo e implica na adição de capacitâncias para filtrá-lo. O parâmetro "*steady-state*" [3] do *gate* mede o quão robusto é o circuito apesar das variações de manufatura e os ruídos do processo. Estes componentes são inerentes a todo circuito CMOS, e para lidar bem com eles é preciso compreender bem como os sinais digitais são representados no universo da Eletrônica.

## 2.2 A importância do inversor e o tempo de propagação

### 2.2.1 O Surgimento do Inversor CMOS

O inversor CMOS é o mais simples circuito estático. Constituido apenas de um transistor PMOS e um NMOS, ele traz em si o "esqueleto" de toda lógica CMOS de forma que toda porta lógica nesta tecnologia pode ser reduzida, em análise, para a investigação de inversores equivalentes (FERREIRA, 2004).

O inversor CMOS é construido por um transitor NMOS em *pull-down* [4] e por um transistor PMOS em *pull-up* [5] de forma que o transistor PMOS transmite o sinal da fonte para a saída, carregando a carga, e o transistor NMOS é bom para descarregar, como pode ser observado na figura 5.

---

[1] Caso o leitor tenha interesse em revisar conceitos mais básicos da Microeletrônica, poderá fazê-lo no Apêndice A.
[2] Também chamada de tensão limiar, $V_T$.
[3] Também chamado parâmetro de comportamento estático.
[4] Este termo se dá devido ao fato de transistores NMOS não conduzirem bem o 1 lógico, ou seja, o nível lógico alto.
[5] Este termo se dá devido ao fato de transistores PMOS conduzirem melhor o 1 lógico.

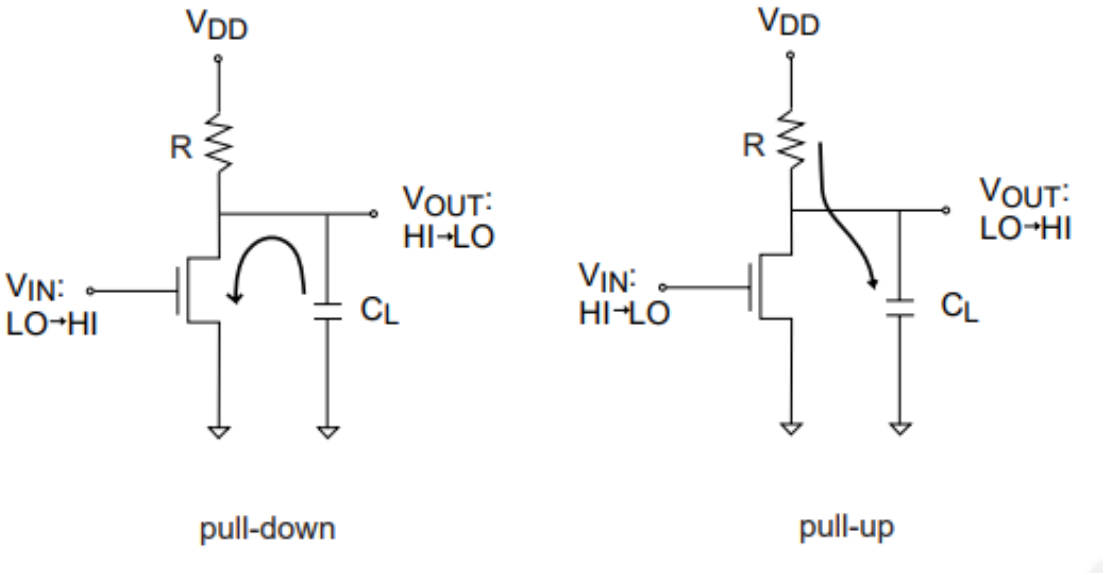



Figura 5 – Inversor NMOS com resistor pull-up

Uma grande dificuldade para o desenvolvimento de circuitos digitais reside no fato de que, com o aumento da margem de ruído, aumenta-se o valor absoluto do ganho do transistor, implicando em:

- Aumento da resistência de *pull* do transistor, que acarreta no aumento do tempo de troca $Low \rightarrow High$ ($tp_{LH}$), que, por sua vez, limita o carregamento de todos os circuitos acoplados em cascata [6];
- Aumento da condutância interna do transistor [7], implicando no aumento de $W$ e, consequentemente do transistor e, consequentemente do preço do dispositivo.

Estas relações são biderecionais, de forma que o aumento das dimensões do transistor implicam, evidentemente, no aumento de sua condutância e da margem de ruído.

Este é o famoso *trade-off* entre velocidade do circuito e margem de ruído. Aumentar a velocidade do circuito, implica em diminuir sua margem de ruído.

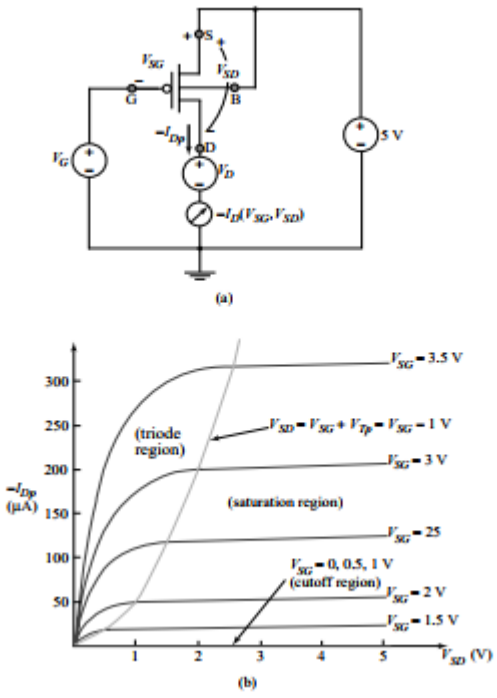
[6] Topologia em cascata é também chamada topologia de *lógica gate*: é uma rede de transistores que conecta a saída de um estágio ao gate do outro, de forma que os primeiros estágios fornecem corrente para os estágios seguintes

[7] $I_D = v_{sat}Q = v_{sat}WC_OX(V_{GS} - V_T)$ e $g_m = \frac{\partial I_D}{\partial V_{GS}}$

Durante o *pull-up* torna-se, pois, necessário obter uma alta corrente para ter-se uma troca rápida de alto para baixo e uma resistência incremental para garantir a margem de ruído.

Para evitar este problema, sugere-se implementar uma fonte de corrente na entrada do circuito. Desta forma, tem-se baixa resistência e alta corrente.

Usa-se, para tanto, um circuito PMOS em configuração de fonte de corrente *pull-up* (figuras 6 e 7).




pullup.PNG

Figura 6 – Fonte de Corrente com PMOS em Pull-up

Este circuito oferece características de circuito inversor com alta margem de ruído e troca rápida de alto para baixo, alta resistência incremental e corrente constante para carregar a capacitância de carga. Entretanto, quando $V_{IN} = VDD$ há um caminho direto de corrente entre a fonte de alimentção e o terra (GND) de forma que haverá um consumo constante de corrente, mesmo se o inversor estiver inativo.

É justamente para resolver este problema que se apresenta a topologia convencional do CMOS, divulgada em toda bibliografia de Circuitos Digitais (figura 8).

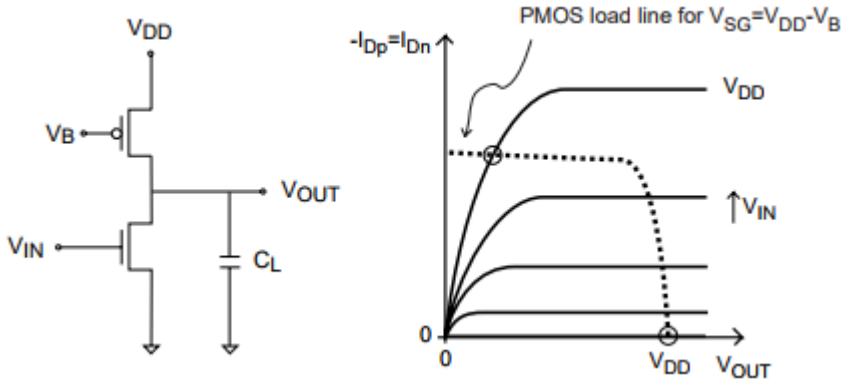



Figura 7 – PMOS funcionando como fonte de corrente pull-up no circuito

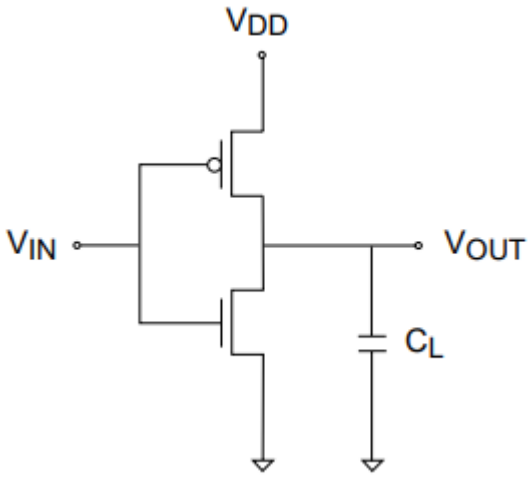



Figura 8 – Topologia do Inversor CMOS

### 2.2.2 Operação Básica do Inversor

Neste circuito, quando $V_{IN} = 0 \Rightarrow V_{OUT} = VDD$. A tensão de entrada começa a crescer e, até um certo instante, não supera a tensão de *threshold*. A carga ainda não começou a ser carregada, porém transistor PMOS começou a funcionar, e tem-se a seguinte configuração:

$$V_{GSn} = 0 < V_{Tn} \Rightarrow \textbf{NMOS OFF}$$
$$V_{GSp} = VDD > -V_{Tp} \Rightarrow \textbf{PMOS ON}$$

Depois deste primeiro instante, a tensão de entrada supera a tensão *threshold* e tem-se que a carga ainda não foi carregada, embora já tenha começado a haver algum fluxo. Assim sendo $V_{OUT} \cong 0 \Rightarrow V_{DS} \cong VDD$. Desta forma:

$$PMOS \Rightarrow |V_{DSp}| < V_{GSp} - V_{Tp} \Rightarrow LINEAR$$
$$NMOS \Rightarrow V_{DSn} \geq V_{GSn} - V_{Tn} \Rightarrow SATURAÇÃO$$

A tensão de entrada continua aumentando até que:

$$V_{DSp} \geq V_{GSp} - V_{Tp}$$

De forma que agora ambos transistores estão em saturação. Ocorre, pois, a inversão:

$$NMOS \Rightarrow |V_{DSn}| < V_{GSn} - V_{Tn} \Rightarrow LINEAR$$
$$PMOS \Rightarrow V_{DSp} \geq V_{GSp} - V_{Tp} \Rightarrow SATURAÇÃO$$

Por fim, a tensão de entrada supera $VDD - |V_{Tp}|$ e:

$$V_{OUT} = 0$$

A figura 9 mostra a curva característica do inversor destacando-se todos estes pontos.

Com este simples modelo [8] de funcionamento, permite-se compreender uma série de características dos circuitos CMOS que se aplicam a todas as portas lógicas nesta tecnologia:

1. Os níveis lógicos independem das dimensões dos transistores, todavia estas são importantes para o tempo de resposta, imunidade ao ruído, potência consumida, etc.

[8] Na figura 9, *cuttof* é região de corte, *triod* a região linear ou triodo e *saturation* a de saturação.

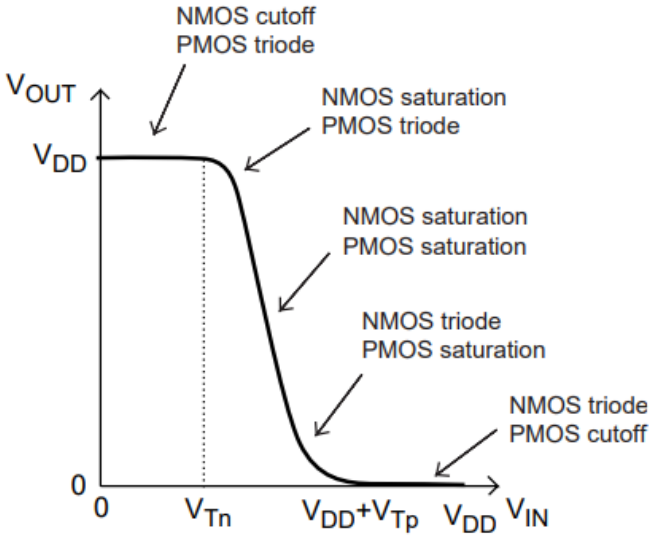



Figura 9 – Curva característica do CMOS

2. A variação de $V_{OUT}$ tem a mesma amplitude da tensão de alimentação implicando na melhor situação possível para a imunidade de ruído, *ide est* margens de ruído elevadas.

3. Em regime estacionário sempre há um caminho entre a tensão de saída e a alimentação, implicando em baixa impedância de saída, sendo assim mais uma proteção contra ruídos e demais perturbações.

4. Em regime estacionário não há caminho direto entre a fonte de alimentação e o terra, de forma a não haver consumo de energia quando o inversor estiver inativo.

5. Uma vez que o *gate* dos transistores CMOS é um isolante perfeito, tem-se que a impedância de entrada é elevadíssima [9].

### 2.2.3 Atraso de propagação

O atraso de propagação [10] é o tempo medido entre a entrada e a saída, ou seja, indica com que rapidez a saída de uma porta digital responde à entrada.

[9] Implicando em corrente nula em regime estacionário e *fan-out* infinito. Ainda que o *fan-out* em regime estacionário seja irrelevante, esta característica persiste em regime transitório onde o *fan-out* interfere.
[10] *Propagation Delay.*

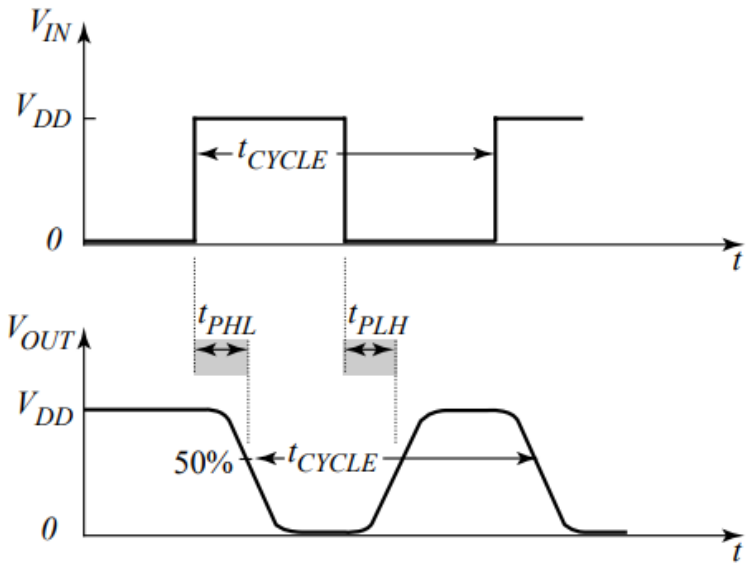



Figura 10 – Ilustração do tempo de propagação usando uma onda quadrada na entrada do inversor (SODINI, 2017).

Como apresenta (FERREIRA, 2004), "o tempo de propagação é medido entre o meio da excursão do sinal de entrada e o meio de excursão do sinal da saída 10 [11]. O atraso associado a uma comutação $H \rightarrow L$ na saída designa-se por $tp_{HL}$[12]; para uma comutação $L \rightarrow H$ é $tp_{LH}$[13]". Em geral, o tempo de subida é diferente do tempo de descida e o atraso na propagação é dado pela expressão 2.1 [14].

$$tp = \frac{1}{2}(tp_{HL} + tp_{LH}) \tag{2.1}$$

Onde se pode deduzir como se vê em (SODINI, 2017):

$$tp_{HL} = C_L \left[ \frac{V_{THN}}{\frac{1}{2}\mu_N C_{OX} \frac{W_N}{L_N}(V_{DD} - V_{THN})^2} \right] \tag{2.2}$$

A propagação típica para a maioria dos casos onde se tem um inversor CMOS é de 100ps.

[11] Supõe-se naturalmente que o sinal de saída comute devido à comutação da entrada.
[12] Chamado tempo de descida.
[13] Chamado tempo de subida
[14] Uma vez que é medido entre 50% dos percursos de alto para baixo e baixo para alto.

Lógicas complexas possuem tipicamente 10 a 50 atrasos de propagação, sendo considerado bom um tempo de propagação de $5ns$ (SODINI, 2017).

Em eletrônica, para se calcular o tempo de propagação considera-se a carga ligada à saída do circuito, modelada na forma de uma capacitância ($C_L$).

A capacitância de carga consiste da soma da capacitâncias de entrada dos próximos estágios, da capacitância parasita do dreno, do *bulk* [15] e das capacitâncias das trilhas.

A capacitância exata dos próximos estágios é de difícil cálculo, exigindo um modelo não-linear bastante acurado, considerando as transições de cada um dos capacitores NMOS e PMOS de todos os estágios seguintes das regiões de triodo para saturação e corte e a extração da capacitância em função da corrente, de forma que uma simples porta NAND (como é o caso deste trabalho) já exigiria um cálculo deveras oneroso [16]. Por isso, opta-se por se fazer uma estimativa da capacitância de entrada dos próximos estágios.

Desta forma, a capacitância total pode ser calculada por:

$$C_L = C_G + C_{DB} + C_{wire} \tag{2.3}$$

Onde:

- $C_G$ é a capacitância do *gate*;
- $C_{DB}$ é a capacitância do dreno e seu *bulk*;
- $C_{wire}$ é a capacitância da trilha.

## 2.3 Cálculo de $C_L$

### 2.3.1 Capacitância de entrada dos próximos estágios

Uma boa aproximação para a capacitância dos próximos estágios é dada pela expressão 2.4:

$$C_G = (C_{ox})_P W.L_P + (C_{ox})_N W.L_N \tag{2.4}$$

[15] Do dreno e do contato do dreno.

[16] O que na maioria esmagadora dos casos é desnecessário e não oferece uma precisão que não possa ser compensada por outros ajustes, tais aqueles a que se propõe este trabalho.

### 2.3.2 Capacitância dos *gates* dos próximos estágios

A capacitância dos *gates* dos próximos estágios pode ser modelada na equação 2.5:

$$C_G = (C_{ox})_P \sum_P W.L + (C_{ox})_N \sum_N W.L \tag{2.5}$$

Como se observa na figura 11.

### 2.3.3 Capacitância parasita no dreno e no *bulk*

As figuras 12, 13 e 14 mostram as seções transversal e horizontal de um inversor CMOS.

O pior caso para o cálculo da capacitância parasita é a atrelada à região de depleção, dado que esta região tem um comportamento dimensional não-linear. No pior caso, pode-se considerar que a **capacitância de polarização zero** [17] ($C_{J0}$) é um elemento armazenador de carga durante o período transiente.

Também os poços de depleção [18] contribuem para a capacitância de depleção e pode ser expresso na equação 2.6.

$$C_{BOTT} = C_{Jn}(W_n L_{diff}) + C_{Jp}(W_p L_{diff}) \tag{2.6}$$

Onde:

1. $C_{Jn}$ e $C_{Jp}$ são a capacitância de polarização zero (em $fF/\mu m^2$) respectivamente das junções do NMOS e do PMOS.
2. Valores típicos destas capacitâncias são de $0,2fF/\mu m^2$

As "paredes laterais" [19] da região de depleção fazem uma contribuição adicional que pode ser calculada pelo modelo expresso na equação 2.7.

$$C_{SW} = (W_n + 2L_{diff_n})C_{JSW_n} + (W_p + 2L_{diff_p})C_{JSW_p} \tag{2.7}$$

Onde:

[17] *Zero-bias capacitance.*
[18] "Botton."
[19] *Sidewall.*

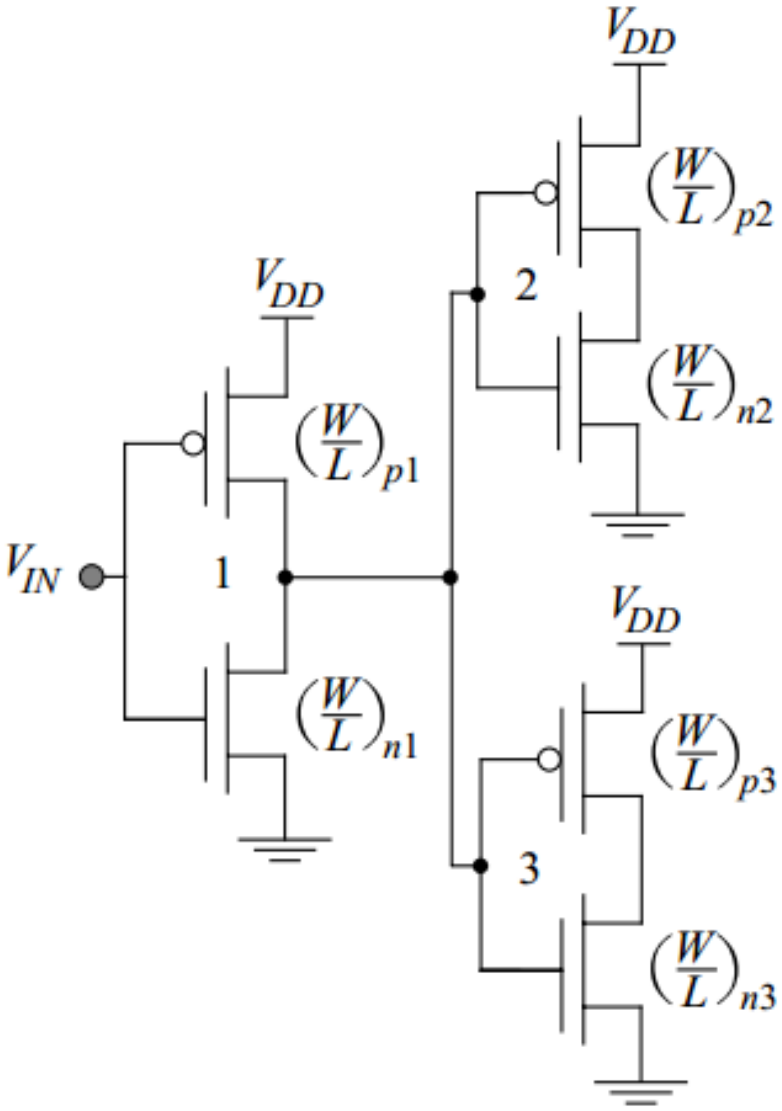



Figura 11 – Circuitos inversores acoplados em cascata.

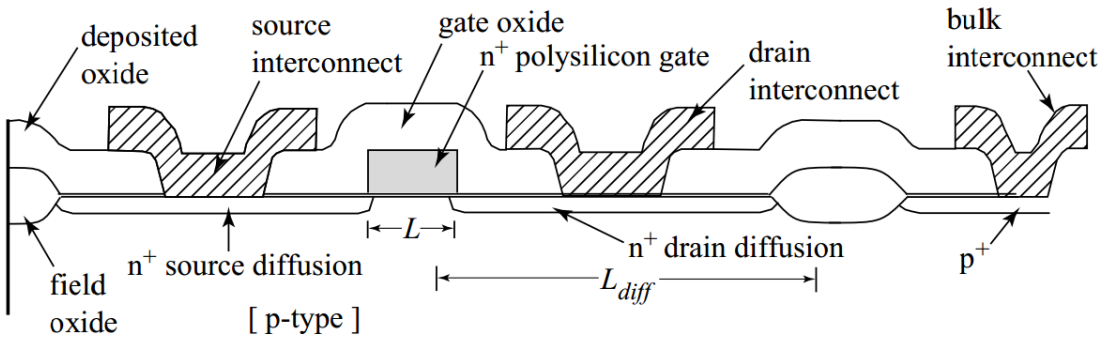



Figura 12 – Seção transversal do inversor

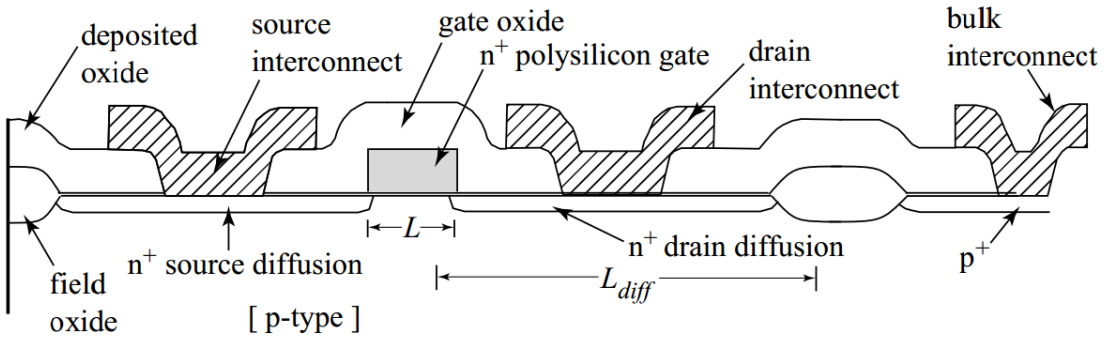



Figura 13 – Seção transversal do inversor

1. $C_{JSW_n}$ e $C_{JSW_p}$ são a capacitância de polarização zero (em $fF/\mu m$) respectivamente das paredes laterais das regiões de depleção do NMOS e do PMOS.

2. Os valores típicos de $C_{JSW_n}$ e $C_{JSW_p}$ são $0,5fF/\mu m$.

A soma de $C_{BOTT}$ com $C_{SW}$ fornecem a capacitância total da região de depleção 2.8.

$$C_{DB} = C_{BOTT} + C_{SW} \tag{2.8}$$

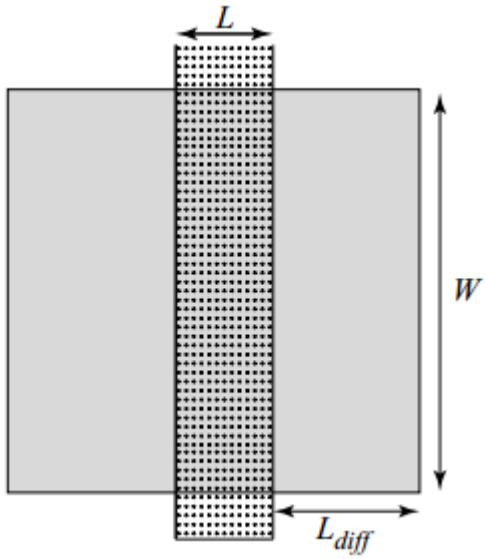



Figura 14 – Seção superficial do *Gate* de um iversor

### 2.3.4 Capacitância parasita nas trilhas

As trilhas [20] consistem de linhas de metal que conduzem a saída de um circuito à entrada do próximo estágio e pode ser calculada pela expressão 2.9 onde o índice $m$ representa o metal, e o índice $thickox$ representa a capacitância acumulada nas camadas de óxido termal (de espessura $0,5\mu m$) e óxido depositado (de espessura $0,6\mu m$).

$$C_{wire} = C_{thickox} W_m L_m \tag{2.9}$$

### 2.3.5 Capacitância de Carga Total: $C_L$

Desta forma, para sistemas digitais de larga escala ou de alta acurácia, a capacitância total de carga acumulada será significante para o tempo de resposta do circuito.

E esta pode ser calculada pela expressão 2.10.

$$C_L = C_G + C_{DB} + C_{wire} \tag{2.10}$$

[20] *Wires.*

Onde $C_{DB} + C_{wire}$ é a **capacitância parasita** 2.11.

$$C_P = C_{DB} + C_{wire} \tag{2.11}$$

### 2.3.6 Esforço lógico

O esforço lógico é o valor que reflete um conjunto suficiente de informações a respeito de uma topologia cascata para determinar o atraso de um circuito. Há ao menos três definições equivalentes para esforço lógico.

> **O esforço lógico de uma *lógica gate* é definido com o número de vezes que esta porta é pior para entregar a corrente de saída do que seria um inversor com idêntica capacitância de carga.**

Esta é uma definição bastante intuitiva. É notório que qualquer lógica será um transmissor de corrente pior que o inversor, uma vez que qualquer lógica terá mais transistores que um inversor. Ora, se os transistores desta lógica estão em série, logo esta lógica entregará menos corrente que o inversor que possui apenas um inversor no lado positivo ou negativo, uma vez que haverá perda de corrente devido à impedância do transistor. Se, por outro lado, o transistor tiver uma lógica em paralelo, não haverá perda de corrente devido à impedância propriamente de um segundo transistor, mas dado que a corrente tentará se dividir nos nós [21], todavia esta perda de tempo já é suficiente para esta lógica ser menos performática que um inversor.

> **O esforço lógico de uma *lógica gate* é definido como a razão entre a capacitância de entrada desta lógica para aquela de um inversor que entrega igual corrente de saída.**

Esta definição alternativa, mais próxima da linguagem matemática, é muito útil para se calcular o esforço lógico.

> **O esforço lógico de uma *lógica gate* é definido como o declive do atraso da porta pela curva de *fanout* dividida pela mesma relação no inversor.**

Esta definição alternativa é mais geométrica, sendo uma ótima alternativa para se medir o esforço lógico através de experimentos ou da simulação de circuitos.

[21] E de fato haverá uma mínima corrente de fuga

Duas questões diretas que daí surgem são: e para um circuito de múltiplas entradas, quantos sinais de entrada devem ser considerados quando se calcula o esforço lógico? E como agrupar tais sinais?

De onde deriva-se uma simples resposta: o esforço lógico total é o somatório de todos os esforços lógicos considerados simultaneamente.

Esses esforços se dividem em esforços de *input* (efetivo, quando o esforço lógico depende de uma única porta) e de *bundle* (quando o esforço lógico depende da combinação de várias entradas, como num multiplexador) [22].

Seja $\Psi$ o esforço lógico de um grupo $\gamma$ de $n$ transistores. Defini-se:

$$C_\gamma = \left( \sum_{i=1}^{n} C_i \right)_\gamma \tag{2.12}$$

Como a **capacitância do grupo de transistores**. E,

$$\Psi_\gamma = \frac{C_\gamma}{C_{inv}} \tag{2.13}$$

Como o esforço lógico de um grupo de transistores.

### 2.3.7 As portas lógicas com CMOS

Vai-se tentar implementar um circuito lógico da expressão 2.14.

$$L = A + B.C \tag{2.14}$$

Desse circuito lê-se: "A ou B e C". Assim, supõe-se que o circuito deve-se permitir uma corrente da entrada A sempre que permitir simultaneamente corrente das entradas B e C. De forma que, considerando NMOS, tem-se a figura 15.

Se, entretanto, os sinais chavearem invertendo-se, perde-se a lógica. Por esse motivo, usa-se a lógica PMOS para quando a lógica se inverte, como na figura 16. A concatenação das duas lógicas, oferece a lógica inversora barrada. Coloca-se na saída um inversor e tem-se a lógica desejada. Para não precisar acoplar o inversor, basta negar a lógica desejada previamente como mostra a figura 17.

Assim tem-se sempre a estrutura da figura 17 e os seguintes passos para a implementação de uma lógica CMOS:

[22] Onde $\sigma^*$ é agora adotado para o sinal multiplexado e $\bar{\sigma}$ para o sinal complementar.

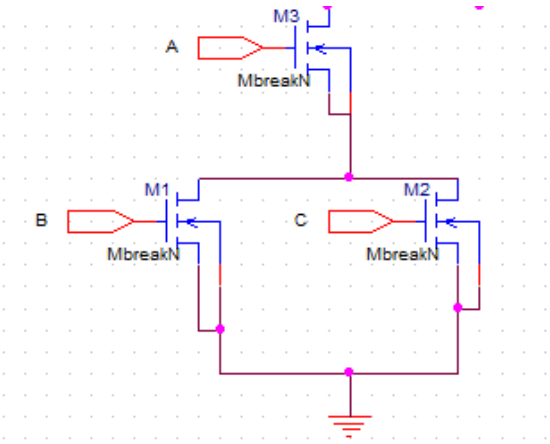



Figura 15 – Lógica *Pull-Up*

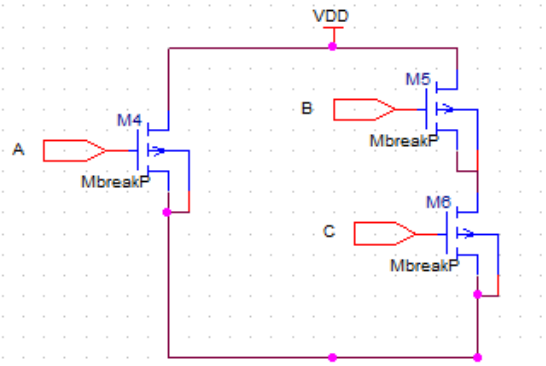



Figura 16 – Lógica *Pull-Down*

1. Negar a lógica desejada.
2. Implementar a lógica dentro da barra, com NMOS.
3. Implementar a lógica inversa à de dentro da barra, com PMOS.
4. Acoplar os circuitos.

Desta forma, pode-se generalizar os circuitos CMOS estáticos na estrutura inversora complementar da figura 18 onde **PUN** e **PDN** são redes lógicas duais, $In1...InN$ representam as entradas e $F(In1, In2, ..., InN)$ a função lógica implementada.

O *delay* depende do padrão das entradas:

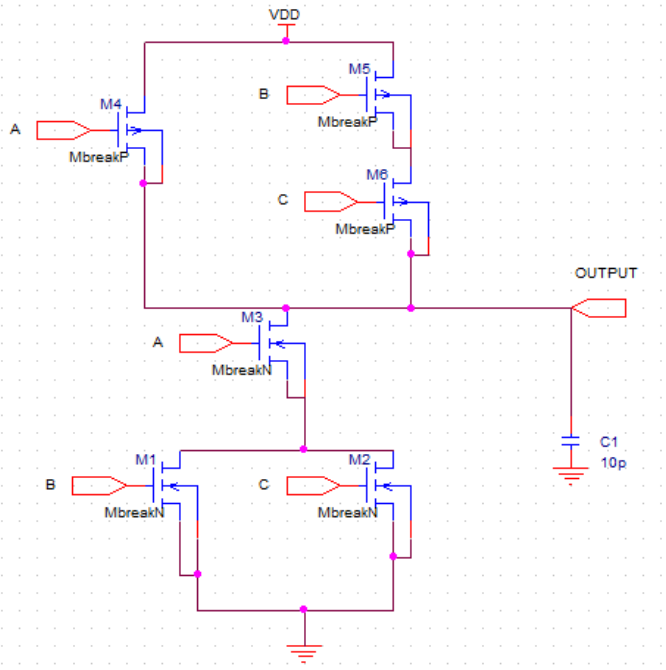



Figura 17 – Lógica inversora *Pull-Up,Pull-Down*

Na transição $L \rightarrow H$, com ambas as entradas em nível lógico baixo, o atraso calculado resultará em na equação 2.15.

$$t_{pLH} = 0.69\frac{R_p}{2}C_L \tag{2.15}$$

Com uma entrada a nível lógico alto, resultará na equação 2.16.

$$t_{pLH} = 0,69R_pC_L \tag{2.16}$$

Já na transição $H \rightarrow L$, com ambas entradas a nível lógico alto tem-se a equação 2.17.

$$t_{pLH} = 0,69 \times 2 \times R_p \times C_L \tag{2.17}$$

A figura 19 ilustra o que fora até aqui explicado. Considere:

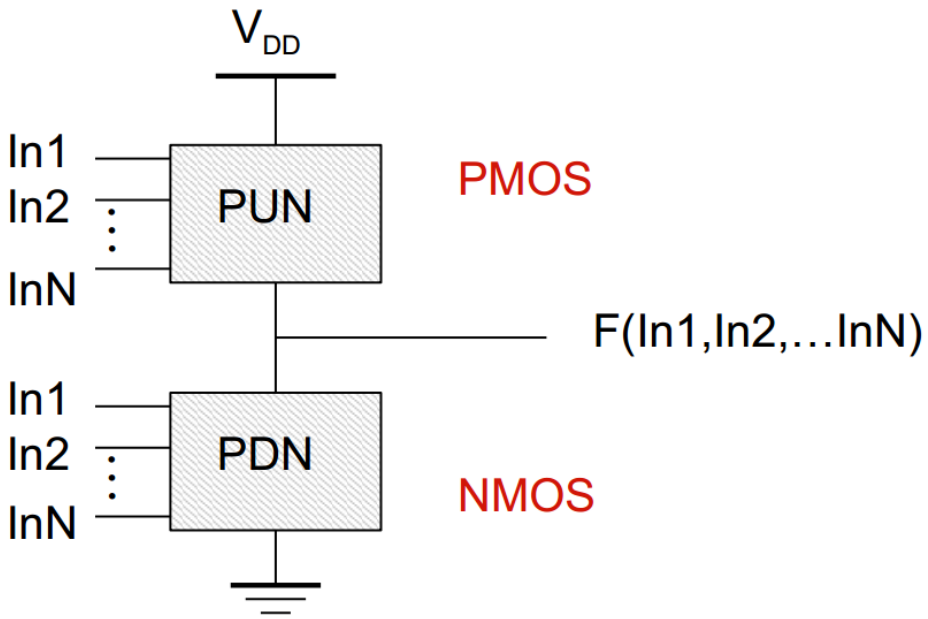
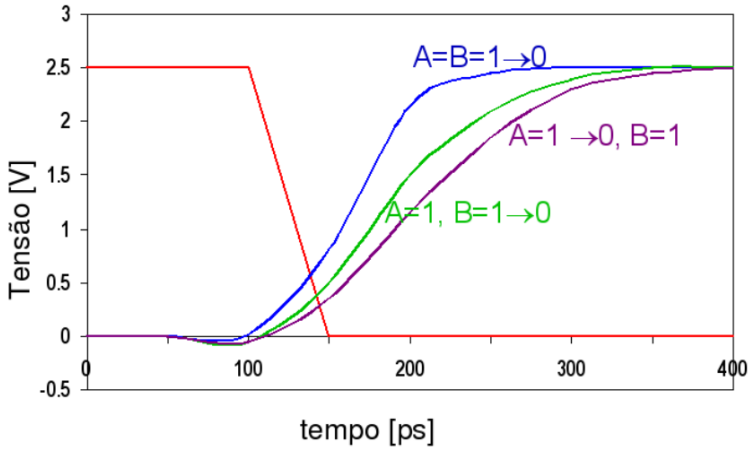



Figura 18 – Generalização para circuitos CMOS estáticos complementares (RABAEY; CHANDRAKASAN; NIKOLIC, 2004)

$$NMOS = 0,5\mu m/0,25\mu m \tag{2.18}$$

$$PMOS = 0,75\mu m/0,25\mu m \tag{2.19}$$

$$C_L = 100fF \tag{2.20}$$




| Padrão de entrada | Atraso (ps) |
|---|---|
| A=B=0→1 | 67 |
| A=1, B=0→1 | 64 |
| A= 0→1, B=1 | 61 |
| A=B=1→0 | 45 |
| A=1, B=1→0 | 80 |
| A= 1→0, B=1 | 81 |

Figura 19 – Atraso em função do padrão das entradas (FERREIRA, 2004)

# 3 Ciência da Computação e a Inteligência Artificial

Ao longo das últimas décadas, os algoritmos evolucionários tornaram-se tópicos de pesquisa muito populares devido à sua eficiência na otimização de problemas intratáveis [1].

Embora uma vastidão de métodos tenham sido propostos, três deles se destacam na maioria das pesquisas, sendo, portanto, os mais populares: os Algoritmos Genéticos (GA [2]), o Enxame de Partículas (PSO [3]) e o Algoritmo de Evolução Diferencial (DE [4]) (KACHITVICHYANUKUL, 2012).

Neste capítulo, primeiramente se explica superficialmente o PSO e o DE e faz-se um breve comparativo entre estas três principais técnicas evolucionárias e, posteriormente, estuda-se os principais conceitos da programação evolucionária, principalmente dos algoritmos genéticos.

## 3.1 Apresentação das principais técnicas evolucionárias

### 3.1.1 Passos Genéricos de um Algoritmo Evolucionário

Há três passos que são genéricos a todo algoritmo evolucionário: a geração da população inicial, a avaliação dos indivíduos desta população e a geração de uma nova população baseada em modelos de fenômenos naturais.

Como bem explica (KACHITVICHYANUKUL, 2012), "após a inicialização, a popuação é avaliada e são conferidos os critérios de parada do algoritmo. Se nenhum dos critérios forem satisfeitos, uma nova população e gerada e o processo se repete até que sejam satisfeitos estes mesmos critérios". Critérios de parada podem ser estáticos (como o número de iterações ou um tempo específico) ou dinâmicos. Um exemplo de critério dinâmico é executar o algoritmo até que um percentual $k$ da população se aproxime de uma já conhecida ou esperada solução ideal [5] (KACHITVICHYANUKUL, 2012).

É de suma importância ao se pensar em resolver problemas com algoritmos evolucionários (em especial os genéticos) [6], atentar para a representação do conjunto de possíveis soluções que gerará a popução inicial. Há particularmente duas formas viáveis de representação: uma represen-

[1] Poderá encontrar a explicação sobre esta classe de problemas na Seção 3.2
[2] Sigla para *Genetic Algorithm*.
[3] Sigla para *Particle Swarm Optimization*.
[4] Sigla para *Differential Evolution*.
[5] Há casos em que uma combinação de M critérios é usada para otimizar o algoritmo.
[6] Além de atentar para os fatos já expostos de sua aplicabilidade, como não ser possível ou viável resolver o problema de forma exata por métodos de Cálculo Numérico.

tação direta, aplicável instantaneamete no modelo a ser otimizado [7] e uma representação codificada [8]. Esta segunda forma é chamada indireta e tem uma certa preferência para soluções muito complexas onde se considera o modelo computacional de um indivíduo como sendo uma composição de diversas características que visem a solução so problema. [9].

### 3.1.2 PSO: O Enxame de partículas

O PSO surgiu no Congresso de Computação Evolutiva, em 1995, apresentado no artigo de Kennedy e Eberhart (KENNEDY; EBERHART, 1995) [10]. Este comportamento é inspirado pelas relações ecológicas e os comportamentos coletivos e colaborativos que muitas espécies desempenham de forma que consigam realizar seus ciclos de vida (ciclos de reprodução, migração, etc.). Exemplos claros destes processos são o movimento migratório os pássaros ou de um cardume de peixes. O algoritmo trabalha em duas frentes bem definidas: o comportamento empírico e meta-heurístico dos indivíduos, e o comportamento empírico e meta-heurístico do coletivo. Na primeira frente, o algoritmo guia-se pela experiência pessoal de cada indivíduo ($PBest$) e, na segunda frente, pela experiência geral ($GBest$) do coletivo (enxame). Essa experiência se caracteriza pela memória acumulada dos indivíduos (comportamento empírico). Já o fator meta-heurístico se dá pelo movimento das partículas da posição atual para uma outra posição (meta-heurística) conduzida por fórmulas matemáticas simples como posição e velocidade das partículas (KACHITVICHYANUKUL, 2012). O pseudo-código seguinte descreve os passos de um algoritmo PSO fundamental.

```
1 BEGIN:
2         Population = Generate_Initial_Population()
3
4         DO:
5
6                 FOR EACH Individual OF Population DO:
7                         Evaluate_Individual_Fitness()
8                 END FOR
9
```

[7] Que foi a adotada no caso de estudo apresentado neste trabalho, devido a ser uma melhor prática de programação (MARTIN, 2017).

[8] Geralmente adotada em soluções em linguagens de baixo nível, ou que utilizem *frameworks* que não dão suporte à outra forma de representação.

[9] De forma particular, o autor não encontra na Bibliografia uma justificativa viável para se violar as boas práticas de programação e usar um modelo complexo de indivíduo, dado que sempre se pode abstrair a complexidade do problema na função de avaliação, como será mostrado em nossa implementação do PSO feita no Capítulo 5.

[10] Alguns autores como (MIRANDA; FONSECA, 2002), não consideram que o PSO possa ser classificado como computação evolutiva, por não possuir as características *sine quibus non* da mesma. Todavia, aproxima-se desta quanto ao *questio exames*. A questão a respeito da classificação mais adequada será melhor analisada no Apêndice D. Por hora, assuma o leitor que o PSO, apesar da classificação dada por (BENI; WANG, 1993) (Inteligência dos Enxames), como sendo uma espécie de algoritmo evolucionário, até mesmo porque muitos autores como (DAS; ABRAHAM; KONAR, 2008) tratam estes algoritmos como sistemas evolucionários (ALAM, 2016).

```
10                  Update_PBest()
11                  Update_GBest()
12                  Update_Velocity()
13                  Update_Position()
14
15          WHILE IsNotStopCriteriaMet();
16
17  END
```

Os dois principais fatores que conferem, para alguns autores, ao PSO um caráter evolutivo é que suas posições e volocidades "evoluem" de foma meta-heurística. Observa-se que as diversas implementações de PSO não requerem nem operações genéticas, nem mesmo algum outro processo árduo para esta atualização, apenas uma simples operação aritimética de números reais.

Um importante elemento comparativo reside no fato de os PSOs não exigirem, em momento algum de sua implementação, qualquer ordenação dos valores de *fitness* [11], o que confere a estes algoritmos uma enorme vantagem computacional em cima dos GAs.

### 3.1.3 DE - Evolução Diferencial

O algoritmo de DE foi proposto ao mesmo tempo do PSO (STORN; PRICE, 1997) para a otimização global em um espaço de busca contínuo utilizando-se meta-heurísticas. Ele é perfeitamente inserido dentre os algoritmos evolucionários de forma a atender a todos os *quid sit* sem sequer necessitar de alguma abstração. Faz uso de processos darwinianos como mutação, cruzamento e seleção a fim de performar uma busca estocástica [12]. Dada sua natureza estocástica, como todo algoritmo desta categoria também ele pode ser aplicado à solução de problemas descontínuos - uma vez que não necessita de derivadas. Uma vez que faz uso dos processos darwinianos, acaba sendo mais lento que o PSO. Todavia, segundo segundo Godfrey e Donald (ONWUBOLU; DAVENDRA, 2009), isso não justifica toda essa diferença de tempo. Por outro lado, funciona muito bem com uma população inicial pequena. Nos algoritmos DE, a solução (indivíduo) é representada por um vetor de dimensão $D$. Uma população inicial de $N$ indivíduos é formada e avaliada pela função de *fitness*. Normalmente, esta população é criada por uma distribuição de probabilidades uniforme, ou então por outros algoritmos que lhe conferem características específicas do problema.

A grande diferença do algoritmo DE para o PSO e o GA reside em seu processo de geração de novos indivíduos [13]; de maneira especialíssima, em seu processo de mutação.

[11] Principalmente devido ao processo de atualização das velocidades e posições que já consideram em si as experiências das partículas e do enxame (valores oriundos de operações anteriores).
[12] Ou seja, aleatória.
[13] Os novos indivíduos são uma combinação de muitos indivíduos candidatos da última iteração.

O processo se dá escolhendo [14] três indivíduos distintos dentre todos os $N$ que formam a população inicial.

Se forem $X_i$ os indivíduos da população $P_c$ corrente, e $X_\alpha$, $X_\beta$ e $X_\gamma$ os indivíduos selecionados, pode-se escrever a operação de mutação na expressão 3.1.3.

$$P_M = X_\alpha + \Phi(\delta) \tag{3.1}$$

$$\delta = X_\beta - X_\gamma \tag{3.2}$$

Onde $P_M$ é a população de indivíduos mutados, $\Phi$ é o fator de mutação e $\alpha, \beta, \gamma \in \mathbb{N}$.

$\Phi$ é o parâmetro principal de um algoritmo DE.

O segundo passo será criar o chamado **vetor de teste** realizando o *crossover* entre o vetor mutante e o vertor alvo. Os indivíduos que serão cruzados são geralmente selecinados aleatoriamente. Todavia, novamente nada impede que um processo meta-heurístico seja aplicado.

Em seguida, ordena-se, através da seleção, os indivíduos com o melhor *fitness*.

O pseudo-código deste algoritmo é apresentado a seguir:

```
BEGIN:
        Population = Generate_Initial_Population()

        GlobalBestVector = Population

        FOR EACH Individual OF Population DO:
                        Evaluate_Individual_Fitness()
        END FOR

        DO:
                TrailVector = GenerateTrailVector()

                FOR EACH Individual OF Population DO:
                        Evaluate_Individual_Fitness()
                END FOR

                PopulationBest = SelectBetterVector(Population, TrailVector
                    )
```

[14] Geralmente de forma aleatória, mas nada impede de se envolver aqui uma meta-heurística.

```
18
19                  GlobalBestVector = PopulationBest
20
21          WHILE IsNotStopCriteriaMet();
22
23  END
```

## 3.2 Os problemas intratáveis

Define-se a complexidade de um problema computacional como o tempo consumido (no pior caso) pelo melhor algoritmo possível para o problema [15].

Há uma infinidade de problemas computacionais e, consequentemente, uma infinidade de algoritmos para solucioná-los. Encontrar o nicho especialíssimo onde se localiza o problema em questão exige-nos o emprego da arte da divisão. Sem muito adentrar em como bem dividir [16], aceita-se neste trabalho a corrente divisão formal empregada pela Academia em problemas de três principais classes: a classe P dos problemas polinomiais (ditos **tratáveis** ou **fáceis**), a classe NP dos problemas **intratáveis** e, por fim, a classe dos problema **indecidíveis**. Um problema é um **não-polinomial** [17] se não existir um algoritmo polinomial que o resolva de forma exata e, por fim, a classe dos problemas **indecidíveis** ou *não solucionáveis* para os quais não existe algoritmo algum [18].

O problema do estudo de caso específico de nosso trabalho situa-se na classe dos problemas decidíveis, porém difíceis, para os quais possivelmente não seja possível encontrar algoritmos que os resolvam em tempo polinomial: os quais já foram definidos como intratáveis, como se resume na figura 20.

## 3.3 Teoria dos Algoritmos Genéticos

Os algoritmos genéticos se encaixam dentro de um conjunto de algoritmos chamados evolucionários, os quais, através de modelos computacionais, utilizam o modelo evolucionário da teoria darwiniana (donde origina-se o nome) [19] para se resolver problemas num espaço de busca [20]. Pode-

[15] É bem verdade que *o melhor algoritmo possível* não necessariamente é o melhor algoritmo conhecido.
[16] Nem tampouco investigar a fundo se esta divisão fora bem feita - uma vez que foge ao escopo deste trabalho.
[17] Aqui o termo foi empregado de forma meramente mnemônica, uma vez que não se deve jamais ler a sigla NP desta forma.
[18] A bem da verdade, sabe-se de antemão que esta divisão é incompleta (ARORA, 1998)
[19] Por mais que o nome seja referência a esta teoria, o autor mostrou através deste trabalho que os processos evolucionários em si não são mérito próprio do darwinismo, mas conseqüências diretas da Genética de Mendel.
[20] O conjunto-universo contendo todas as possibilidades de solução de um determinado problema.

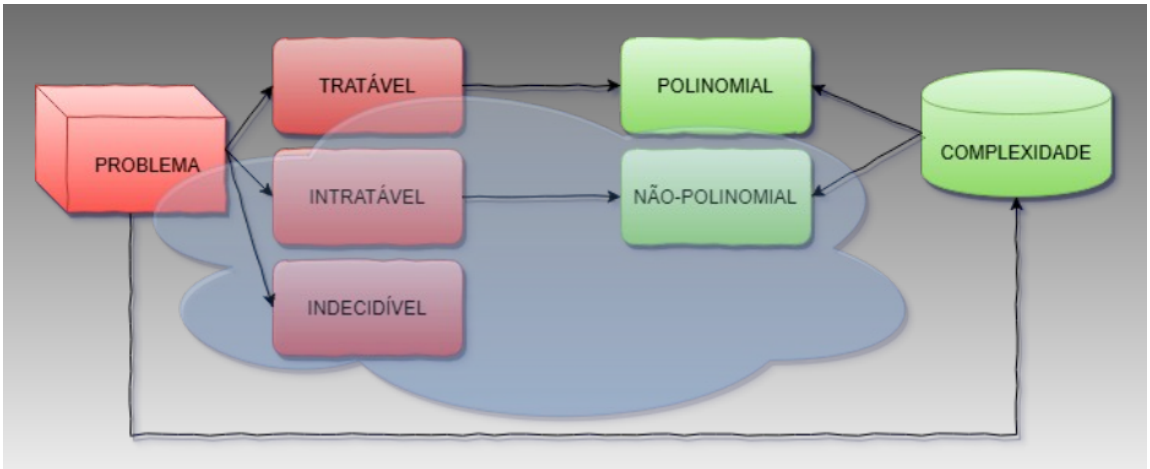



Figura 20 – Resumo dos Tipos de Problemas e onde se encontra o do estudo de caso

se dizer de maneira geral, que um algoritmo genético simula a evolução das espécies [21] de forma que um **indivíduo** evolui ou morre dependendo de sua adaptabilidade ao meio e que esta "evolução" se dá através da aplicação de operadores genéticos, como *crossover* [22], mutação, dentre outros.

## 3.4 Teorema da Inexistência de Almoço Grátis

[23]

Os Algoritmos Genéticos se enquadram dentro do Universo dos Algoritmos de Busca. Neste Universo de problemas, há de se pensar na eficiência do código. Desde a década de 1960 o interesse por encontrar algoritmos de otimização que atendesse a propósitos gerais cresceu avassaladoramente. Este crescimento deu-se até o final da década de 1990, quando, em 1995, David H. Wolpert e William G. Macready publicaram seu artigo *"No Free Lunch Theorems for Optimization"*. Percebendo este crescimento no interesse por esta classe de algoritmos, viram ser importante compreender a relação entre quão performático é um dado algoritmo $A$ e o problema de otimização $P$ sobre o qual ele executa. Com este artigo, eles responderam às seguintes perguntas, dada uma abundância conhecida [24] de algoritmos de otimização e de problemas nos quais eles possam auxiliar a encontrar a solução:

- Como escolher o par ideal de algoritmo-problema?

[21] A priori pela teoria Darwiniana.
[22] Também chamado **operador de recombinação genética**.
[23] *No Free Lunch Therorem, NFL.*
[24] E uma vastidão ainda maior desconhecida de algoritmos.

- A escolha de tais pares pode ser feita de forma não-heurística, mas através de uma análise formal?
- Qual é o "esqueleto" matemático da teoria de otimização antes de ser revestida pela "carne" das distribuições probabilísticas de um contextos particular e de ser imposto um conjunto de problemas de otimização?
- O que a Teoria da Informação e a Análise Bayesiana contribuem para a compreensão destas questões?
- Quão - *a priori* - genérico são os resultados performáticos de um algoritmo aplicado a uma certa classe de problemas e a outra? E como medir esta generalização a fim de poder-se comparar programaticamente tais algorítmos?

Restringindo-se a otimização combinatorial num espaço de busca finito (embora tomado tão grande quanto se queira), com custos também finitos, sendo este limitado pela arquitetura do processador [25], eles demonstraram de maneira formal em seu artigo que, se um dado algoritmo $F$ é melhor que outro ($G$) em uma determinada classe de problemas $P$ deste universo, haverá indubitavelmente outra classe de problemas ($P'$) neste universo no qual $B$ seja mais perfomático que $A$.

Disto deriva-se diretamente que nenhum algoritmo genérico conseguirá ter um desempenho melhor que um algoritmo específico neste universo.

Este teorema é importante para este trabalho por diversas razões além da histórica:

- Primeiramente porque por muitas vezes este teorema é levantado como bandeira para justificar a não-aplicabilidade dos Algoritmos Genéticos para solucionar um problema.

  De fato, esta questão deve ser levantada. Todavia, primeiramente, para tanto, deve-se colocar o problema em questão dentro do universo de aplicabilidade deste teorema. Em segundo lugar, que a utilização de algoritmos genéticos não escusa o pesquisador de modelar o problema computacionalmente e de considerar aspectos fundamentais do seu problema.

- Além disso, o NFL justifica nossa opção por desenvolver nosso próprio algoritmo genético ao invés de adotar uma solução genérica já pronta em diversos *frameworks*.

- "Escolher um algoritmo genético para resolver um problema não significa que você pode ignorar as peculiaridades do problema, todas as suas condições importantes e formulações matemáticas. Ao contrário: um algoritmo genético só funciona bem se embutir-se nele o máximo

[25] Em números de 32 ou 64 bits.

de conhecimento possível sobre o problema, tanto na função de avaliação, quanto na escolha de representação e dos operadores genéticos" (LINDEN, 2012). [26]

## 3.5 Terminologias e Características dos Algoritmos Genéticos

Embora a totalidade das bibliografias afirme que toda teoria de Algoritmos Genéticos se baseia exclusivamente na Teoria da Evolução Darwinista, há de se fazer justiça às demais pesquisas, sendo a mais importante delas, as descobertas do Monge Agostiniano e "Pai da Genética", Gregor Johann Mendel. Durante sua vida, Mendel publicou dois grandes trabalhos: "Ensaios com plantas Híbridas [27], que não abriga mais de 30 páginas e "Hierácias obtidas pela fecundação artificial".

Reservar-se-há, neste momento, expor os termos comuns utilizados na Teoria dos Algoritmos Genéticos e apontar sua origiem biológica e sua significação na computação.

- ***Fitness Function*:** também chamada de função de avaliação é uma função ou método computacional que se modela o problema em questão e a qual se deseja otimizar. Para algoritmos de otimização padrões tal função é conhecida como **função objetivo**. Sua origem terminológica é da teoria de máximos e mínimos matemática, motivo pelo qual, ainda que se convertendo em métodos em linguagens orientadas a objeto, mantém-se o nome "função". Geralmente, Algoritmos Evolucionários justificam sua aplicação quando esta função é multimodal [28], ou a complexidade computacional para otimizá-la por métodos matemáticos tradicionais é elevada, ou ainda quando a representação para a otimização é complexa.

- **Indivíduo:** é qualquer ponto no qual possa ser aplicada a *fitness function*. Em outras palavras, é o ente do modelo que se deseja otimizar. No caso deste trabalho, por exemplo, é o transistor CMOS, dado que a otimização do transistor CMOS implicará na otimização do circuito modelado na função de avaliação. O termo se origina da teoria genética de Mendel, onde o indivíduo era o pé de ervilhas na qual ele promovia a fertilização artificial.

  **Genoma:** é propriamente o indivíduo e é sinônimo encontrado na mesma teroria de Mendel.

  **Gene:** é cada elemento que constitui o Genoma, igualmente encontrado na teoria de Mendel. Em termos de engenharia, pode-se bem dizer que o gene é o **vetor de entradas de um indivíduo**.

[26] Esta nota do Ricardo Linden é de extrema importância para o que se pretende discutir filosoficamente a respeito das bases da Teoria dos Algoritmos Genéticos.
[27] Versuche über Pflanzen-hybriden
[28] Uma função multimodal é aquela que oferece mais de uma moda, ou seja, que oferece muitos máximos ou mínimos que se repetem num dado conjunto.

- **Heurística:** o termo se origina do matemático húngaro George Pólya [29] , com etimologia grega baseada na frase atribuída a Arquimedes (heureca - "'eu achei!"') na qual define um conjunto de regras simples tais quais o Paradoxo do Inventor [30], a concretização de um problema abstrato, buscas racionais ou irracionais. Em termos menos técnicos tratam-se de algoritmos de "chute", ou seja, de processos cognitivos utilizados em decisões não racionais, onde se vale de saltos lógicos e se ignora diversos elementos para tomar decisões de forma ráida. Diferente de um algoritmo convencional, as heurísticas não garantem a melhor solução nem sequer, muitas vezes, podem ser provadas logicamente. São métodos intuitivos que visam encontrar soluções satisfatórias. Heurísticas são utilizadas o tempo todo por nós, seres humanos: quando vamos atravessar uma rua, ou jogar na loteria, por exemplo. Os algoritmos genéticos são metaheurísticas que são propriamente heurísticas alteradas, uma heurística de alto nível designada para gerar uma heurística a fim de encontrar, num espaço de busca, uma solução plausível para um problema de otimização.

- **População:** Uma população é um conjunto de indivíduos. O mesmo indivíduo pode aparecer em mais de uma população.

- **Geração:** é o grupo de organismos produzidos aproximadamente no mesmo tempo; um grupo de organismos que estão no mesmo estado do ciclo de vida. Pode ser ainda o período médio entre o nascimento de um indivíduo até que ele reproduza e gere descendência. Pode também ser uma classe de indivíduos que se originaram de uma classe anterior, porém que sejam signficativamente distintos, ou com genes distintos. Quando Mendel descreveu as sucessivas gerações de ervilhas, ele falava dos sucessivos anos de descendências. No caso específico dos algoritmos evolucionários, abstrai-se o conceito de geração da definição biológica. O ciclo de vida de um indivíduo depende muito da técnica utilizada na implementação do algoritmo genético. De uma forma geral, a cada iteração do algoritmo genético uma nova população é gerada (ocorre uma reprodução). Essas sucessivas populações são chamadas gerações.

- **Diversidade:** Diversidade se refere à distância média entre os indivíduos em uma população. Na teoria de Mendel, eram as diferenças entre os genes dos indivíduos em uma mesma população. Uma população tem alta diversidade se a média das distâncias entre seus indivíduos for grande; e, consequentemente, baixa diversidade se a média das distâncias for pouca.

  **Pais e Filhos:** Para criar uma próxima geração, necessita-se de um pai e uma mãe. O algoritmo seleciona certos indivíduos da população, com características distintas, que são usados para a reprodução. No processo de reprodução ocorre o cruzamento e gera-se os filhos. A geração dos filhos garante a perpetuidade da espécie e transfere às gerações futuras, carac-

[29] Em seu livro "A arte de resolver problemas"
[30] Que consiste em tentar abordar um problema de maneira mais geral, para depois adentrar no modelo específico.

terísiticas das gerações anteriores. Geralmente, os algoritmos genéticos selecionam pais que melhor se adequem à *fitness function* (chamados *fittest*).

## 3.6 A estrutura de um Algoritmo Genético e os Principais Operadores

A estrutura dos Algoritmos Genéticos é uma abstração matemática dos mecanismos de seleção natural e da genética.

O pseudo-código a seguir mostra este funcionamento.

```
1  BEGIN:
2  Genetic Algorithm Main Function:
3          define number_of_generations G = N
4          initialize(Population P, G);
5          evaluate(P);
6          while G < MAXIMUM_OF_GENERATIONS, do:
7                  G = G + 1;
8                  [father, mother] = select_parents(P);
9                  son = recombination(father, mother);
10                 mutation(son);
11                 evaluate(son);
12                 add(son in P);
13                 delete_individual_following_a_rule(P);
14                 evaluate(P);
15 END
```

Na linha 4 a população de cromossomos é inicializada com N cromossomos. A inicialização da população pode ser feita de várias formas, mas a prática da área é utlizar a estratégia mais simples de inicialização, que consiste em simplesmente escolher aleatoriamente N indivíduos (LINDEN, 2012)[31]. Na linha 5 a população é avaliada. Neste ponto verifica-se a compatibilidade dos indivíduos gerados na população inicial com a função *fitness* [32]. Nas linhas 6 até o fim do pseudo-código, repete-se a cada geração ($G = G + 1$) a seleção dos pais segundo um critério de seleção [33], a recombinação gênica [34], a avaliação do indivíduo descendente (filho), sua adição na população, a exclusão dos indivíduos segundo algum critério [35] e nova avaliação da população.

[31] "Isso não se deve à preguiça dos desenvolvedores, mas sim ao fato de que a inicialização aleatória de forma geral gera uma boa distribuição das soluções no espaço de busca e o uso do operador de mutação de forma eficaz garante uma boa exploração de todo o espaço de busca"(LINDEN, 2012)

[32] A qual já se viu, é o modelo do problema propriamente.

[33] Discutir-se-há ao longo deste trabalho alguns critérios de seleção.

[34] Também chamada de cruzamento ou *crossover*.

[35] Há uma variedade de critérios de exclusão: aptidão, idade, adaptabilidade, etc. Discutir-se-há ao longo deste trabalho alguns critérios de exclusão.

Fica claro, portanto, que o objetivo dos algoritmos genéticos é modificar a população ao longo das gerações [36].

Para que ocorra essa diversidade empregam-se basicamente, como visto, dois **operadores genéticos**: mutação e *crossover*.

Através do operador de mutação altera-se de forma aleatória, ou através de uma heurística [37] um ou mais genes de um conjunto de indivíduos [38].

Através do fator de *crossover* pretende-se perpetuar as características adaptativas dos pais nas gerações futuras. Isso se dá através da transferência genética dos pais para os filhos.

## 3.7 Exploração das possibilidades dos operadores Genéticos

Como visto na seção 3.6, há uma vastidão imensa de possibilidades de operadores genéticos. Nesta subseção, tratou-se de explorar o conjunto mais conhecido dessas possibilidades.

### 3.7.1 As diferentes formas de Seleção

A maioria dos trabalhos ignora a complexidade do processo de seleção. Se, por um lado, busca-se os indivíduos mais aptos de acordo com a função *fitness*, por outro, esta mesma abordagem pode balizar o processo exploratório do espaço de busca de diversidade. Essa força pressionadora da função *fitness* é chamada **pressão seletiva** ou **intensidade de seleção** (LINDEN, 2012).

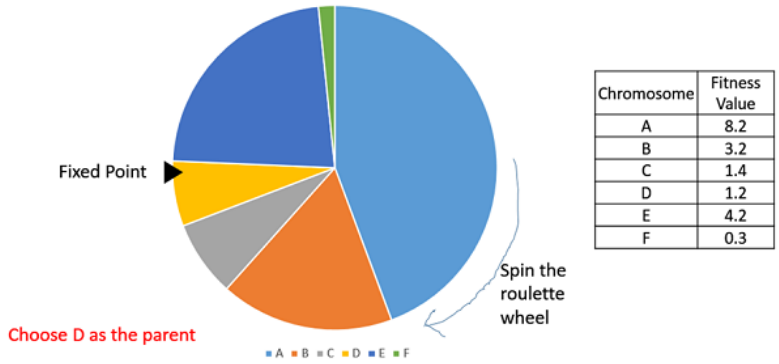
Manter uma boa diversidade é crucial para um valor adequado de convergência gênica, conseqüentemente para o sucesso do algoritimo genético. Uma alta pressão seletiva ocasiona uma convergência prematura, fazendo o algoritimo estacar-se num ponto de máximo ou mínimo local.

A seguir, apresenta-se os principais métodos de seleção.

- **Método da roleta viciada** Considere uma roleta circular. Divide-se esta roleta em $n$ pedaços, onde $n$ é o número de indivíduos no espaço de busca (população). Cada indivíduo tem na roleta uma porção relativa ao valor de sua função *fitness*. **Um ponto fixo** é então escolhido na circunferência e a roleta é girada. A região da roleta que parar na frente do ponto fixo é escolhida como primeiro pai. Repete-se o processo para o segundo e assim sucessivamente.

```
1 DEFINE:
2         EvaluatedIndividual:
```

[36] Observe-se bem aqui que não se trata de **aperfeiçoar**, mas modificar. Dado ser um processo heurístico não se pode necessariamente pensar que as gerações posteriores serão melhores. A idéia todavia, é tentar manter as características de adaptabilidade adquiridas pelas gerações anteriores.
[37] Ou de ambas as formas.
[38] Nada impede deste conjunto ser unitário.




| Chromosome | Fitness Value |
|---|---|
| A | 8.2 |
| B | 3.2 |
| C | 1.4 |
| D | 1.2 |
| E | 4.2 |
| F | 0.3 |

Figura 21 – Método da Roleta Viciada

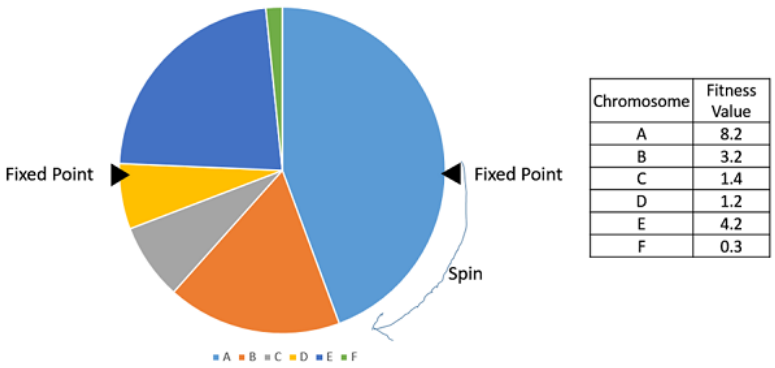
```
3                 individual = Individual
4                 fitnessValue = RealNumber
5   END
6
7   BEGIN:
8   Function RouletteWheelSelection <- (Population):
9           evaluated = EvaluatedIndividual
10          evaluated_population = []
11          sum, aux = 0
12
13          FOR EACH individual IN Population:
14                          evaluated.individual = individual
15                          evaluated.fitnessValue = fitness(individual)
16                          sum = sum + evaluated.fitnessValue
17                          evaluated_population.add(evaluated)
18          END FOR
19
20          limit = random_uniform(0,sum)
21          evaluation_sum = sum
22          limit = Random.uniform(0, sum) * evaluation_sum
23
24          choose = Individual
25
26          FOR EACH ind IN evaluated_population:
27                  aux = aux + ind.fitnessValue
28                  IF aux >= limit:
29                          choose = ind
```

```
30                              STOP FOR
31                      END IF
32              END FOR
33
34              RETURN choose
35
36  END
```

- **Amostragem Estocástica Universal**: é um processo muito parecido com o da roleta viciada, com a única diferença que, ao invés de um único ponto fixo, coloca-se múltiplos pontos fixos. Para se escolher estes pontos, define-se o número $k$ de pais e sorteia-se um número $j \in \{0, \frac{1}{k}\}$. A posição dos pontos é uma série geométrica dada por:

$$a_n = i + \frac{n-1}{n}, n \in \{0, k\}$$

Todos os pais são escolhidos em apenas uma rodada da roleta (ao invés das $k$ vezes que seria no método da roleta viciada). Além disso, cada indivíduo será selecionado um número de vezes muito próximo à verdadeira razão de sua avaliação para a soma de todas as avaliações, evitando assim que superindivíduos [39] surjam assim como elimina rapidamente indivíduos pouco aptos. Isso é um fator geralmente muito indesejado, dado que faz o algoritimo tender a um ponto de máximo ou mínimo local, geralmente tão rápido quanto o método da roleta viciada.




| Chromosome | Fitness Value |
|---|---|
| A | 8.2 |
| B | 3.2 |
| C | 1.4 |
| D | 1.2 |
| E | 4.2 |
| F | 0.3 |

Figura 22 – Método de Amostragem Estocástica Universal

[39] Indivíduos com valores de aptidão extremos.

Os métodos acima são chamados métodos **proporcionais a *fitness*** e não funcionam caso a função fitness possa retornar valores negativos, dado que fazem a seleção considerando este retorno como um espaço, uma porção da roleta [40].

- **Seleção por torneio:** Neste método, seleciona-se, aleatoriamente, os $k$ indivíduos da população que irão competir em um torneio que definirá os pais[41]. Este torneio seleciona os pais através de sua função *fitness*. Dentre os $k$ indivíduos escolhidos para o torneio, aquele que for o mais apto será selecionado. Sua grande vantagem é, pois, acabar com a predominância do indivíduo mais forte da população, dado que os indivíduos mais aptos podem não ser escolhidos. Seja $n$ o tamanho da população $Pop(n)$. Se $k = n$, então ocorrerá que sempre o indivíduo mais apto da população será o pai. Caso contrário, o método seleciona o mais apto em $Pop(n - k)$.

```
BEGIN:
Function TournamentSelection <- (population, k_parents):
        i = 0

        selected = []
        choosed_individuals_index = []

        WHILE i < k_parents DO:
                WHILE j IS IN choose_individuals_index:

                        // j is a random number between 0 and k
                        j = Random.uniform(0, k_parents)

                END WHILE

                // add the individual of j index in selected list
                selected.add(population[j])

                // add the j index in used indexes list
                choosed_individuals_index.add(j)
        END WHILE
```

[40] Exceto, obviamene, se aplicar-se aos valores fitness uma transformação, por exemplo, linear.

[41] Aqui é importante duas características: a primeira delas a se ressaltar é que aleatoriedade propriamente não existe na natureza, nem sequer numa representação matemática. Até mesmo nos computadores a aleatoriedade se dá através de uma regra, geralmente uma distribuição probabilística uniforme. No caso da natureza, a aleatoriedade nem sempre pode ser descrita pelo homem através da linguagem Matemática. Quando se fala em **randômico** tende-se a pensar sempre num valor que "surge do nada". Aqui ressalta-se o fato de que há sempre uma inteligência por trás, principalmente no caso dos computadores onde as funções *Random()* são procedimentos algoritmicos.

```
22
23          fitnessValue = 0
24          father = population[0]
25
26          FOR EACH individual IN selected:
27                  fit = fitness(individual)
28
29                  IF fit > fitnessValue:
30                          fitnessValue = fit
31                          father = individual
32                  END IF
33
34          END FOR
35
36          RETURN father
37  END
```

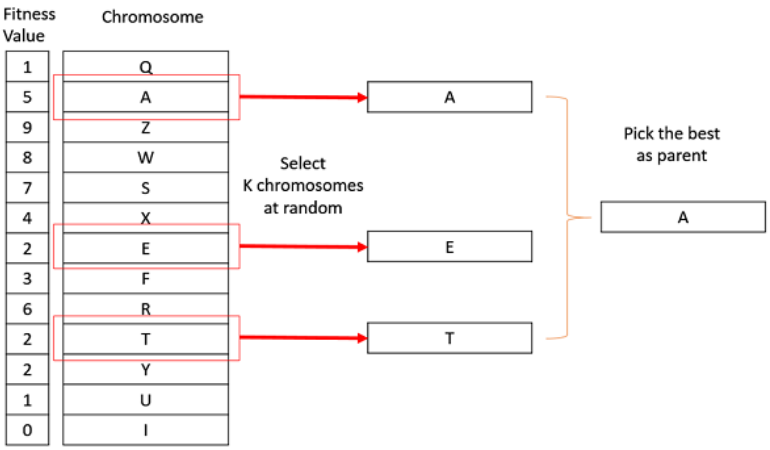



Figura 23 – Seleção por Torneio

Embora boa parte dos autores, tais quais Linden [42], concordem que não há nenhum problema em um indivíduo ser selecionado mais de uma vez, considero pessoalmente uma falha em eficiência. Afinal, isso implicaria no cálculo do mesmo *fitness* mais de uma vez o que pode ser um desperdício, além de diminuir a diversidade.

Em termos práticos para o algoritmo, todavia, o mesmo indivíduo ser selecionado várias vezes não implica em nada substancial.

[42] (LINDEN, 2012), p.205.

A única coisa mais problemática, caso indesejável, é que o algoritmo de certa forma ainda é elitista. Se, por um lado, ele diminui o elitismo impedindo que sempre superindivíduos sejam selecionados para o torneio, por outro, a probabilidade $p(i)$ [43] do indivíduo menos apto participar de um *crossover* ou de uma mutação é de:

$$p(i) = \frac{1}{n^k}$$

Onde $n$ é o tamanho da população e $k$ o número de pais [44].

- **Seleção por *Ranking***: Neste método os indivíduos são avaliados pela função *fitness* e, a seguir, ordenados segundo seus valores de avaliação. Depois, essa posição no *ranking* é definida como sendo a base da seleção, ao invés do valor *fitness*. Este processo é uma espécie de normalização: o *rank* $N$ é dado ao indivíduo mais apto e o *rank* 1 ao menos apto. A probabilidade de seleção é, então, linearmente pontuada de acordo com o seu peso.

  Seja $\nu$ o valor de *fitness* de um indivíduo na população e os símbolos de "$-$" e "$+$" indiquem, respectivamente, o indivíduo menos e o mais apto, tem-se:

$$p_i = \frac{1}{N}\left[\nu^- + (\nu^+ - \nu^-)(\frac{i-1}{N-1})\right]; i \in \{1, ..., N\} \tag{3.3}$$

  De onde observa-se que $\frac{\nu^-}{N}$ é a probabilidade de seleção do pior indivíduo e $\frac{\nu^+}{N}$ a do mais apto.

  Obsrva-se também que, mesmo que dois indivíduos tenham o mesmo valor *fitness* ainda sim ele terá um *rank* $i$ diferente; logo, probabilidade $p_i$ diferente de ser selecionado.

  Diferentes formas de se calcular o *rank* foram propostas ao longo dos anos.

  Koza (KOZA, 1992) determina a probabilidade pela multiplicação de um fator $r_m$ que determina o gradiente da função linear, substituindo $\nu^-$ e $\nu^+$ em $p(i)$ pelos seguintes valores gradientes:

$$\nu^- = \frac{2}{r_m + 1}$$

$$\nu^- = \frac{2r_m}{r_m + 1}$$

  Já Whitley (WHITLEY, 1989) propõe uma distribuição uniforme no lugar da distribuição linear:

[43] Este é outro nome para a pressão de seleção.
[44] Também chamado **tamanho do torneio**.

$$p(i) = \left\lfloor \frac{N}{2(c-1)}[c - \sqrt{c^2 - 4(c-1)\zeta}] \right\rfloor, 1 < c \leq 2$$

onde o parâmetro $c$ é chamado "*bias* de seleção".

Se $\nu^+ = c$, o método de Whitley aproxima-se muito da equação 3.3.

Uma forma comum de manter a pressão seletiva num nível elevado é utilizar uma função de mapeamento exponencial:

$$p(i) = f(e^{ki}, N) | p(i) \propto e^{ki}, f : \mathbb{R} \to \mathbb{R}$$

Isso porque o fator exponencial acelera a curva de convergência. Por um lado, o método de *ranking* causa a menor convergência gênica. Por outro, por diminuir a pressão seletiva, aumenta a diversidade.

```
DEFINE:
        EvaluatedIndiviudal:
                individual = Individual
                fitness = fitness(individual)

DEFINE:
        evaluated_population = list of evaluated individuals

DEFINE:
        RankedIndividual:
                individual = EvaluatedIndividual
                rank = RealNumber
BEGIN:
Function RankingSelection <- (population):
        evaluated_population = fitness(population)
        sorted_by_fitness = sort(evaluated_population, fitness)
        N = sorted_by_fitnes.lenght

        min = sorted_by_fitnss[0].individual.fitness
        max = sorted_by_fitnss[N].individual.fitness

        ranked_individual_list = [RankedIndividual]
        rank = 0

        FOR EACH ind IN sorted_by_fitness:
                ri = RankedIndividual
```

```
27                  ri.individual = ind.individual
28                  ri.rank = min + (max-min)*(rank - 1)/(N-1)
29                  ranked_individual_list.add(ri)
30          END FOR
31
32          // Wheel Roulete selection
33          sum = 0
34          evaluation_sum = 0
35
36          FOR EACH ind IN ranked_individual_list:
37                  evaluation_sum = evaluation_sum + ind.rank
38          END FOR
39
40          limit = Random.uniform(min, max) * evaluation_sum
41
42          aux = 0
43          choose = Individual
44
45          FOR EACH ind IN ranked_individual_list:
46                  aux = aux + ind.rank
47                  IF aux >= limit:
48                          choose = ind
49                          STOP FOR
50                  END IF
51          END FOR
52
53          RETURN choose
54
55  END
```

Neste pseudo-código utilizou-se forma mais simples de *ranking*. Percebe-se nele que a complexidade deste tipo de seleção é de $O(n.log(n))$ operações devido a exigir um processo de ordenação que é computacionalmente caro. Outro fator interessante é que foi utilizado, a título de demonstração, a seleção complementar por roleta viciada substituindo o valor *fitness* pelo *rank*. Outros modelos de seleção poderiam ser muito bem aplicados aqui. Numa implementação real, tanto a lista com valores *fitness* quanto a lista com *rank* podem ser a mesma, dado que seus elementos são apenas **indivíduo** e **um valor real**.

### 3.7.2 As diferentes formas de Recombinação

Na Biologia, existem dois processos pelos quais as células se dividem: mitose e meiose [45]. A diferenciação mais fundamental destes dois processos se dá que, no primeiro, as células filhas são idênticas às células mãe e são, pois responsáveis pela sua substituição ou seu aumento, enquanto que o segundo (a meiose) cria gametas [46], gerados pela recombinação das células do pai e da mãe, sendo portanto diferente das primeiras sendo ela, muito mais até do que as mutações, as grandes responsáveis pela variabilidade genética nos seres vivos [47]. Este processo só é possível através da chamada **reprodução sexuada**. O *crossing-over* é um fenômeno composto pela quebra e troca de partes de cromátides [48] homólogas. A operação de *crossover* (recombinação) é uma abstração da recombinação biológica: essencialmente trata-se da divisão do cromossomo segundo uma regra matemática e sua posterior combinação gerando cromossomos filhos. Dentre as formas comumente utilizadas de *crossover* encontram-se:

1. **Crossover em um ponto**: esta é a forma mais trivial de crossover e trata-se de dividir dois cromossomos em uma posição sorteada $K$, trocando a primeira parte de um com a segunda do outro.

$$cr_{pai}(x_1, x_2, x_3, ..., x_N), N\%2 = 0, \{N, K\} \in \mathbb{N}$$

$$cr_{mãe}(y_1, y_2, y_3, ..., y_N), N\%2 = 0, \{N, K\} \in \mathbb{N}$$

$$cr_{filho}(x_1, x_2, ..., x_K, y_{k+1}, ..., y_N), N\%2 = 0, n \in \mathbb{N}$$

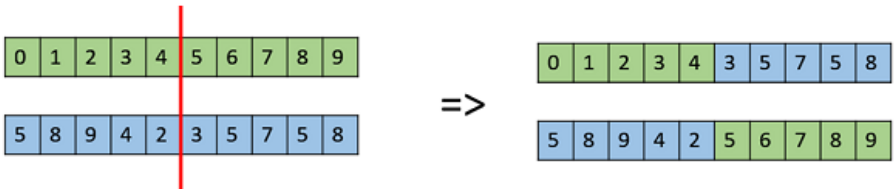



Figura 24 – Crossover de um ponto

[45] (AL., 2002), especialmente nos capítulos 18 a 20.
[46] Células sexuais.
[47] O que é uma simples, todavia, grande evidência contra a evolução darwinista, conforme explicado no Apêndice D.
[48] Um dos filamentos interligados, formados pela duplicação de um cromossomo durante os processos de divisão celular.

2. **Crossover em múltiplos pontos**: É uma generalização do esquema de *crossover* de um ponto. Aqui a única diferença é que ao invés de se selecionar um único ponto $K$ no qual haverá a troca das partes das "cromátides", sorteia-se múltiplos pontos fazendo-se a troca de forma alternada.

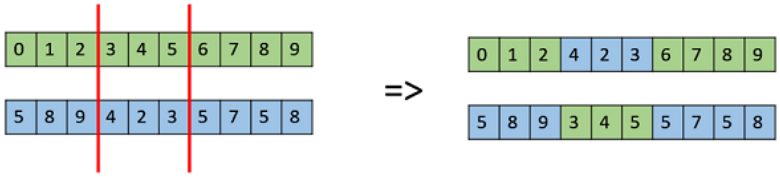



Figura 25 – Crossover de múltiplos pontos

3. **Crossover uniforme**: Nesta forma de recombinação, não se divide o cromossomo em múltiplas partes, mas trata-se cada gene separadamente. Isso permite recombinar todos os padrões de cromossomos. O processo basicamente trata-se de lançar uma moeda para cada gene específico: se o resultado for cara, pega-se o gene do pai; se for coroa, pega-se o gene da mãe.

$$coin = random(0, 1)$$

Onde $random(x, y)$ é uma distribuição probabilística randômica uniforme que seleciona um número natural entre 0 e 1.

$$cr_{pai}(x_1, x_2, x_3, ..., x_N), N\%2 = 0, \{N, K\} \in \mathbb{N}$$

$$cr_{mãe}(y_1, y_2, y_3, ..., y_N), N\%2 = 1, \{N, K\} \in \mathbb{N}$$

$$cr_{filho} = \{z_i\}_1{}^N, z_i = \begin{cases} z_i = x_i, coin = 0 \\ z_i = y_i, coin = 1 \end{cases}$$

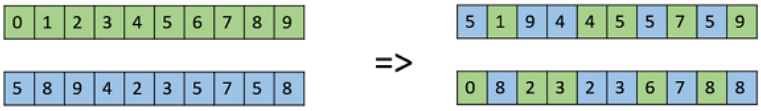



Figura 26 – Crossover uniforme

4. **Crossover aritimético**: Trata-se de utilizar uma combinação linear para compor os cromossomos. Geralmente a equação utilizada compõe uma fórmula complementar como a que se apresenta a seguir, mas isso não é mandatório.

$$cr_{pai}(x_1, x_2, x_3, ..., x_N), N\%2 = 0, \{N, K\} \in \mathbb{N}$$

$$cr_{mãe}(y_1, y_2, y_3, ..., y_N), N\%2 = 0, \{N, K\} \in \mathbb{N}$$

$$cr_{filho} = \{z_i\}_1{}^N, z_i = \alpha x + (1 - \alpha)y$$

5. **Crossover de múltiplos pais**: matematicamente, à exceção da recombinação de um ponto, todas as demais podem seguir uma regra tendo múltiplos pais onde a escolha dos pais pode se dar por um fator $p(\vec{X})$ [49] que determina o índice do vetor de pais $\vec{\zeta}$ a ser explorado e executa uma regra de composição $R(\vec{\zeta})$ que pode ser qualquer uma das acima mencionadas ou outras quaisquer:

$$\vec{\zeta}_{MN} = \begin{pmatrix} x_{11}, x_{12}, x_{13}, ..., x_{1N} \\ x_{21}, x_{22}, x_{23}, ..., x_{2N} \\ ..., ..., ..., ..., ... \\ x_{M1}, x_{M2}, x_{M3}, ..., x_{MN} \end{pmatrix}$$

$$z_{ij} = p(\vec{\zeta}_{MN})$$

Onde $z_{ij}$ é um gene retornado do vetor de pais $\vec{\zeta}_{MN}$.

$$\rho_i = R \rightarrow \{1, N\} | \{(z_{ij})_k\}_1^N$$

Onde $R$ é uma relação matemática que define a regra de escolha dos pais até que se forme um vetor filho $cr_{filho} = (\rho_1, \rho_2, ..., \rho_N)$.

Pode-se ver, por esta última classificação, que existem infinitas possibilidades de recombinação gênica, sendo que algumas mais podem ser vistas na vasta bibliografia apontada por Linden em seu livro (LINDEN, 2012). **Cabe ao projetista escolher** qual irá aplicar e verificar se foi satisfatória para o caso em questão.

[49] Que tende a ser uma probabilidade dentro de uma distribuição uniforme, embora possa ser qualquer função.

### 3.7.3 As diferentes formas de mutação

De uma forma bem simples, a mutação é uma pequena mudança [50] em um cromossomo e que gera mudanças no mesmo indivíduo [51]. Assim como no operador genético de *crossover*, também as mutações podem ser de diversas formas. Nesta subseção se apresentará algumas delas.

- **Mutação por troca de *bit***: Este tipo de mutação é o mais simples e se dá pela simples troca de um ou mais *bits* selecionados aleatoriamente. É mais comumente aplicado a representações binárias.
- ***Random Reseting***: é uma variação da troca de *bit* para números inteiros ou reais e se trata de substituir o valor antigo de um ou mais genes por um valor do conjunto de valores permitidos.
- **Mutação por troca**: neste tipo de mutação seleciona-se um ou mais pares de *bits* e trocam seus valores nos cromossomos.
- **Mutação *Scramble***: Nesse tipo de mutação um conjunto de genes de um cromossomo é embaralhado ou permutado.
- **Mutação por inversão**: Neste tipo de mutação, seleciona-se um sumbconjunto de genes de um cromossomo e inverte-se.

Todos estes métodos enquadram-se dentro da chamada **mutação aleatória**, pois sejam os *bits* do cromossomo, sejam valores ou sequências de genes, todos são escolhidos aleatoriamente.

Outras formas mais complexas de mutação podem ser utilizadas. A seguir, apresentar-se-há duas das mais utilizadas.

- **Mutação Direcional Adaptativa**[52]: este tipo de mutação, proposta por Tang (TANG; TSENG, 2013), busca evitar a concentração da população em um ótimo local, causado pelo *crossover*; e a busca não sistemática da mutação aleatória. Para atingir estes objetivos, o novo ponto de busca é dirigido pela própria solução, baseado na aptidão das três últimas gerações. Gera-se, também, uma probabilidade de mutação $p_m(x)$ que aumenta a chance de cromossomos $x$ menos aptos passarem pela mutação. Seja $\Psi$ a função *fitness*, $x(t)$ o cromossomo na

[50] Em tese aleatória (é o que mais se aceita); ou de alguma forma obscura determinada pelo meio segundo o darwinismo.
[51] A teoria darwinista sugere que as sucessivas mutações ao longo das gerações produzirão indivíduos cada vez mais diferentes até que, em algum momento, este processo adaptativo, juntamente com a recombinação, produzirá uma nova espécie: especialização. As discussões a respeito disso podem ser vistas no Apêndice D, para o qual este capítulo é de suma importância.
[52] **ADM**: *Adaptativa Directed Mutation*

geração $t$ [53], $x_k(t)$, um gene do cromossomo $x$ na geração $t$, $\Delta x_k(t)$ a variação do gene do indivíduo na geração $t$ e $\Delta x(t)$ o indivíduo resultante da varição dos genes $x_k(t)$, tem-se:

$$\Delta\Psi(t) = \Psi[x(t)] - \Psi[x(t-1)] \tag{3.4}$$

$$\Delta\Psi(t-1) = \Psi[(x(t-1)] - \Psi[x(t-2)] \tag{3.5}$$

$$\Delta x_k(t) = x_k(t) - x_k(t-1) \tag{3.6}$$

$$\Delta x_k(t-1) = x_k(t-1) - x_k(t-2) \tag{3.7}$$

$$\Delta x(t) = \forall x_k(t) \in x(t) | x_k(t) = \Delta x_k(t) \tag{3.8}$$

A probabilidade de mutação é então calculada segundo a regra:

$$p_m = \begin{cases} \frac{1}{2}\left[\frac{\Psi_{max}(t)-\Psi(x(t))}{\Psi_{max}(t)-\bar{\Psi}(t)}\right] & \text{se } \Psi(x(t)) \geq \bar{\Psi}(t); \\ \frac{1}{2} & \text{se } \Psi(x(t)) < \bar{\Psi}(t). \end{cases} \tag{3.9}$$

Onde $\Psi(x(t))$ é a aptidão do indivíduo $x$ na geração $t$; $\bar{\Psi}(t)$ é a média de aptidão da população na geração $t$ e, $\Psi_{max}(t)$ é a maior aptidão da população na geração $t$.

Utilizando-se esta probabilidade há várias formas de se fazer a mutação. Aqui, apresenta-se a mutação direcional de Tang (TANG; TSENG, 2013), usando-se as mesmas nomenclaturas utilizadas até agora.

$$x_k(t+1) = x_k(t) + sign(\Delta\bar{\Psi}(t))\Delta x_k(t) p_m \tag{3.10}$$

$$sign(\alpha) = \begin{cases} -1 & \text{se } \alpha < 0 \\ +1 & \text{se } \alpha > 0 \\ 0 & \text{se } \alpha = 0 \end{cases}$$

Mutação Uniforme: o operador de mutação uniforme garante que a mutação tenha um comportamento exploratório no início do processo e um comportamento de ajuste fino ao seu fim através de um sorteio de um valor $\tau$ binário e determina o valor da mutação através da expressão 3.11 (LINDEN, 2012).

$$x'_i = \begin{cases} x_i + \Delta(t, \nu_i^+ - x_i) & \text{se } \tau = 0 \\ x_i - \Delta(t, \nu_i^- - x_i) & \text{se } \tau = 1 \end{cases} \tag{3.11}$$

[53] Sendo $(t-1), (t-2)$ as gerações anteriores.

Onde $\nu_i^+$ e $\nu_i^-$ são, respectivamente, os valores máximo e mínimo admitidos para o gene; $x_i$, o valor do gene sofrendo a mutação; $t$, a geração corrente; e $x'_i$ o gene mutado. Por sua vez, o valor de $\Delta$ é calculado pela expressão 3.12.

$$\Delta(t, y) = y \left[1 - \rho^{(1-\frac{t}{g_{max}})^{\beta}}\right] \tag{3.12}$$

Onde:

- $t$ é a geração corrente;
- $y$ é o valor máximo da mutação;
- $g_{max}$ é o número máximo de gerações;
- $\rho \in \{0, 1\}$
- $\beta$ é um parâmetro que controla o quanto a mutação é dependente do número de gerações.

Mais operadores podem ser introduzidos na programação de algoritmos evolutivos. Um deles, apresentado por Linden [54] é o operador de *Inver-Over*. Este operador substitui os operadores de *crossover* e de mutação por um processo seletivo arrojado que é executado sobre todos os indivíduos da geração anterior. Possui um custo computacional claro e, durante esta pesquisa bibliográfica, não foram encontrados casos reais de sua aplicação.

## 3.8 Múltiplos objetivos e Restrições

Em diversos casos, os problemas propostos não possuem apenas um objetivo, mas um conjunto de objetivos que podem, inclusive, competir entre si e que, na maioria das vezes ainda são restringidos por condições que os limitam.

O problema matemático de otimização multiobjetivo (**POM**) consiste na obtenção de um conjunto de variáveis (conjnto solução) que satisfaça algumas restrições e otimize uma função constituída por diversos termos ou funções objetivos.

[54] (LINDEN, 2012), p. 254.

Considere $p$ o número de funções objetivo. Seja $A \models B$ a representação para indicar que $A$ está **sujeito a** $B$, sendo $A$ e $B$ conjuntos de funções, pode-se descrever um POM em linguagem matemática da seguinte forma:

$$\begin{aligned} & z(x) = \{f_1(x), f_2(x), ..., f_r(x)\} \\ & \Phi x \in X^* \\ & X^* = \{\forall x \in X | g(x) \leq b\}, b \in \mathbb{R}^p \\ & \lfloor z \rfloor \models \Phi \end{aligned} \tag{3.13}$$

onde, $p$ é o número de funções objetivo, $g : \mathbb{R}^n \to \mathbb{R}$ é uma função de restrição; $z : \mathbb{R}^n \to \mathbb{R}^p$, a função multiobjetivo coposta por $r$ funções objetivo $f_r : \mathbb{R}^n \to \mathbb{R}$. O **conjunto viável** [55] do problema será, portanto, definido como $X^*$ e $X$ o **espaço objetivo**. Desta forma, é plausível notar por $Z$ o conjunto imagem e $Z^*$ o conjunto imagem de soluções viáveis, ou **espaço objetivo factível**.

$$Z^* = \{(f_1(x), f_2(x), ..., f_r(x)), x \in X^*\} \tag{3.14}$$

O problema de otimização consiste, pois (como descrito na equação 3.13), em encontrar o valor mínimo de $z(x)$ que esteja sujeito às regras $\Phi$.

Desta forma, segundo Eiben (EIBEN; SMITH, 2015) pode-se muito bem definir um problema de satisfação de restrições como sendo um conjunto $< X, \Psi(x) >$, onde $\Psi(x) : X^* \to \{Verdadeiro, Falso\}$ - uma função booleana em $X$ que mapeia as soluções pertencentes a $\Phi(x) = \{x_1, x_2, ..., x_r\}$ que respeitem a regra $g(x) \leq b$. $\Phi(x)$ é chamado **vetor de decisão**, $r$ o número de objetivos.

Assim:

$$\Psi(x) = \begin{cases} \text{Verdadeiro} & X^* \text{ é uma região } \textbf{admissível}. \\ \text{Falso} & X^* \text{ é uma região } \textbf{inadmissível}. \end{cases} \tag{3.15}$$

Enquanto na otimizaçao mono-objetivo uma solução ótima é claramente identificada, dada a ordenação do espaço de busca, para Fleming (FONSECA; FLEMING, 1993), existem um conjunto de soluções alternativas que também podem ser consideradas eficientes e compor o conjunto de soluções eficientes, ou conjunto admissível do problema (PANTUZA, 2011);

[55] Também chamado de conjunto de soluções factíveis.

### 3.8.1 O conceito de Pareto

Um dos problemas pioneiros das ciências econômicas foi a busca de um critério que definisse uma alocação eficiente de recursos sendo assim, um conceito muito anterior aos computadores transistorados e aos algoritmos genéticos (PARETO, 1971 (1906)).

Originalmente, o conceito do ótimo de Pareto, fora desenvolvido pelo italiano Vilfredo Pareto e define um estado de alocação de recursos em que seria impossível relocá-los de tal forma que a situação de qualquer um dos participantes sejam melhorada sem que a situação de outro indivíduo participante seja piorada (BARR, 2004).

Aplicado à computação, o conceito do ótimo de Pareto implica na alocação dos recursos computacionais da forma mais entrópica possível.

Assim sendo, o conceito de **pareto-ótimo**[56] é um problema de otimização. Deve-se, portanto, buscar um ótimo entrópico num conjunto de soluções. Esta característica entrópica dos problemas nos quais se aplicam o ótimo de Pareto se dá justamente pelo equilíbrio que todos os elementos do conjunto (os indivíduos no sistema econômico originário) devem atingir de forma a manifestarem o melhor de si sem prejudicar os demais indivíduos. É, portanto, uma busca por este ponto de equilíbrio.

Na linguagem Matemática, pode-se pressupor inicialmente que cada indivíduo $z_k$ de conjunto de indivíduos $\vec{z} = \{z_1, z_2, ..., z_N\}$ deverá atingir um ponto de máximo ou mínimo numa dada função *fitness* que não cause a deteriorização dos demais indivíduos do conjunto.

Pela definição, um vetor $\vec{z}$ qualquer é Pareto-ótimo se não existe outro vetor admissível $\vec{z^*}$ que possa melhorar algum objetivo sem, todavia, causar uma piora em pelo menos outro objetivo. Em outras palavras, um vetor solução $z$ pertence ao conjutno de soluções Pareto-ótimas se não existe nenhum $\vec{z^*}$ que domine $\vec{z}$ [57].

Assim sendo, sendo $\vec{A} \succ \vec{B}$ usado para expressar que o vetor $\vec{A}$ domina o vetor $\vec{B}$, tem-se:

$$\vec{z} \succ \vec{z^*} \Leftrightarrow \forall i \in \{0, ..., \|z^*\|\} \begin{cases} f_i(\vec{z}) \leq f_i(\vec{z^*}) \wedge \exists j \in \{0, ..., \|z^*\|\} | f_j(\vec{z}) < f_j(\vec{z^*}) & \text{(MIN)} \\ f_i(\vec{z}) \geq f_i(\vec{z^*}) \wedge \exists j \in \{0, ..., \|z^*\|\} | f_j(\vec{z}) > f_j(\vec{z^*}) & \text{(MAX)} \end{cases} \quad (3.16)$$

Onde a primeira expressão apresenta um caso de minimização, e a segunda um caso de maximização.

[56] Ou ótimo de Pareto.

[57] Esta solução apresentada por Guido (PANTUZA, 2011) (p.27) é mais correta que a apresentada por Linden (LINDEN, 2012) (p. 369), dado que este restring o conceito de dominância apenas à maximização.

O conjunto formado por todos os vetores $\vec{z^*}$ que são dominados por $\vec{z}$ recebe o nome de **conjunto Pareto**.

### 3.8.2 Método da Pena de Morte

A primeira solução e, geralmente a mais óbvia, é a eliminação dos indivíduos que não satisfaçam às restrições e não possam ser admissíveis. Esta solução pode não ser viável quando o espaço admissível for muito pequeno em relação ao espaço de busca $X$. Neste caso, a população inicial poderia ser toda de indivíduos inviáveis o que impossibilitaria a evolução do algoritmo genético. Além deste, há outros dois motivos a serem considerados: o primeiro é que durante o processo evolutivo, o algoritmo genético pode gerar indivíduos inadimissíveis a todo momento, o que significa um esforço desnecessário, pois será em seguida testado e eliminado; o segundo é que indivíduos muito próximos de serem admissíveis e que possuem genes interessntes para o problema, sejam eliminados; desconsiderar tais genes durante o processo evolucionário pode prejudicar na busca pela melhor solução.

### 3.8.3 Penalização dos cromossomos inadequados

Um outro método é associação de um peso $\kappa(x)$ a cada indivíduo considerado inadmissível [58] aumentando (ou diminuindo, no caso de maximização) seu valor *fitness* $f(x)$ proporcionalmente ao tanto que afasta destes critérios.

$$\begin{aligned} &f_{penalizado}(x) = f(x) \pm \kappa(x) \\ &\varphi(x) = \{f_{penalizado}(x)_i\}, i = 1, 2, 3, ..., r \\ &r = |X^*| \\ &x_{penalizado} = \{x \in < X, \varphi(x) >\} \end{aligned} \tag{3.17}$$

### 3.8.4 Método da soma ponderada

Este método é uma variante do método de penalização, mas consiste em transformar um método multiobjetivo em um método mono-objetivo por uma transformada linear.

$$\begin{aligned} &f(x) = \textstyle\sum_{i=1}^{r} \kappa_i(x) f_i(x) \\ &r = |X^*| \\ &f(x) \models \{x \in X^*\} \end{aligned} \tag{3.18}$$

[58] Motivo pelo qual o método recebe em inglês o nome de *Weight-based Genetic Algorithm* (Algoritmo Genético baseado em peso), WBGA.

Se $W = \{w_1, w_2, ..., w_r\}$ for um vetor de pesos, uma solução $x \in X^*$ será ótima se:

$$x \text{ for único} \wedge w_i > 0, i = \{1, ..., r \tag{3.19}$$

Para que os pesos $w_i$ reflitam aproximadamente a importância dos objetivos, o decisor deve normatizá-los (PANTUZA, 2011). Além disso, segundo Arroyo (ARROYO, 2002), a principal desvantagem deste método consiste em não gerar todas as soluções Pareto-ótimas quando o espaço de busca é não convexo.

### 3.8.5 Pareto com $\varepsilon-$restrito

O vetor correspondente ao conjunto Pareto que é composto por todos os vetores de decisão ótimos é chamado ***front* Pareto** (YILMAZ; DURMUŞOĞLU, 2018). Seja $\varepsilon$ um valor arbitrário:

$$\begin{aligned} &d(z_A, z_B) = \sqrt{\sum (z_{Ai} - z_{Bi})^2}, i = 1, 2, ..., r \\ &\varphi \supseteq \Phi | \forall \{z_A, z_B\} \in \varphi, d(\vec{z_A}, \vec{z_B}) \leq \varepsilon \end{aligned} \tag{3.20}$$

### 3.8.6 A questão de dominância múltipla

Quando tem-se $n$ objetivos, pode-se ter mais do que $n$ elementos dominados pelo mesmo número de elementos. Neste caso, pode-se acrescentar o conceito de *crowding* que mede o quão apertado está o espaço em torno de um indivíduo (LINDEN, 2012).

# 4 Desenvolvimento do caso de Estudo

Neste capítulo, apresenta-se o desenvolvimento do caso de estudo específico: **encontrar as menores dimensões ($W$ e $L$) de uma porta NAND com M entradas em tecnologia CMOS, sem utilizar o Cálculo, mas através de Algorítimos Genéticos, de forma a encontrar um circuito de boa performance**.

## 4.1 Organização do trabalho

O presente trabalho se organizou da seguinte forma:

- **A questão inicial:** Numa primeira etapa, procurou-se um problema real enfrentado pelos projetistas a fim de que a otimização não gerasse apenas frutos acadêmicos como também práticos no projeto de hardwares digitais. Encontrou-se então o problema de se construir portas lógicas de M entradas sem prejudicar a performance e com a menor área possível.
- **A Exploração do tema:** Encontrado este problema procurou-se equacioná-lo a fim de encontrar os possíveis métodos de solução. Considerou-se utilizar métodos matemáticos tradicionais, oriundos do Cálculo (derivadas parciais e equações diferenciais). Todavia, a característica do problema sugeriu que os métodos evolucionários fariam mais sentido. A partir de então, buscou-se nas bibliografias qual destes métodos evolucionários conceitualmente seria de mais fácil implementação, traria conhecimentos mais basilares para o pesquisador e os leitores deste trabalho e traria discussões mais complexas que a simples implementação de um algorítimo para resolver um determinado problema.
- **A problemática:** Percebeu-se que os projetistas enfrentam sempre um "problema da gangorra" [1] ao se tentar projetar circuitos digitais com múltiplas entradas. Além dos cálculos serem complexos, sempre provocam ou perda de performance ou aumento de potência, ou aumento do tamanho do circuito (o que o torna muito mais caro), etc. Encontrar o ponto de equilíbrio é uma tarefa feita na base da tentativa e erro.
- **Construção do modelo de análise:** Partiu-se, então, para a solução do problema em si: a implementação de um algoritmo genético que oferecesse valores aceitáveis e que otimizasse o processo construtivo de portas NAND com M entradas de forma a obter-se um circuito com boa performance, menor área, menor consumo de potência.

[1] Onde uma solução gera uma outra dificuldade.

- Deste passo surgiu a construção de um *framework* básico para a solução deste tipo de problema, onde os indivíduos são modelados em função dos valores de $W$ e $L$ do circuito inversor e não de adaptações com valores binários. Tal framework possui uma estrutura orientada a objetos e seu valor é entregue através de processos de *Continuous Integration*, *Continuous Deployment* e *Continuous Delivery*, sendo todo esse projeto compartilhado em repositório público na internet onde qualquer um possa fazer um *fork* submeter um *Pull-Request* e melhorá-lo. Isso nos garante as melhores práticas de desenvolvimento de software, integradas num *framework* simples com fins majoritariamente acadêmicos.

- **Coleta de dados e análise das iformações:** Como parte final deste projeto, fizeram-se os testes simulando um circuito com os melhores resultados obtidos.

- **Conclusões:** Por fim, são apresentadas as conclusões deste trabalho sobre a possibilidade de se usar algoritmos genéticos para se resolver problemas desta natureza, quais suas vantagens e desvantagens, quais as dificuldades e quais seriam os próximos passos.

### 4.1.1 Metodologia

Este trabalho enquadra algumas das principais metodologias de pesquisa. Quanto à abordagem, o presente trabalho se enquadra como pesquisa qualitativa. Ele pretende mostrar as características de tais algoritmos e sua aplicabilidade para resolver problemas de otimização de hardware. De certa forma, o próprio algoritmo genético é substancialmente quantitativo, dado que avalia uma quantidade $Q$ de possiveis soluções num espaço de busca. Todavia, crê-se que isso se caracteriza neste trabalho muito mais como seus objetivos (pesquisa exploratória) do que propriamente sua abordagem. Quanto à natureza é, concomitantemente, uma pesquisa básica dado ser um estudo do estado da arte; mas também aplicada, dado este caso de estudo específico e o *framework* gerado. Quanto aos objetivos, não é apenas uma pesquisa exploratória [2], como também o se trata principalmente de uma pesquisa descritiva (uma vez que é fundamentalmente um estado da arte) e explicativa, dado pretender elucidar não apenas as nuances de um projeto de circuito em tecnologia CMOS como a implementação de um algoritmo genético para se resolve um problema de otimização. Quanto aos procedimentos é bibliográfica e documental fundamentalmente (por se tratar de um estado da arte), e experimental no caso específico implementado e no *frameowrk* produzido.

### 4.1.2 Equacionamento do problema

Primeiramente, considerou-se que todos os transistores do mesmo tipo (N ou P) são do mesmo tamanho. Esta é uma consideração muito importante, pois reduz absurdamente a quantidade

[2] O que o é devido primeiramente à natureza exploratória dos AGs, como também e principalmente por explorar, ainda que de forma explicativa, as várias opções de metodologias de solução possíveis para se solucionar este tipo de problema.

de conta a se fazer e, consequentemente, o custo computacional de nosso algorítimo.

Chamemos de $\epsilon$ [3] ao fator de performance de tempo:

$$\epsilon = |t_{pHL} + t_{pLH}| \tag{4.1}$$

Poderia-se pensar que este fator deveria ser $tp_{HL} - tp_{LH}$. No entanto, se assim o fosse, garantiria-se apenas que a diferença entre o tempo de subida e o de descida seria a menor possível. Ao somar, todavia, e fazer tal soma tender a zero, como será feito a seguir, além de garantirmos estes mesmos fatores, garantimos que ambos tempos de resposta são o menor possível.

Sabe-se pela equação 2.1 que o tempo de propagação é metade, portanto, do fator de performance de tempo. Aqui desconsidera-se essa metade pois interessa-nos apenas as diferenças uma vez que busca-se atingir o valor ideal:

$$\epsilon_{ideal} = |t_{pHL} + t_{pLH}| = 0 \tag{4.2}$$

Da equação 4.2 verifica-se que, uma vez que também [4] se deseja $t_{pHL} = t_{pLH}$ , para portas NAND de M entradas, para se ter um desempenho transiente próximo ao ideal as dimensões devem respeitar a seguinte condição [5] (SODINI, 2017):

$$\left(\frac{W}{L}\right)_N = \frac{M}{2}\left(\frac{W}{L}\right)_P \tag{4.3}$$

Donde, primeiramente, deriva a que chamamos **Condição de Performance Transiente**, denotada por $\phi$:

$$\phi = \left|\left(\frac{W}{L}\right)_N - \frac{M}{2}\left(\frac{W}{L}\right)_P\right| \tag{4.4}$$

Onde, no caso ideal:

$$\phi_{ideal} = \left|\left(\frac{W}{L}\right)_N - \frac{M}{2}\left(\frac{W}{L}\right)_P\right| = 0 \tag{4.5}$$

[3] Nomeado *Time Equality Performance Factor*.
[4] Curvas de transferência simétricas.
[5] Transient Performance Condition

Em segundo lugar, aplicando-se a equação 4.3 às equações 2.3, 2.5, 2.6, 2.7, 2.9 e 2.10, encontra-se a capacitância total que pode ser expressa pela equação 4.6 - a qual considera já os valores comuns convencionais descritos no capítulo 2:

$$C_L = C_{OX_P}(M.W.L)_P + C_{OX_N}(M.W.L)_N + C_J L_{diff}(W_N + W_P) + C_{JSW}(W_N + W_P + 4L_{diff}) \quad (4.6)$$

Adotaram-se, pois, a título de exemplo, os seguintes valores [6]para este projeto:

- $C_{OX_N} = C_{OX_P} = C_{OX} = 6fF/\mu m^2$
- $\mu_N C_{OX} = 115\mu$
- $\mu_P C_{OX} = 30\mu$
- $V_{THN} = 0,4V$
- $V_{THP} = -V_{THN}$
- $V_{DD} = 2,5V$
- As trilhas ocuparão 20% da área ocupada pelo circuito [7].

E, de acordo com o capítulo 2:

- $C_J = 0,2fF/\mu m^2$
- $C_{JSW} = 0,5fF/\mu m^2$

### 4.1.3 Cálculo do *score* da função de avaliação

A fim de se manter uma uniformidade com os padrões de tecnologia do Mercado, escreveu-se a equação 4.6 em função de uma margem referente à esta tecnologia e o $L_{diff}$ proporcional às distâncias do canal. Definiu-se também valores máximos e mínimos relativos às tecnologias existentes no Mercado.

- Valor Máximo de $W_N = 0,36\mu$
- Valor Máximo de $L_N = 0,24\mu$

[6] Condições físicas.
[7] O que é um critério adotado com base na experiência do projetista (empírico). Um tratamento melhor para este ponto específico pode gerar grande impacto.

- Valor Mínimo de $W_N = 32n$
- Valor Mínimo de $L_N = 11n$
- Valor Máximo de $W_P = 0,72\mu$
- Valor Máximo de $L_N = 0,48\mu$
- Valor Mínimo de $W_N = 64n$
- Valor Mínimo de $L_N = 22n$

E a equação 4.6 foi reescrita como:

$$C_L = C_{OX}M(W_P L_p + W_N L_N) + C_J L_{diff}(W_N + W_P) + C_{JSW}(W_N + W_P + 4L_{diff}) \quad (4.7)$$

Como deseja-se que tanto $\epsilon$ quanto $\phi$ sejam idealmente nulos, criou-se o parâmetro algoritmico $y$, dado por:

$$y = |\epsilon + \phi|$$

Deseja-se também que a área seja a menor possível. Como todos os transistores têm o mesmo tamanho, a área será a menor possível quando os transistores forem os menores possíveis.

$$S = (W_N L_N + W_P L_P)(1.2)$$

Onde 20% é considerado um acréscimo justo para dimensionar a área das trilhas.

Calcula-se o *score* da função de avaliação da seguinte maneira:

$$score = y + S(10^{12})$$

Onde, $10^{12}$ é um fator de correção para que a área (certamente muito pequena se comparada a $y$) não seja descartada durante os cálculos.

### 4.1.4 Aplicação do Método do Cálculo Aritimético do Ponto do Crossover

No mundo dos Algorítimos Genéticos, usa-se mais comumente uma distribuição probabilística constante [8] como no Método da Roleta Viciada 3.7.1. Entretanto, como se viu na subseção

[8] De forma que a soma das probabilidades implique no inteiro (100%).

3.7.1, um modelo mais real considera que não se conhece ou não há uma probabilidade adequada para o processo durante toda a execução do algorítimo.

Diversas técnicas de probabilidades variáveis foram propostas ao longo do tempo, em sua maioria visando um decrescimento da probabilidade ao longo das gerações (Veja Subseção3.7.1).

Neste trabalho, utiliza-se, além desta probabilidade variável, uma função aleatória para o cálculo do ponto de cruzamento (Veja Subseção3.7.2).

Isso se dá baseado no fato de que, na Biologia, os dados genéticos provenientes do pai e da mãe nem sempre são utilizados em sua completude, mas com uma certa inteligência se combinam em novos dados.

Utilizou-se, neste ponto, um modelo aritimético-probabilístico (Subseção 3.7.2) para recombinar os genes dos pais de forma a garantir essa aproximação com o modelo biológico.

Seja $\alpha$ o fator heurístico [9], tem-se a função:

$$Al(\alpha) = \alpha g_{father} + (1 - \alpha) g_{mother} \tag{4.8}$$

### 4.1.5 Desenvolvimento do Algorítimo

Como todo Algorítimo Genético, também o desenvolvido neste trabalho é composto de parâmetros de limitação [10] e dos seguintes passos formando um ciclo evolutivo:

- Inicialização da população.
- Avaliação dos indivíduos da população.
- Seleção dos pais.
- Aplicação dos operadores. Neste caso, *crossover* [11] e mutações.
- Evolução.
- Reavaliação.

Para o *crossover*, optou-se pela técnica do cálculo aritimético do ponto do crossover (Subseção 4.1.4).

[9] Que pode ser compreendido como uma aleatoriedade.
[10] Onde se faz presente a engenhosidade e o conhecimento prévio do projetista.
[11] Ou recombinação.

### 4.1.6 Inicialização da População

Como se viu existem diversas formas de se inicializar a população, porém a inicialização randômica atende bem na maior parte dos problemas. E este caso não é distinto.

Todavia a própria aleatoriedade pode ser distribuida segundo alguma função. Aqui, optou-se pela forma mais fundamental e que geralmente obtém resultados satisfatórios: a distribuição uniforme.

Desta forma, o pseudo-código de inicialização da população pode ser visto a seguir:

```
BEGIN:
Population:
        POPULATION = []

        BEGIN:
        Create_Individual:
                ind = new Individual
                ind.wn = uniform(minWn, maxWn)
                ind.wp = uniform(minWp, maxWp)
                ind.ln = uniform(minLn, maxLn)
                ind.lp = uniform(minLp, maxLp)
        END

        BEGIN
        Generate_Initial_Population <- {population_size}:
                IF population_size is not 0:
                        raise ''Population already exists''
                        STOP HERE
                ENDIF

                FOR i in range 0:population_size:
                        individual = Create_Individual()
                        POPULATION.ADD(individual)
        END
END
```

### 4.1.7 Avaliação dos indivíduos da população

Uma vez que este se trata de um problema multiobjetivo, a melhor opção, para se encontrar valores ótimos, seria aplicar as soluções de Pareto 3.8. Todavia as soluções mais robustas de Pareto envolvem cálculo de distância e *crowding*, o qual comumente é usado apenas como critério de

desempate. De qualquer forma, estas soluções complexas consomem um tempo de ordem $O(n^2)$, dada a necessidade de calcular todas as distâncias de todos os indivíduos entre si, a cada geração. Além disso, estas abordagens, embora gerem uma diversidade muito maior, são incapazes de lidar com questões relativas aos *trade-offs*, que geralmente são deixados a encargo do projetista.

Neste caso específico, tentou-se primeiramente utilizar a abordagen mais simples: a soma ponderada (com peso 1 dado que todos nossos objetivos são igualmente importantes) (Subseção 3.8.4), aliada à pena de morte. Com isso, pagou-se o preço de reduzir-se consideravelmente a diversificação que surgiria de não matar os indivíduos menos aptos conforme descrito em 3.8, porém foram encontradas soluções plausíveis para o problema em questão.

Assim, a formulação matemática fica definida a seguir:

**Definição e equações do vetor de *fitness*:**

$$\begin{aligned}
&\vec{x} = \{W_N, L_N, W_P, L_P\} \\
&f_1(\vec{x}) = tp_{HL} = C\frac{Vth_N}{\frac{1}{2}\mu_N C_{ox}\frac{W_N}{L_N}(V_{DD}-Vth_N)^2} \\
&f_2(\vec{x}) = tp_{LH} = C\frac{Vth_P}{\frac{1}{2}\mu_P C_{ox}\frac{W_P}{L_P}(V_{DD}-Vth_P)^2} \\
&f_{31}(\vec{x}) = TPerCond = \left|\frac{W_N}{L_N} - \frac{M}{2}\frac{W_P}{L_P}\right| \\
&f_{32}(\vec{x}) = TEq = \|tp_{HL} - tp_{LH}\| \\
&f_4(\vec{x}) = \|f_{31} + f_{32}\| \\
&f_5(\vec{x}) = (W_N L_N + W_P L_P)(1.2)(10^{12})
\end{aligned} \tag{4.9}$$

Onde, o fator $10^{12}$ aparece para que a área não se torne o fator dominante. Poderia-se algebricamente chamar $(1.2)(10^{12})$ de peso da área algebricamente. Todavia, aqui não se trata propriamente de peso no sentido que se dá nesta abordagem, como se verá na equação 4.11.

Como se deseja minimizar tanto a área quanto a condição de performance e a diferença de tempo, toma-se:

$$f_6(\vec{x}) = \frac{1}{f_4(\vec{x})}, f_7(\vec{x}) = \frac{1}{f_5(\vec{x})} \tag{4.10}$$

Dado que todas as condições são consideradas com mesma importância, toma-se o peso dos vetores igual a 1. Aqui sim fala-se em "'peso"'. Nesta técnica, "'peso"' é a importância de uma função vetorial no vetor de funções *fitness*. Desta forma, o vetor de *fitness* linearizado:

$$\vec{f}(x) = \{f_6(x), f_7(x)\} \tag{4.11}$$

A função limite $g(x)$ é assim definida:

$$g(x) = \begin{cases} 32\eta \leq W_N \leq 0,36\mu \\ 11\eta \leq L_N \leq 0,24\mu \\ 64\eta \leq W_P \leq 0,72\mu \\ 22\eta \leq L_P \leq 0,48\mu \end{cases} \quad (4.12)$$

Então, pode-se por fim, definir-se assim o problema:

$$\vec{x^*} = \{\forall x_i \in \vec{x} : x_i \text{ satisfaz a } g(x)\} \Rightarrow \vec{x} \succ \vec{x^*} \quad (4.13)$$

Onde $\succ$ indica a **dominância de Pareto**, como apresentado na subseção 3.8.1.

A avaliação dos indivíduos é feita pela função de avaliação a qual calcula um *score* [12] a qual pode ser escrita em pseudo-código da seguinte forma:

```
BEGIN:
Evaluation Function <- {individual, M}
        Declare physical conditions
        Declare Ldiff

        # Calculate Expression Load Capacitance

        C = Cox*M*(wp*lp+wn*ln)  + Cj * Ldiff * (wn+wp) + Cjsw * (wn+wp+4*
            Ldiff)

        # Calculate Tphl and Tplh

        tphl = C * Vth_n / ((1 / 2) * unCox * (wn / ln) * (Vdd - Vth_n) **
            2)

        tplh = C * Vth_p / ((1 / 2) * upCox * (wp / lp) * (Vdd - Vth_p) **
            2)

        transientPerformanceCondition = abs((wn / ln) - (M / 2) * (wp / lp)
            )

        timeEqualityPerformance = abs(tphl - tplh)

```

[12] Que foi calculado na subseção 4.1.3.

```
   y = 1/abs(transientPerformanceCondition + timeEqualityPerformance)

        surface = 1/(wn * ln + wp * lp)*(1.2)*(10**(12))

        score = y + surface

        return score

END
```

### 4.1.8 A reprodução dos indivíduos: recombinação genética

A reprodução dos indivíduos pode ser feita de muitas formas como descrito na Subseção 3.7.2.

Primeiramente se calcula o ponto aritimético do *crossover*:

```
BEGIN:
ArithmeticalCrossoverPoint <- (fatherValue, motherValue, alpha):

IF alpha < 0 OR alpha > 1:
        RAISE OverflowError()

RETURN alpha * fatherValue + (1-alpha) * moterValue

END
```

Depois faz-se o cruzamento selecionando $\alpha$ randomicamente de uma distribuição uniforme entre 0 e 1.

```
BEGIN:
Crossover <- (father, mother):
        alpha = random.uniformDistribution()

        child = Individual()

        child.wn = ArithmeticalCrossoverPoint(father.wn, mother.wn, alpha)
        child.ln = ArithmeticalCrossoverPoint(father.ln, mother.ln, alpha)
        child.lp = ArithmeticalCrossoverPoint(father.lp, mother.lp, alpha)
        child.wp = ArithmeticalCrossoverPoint(father.wp, mother.wp, alpha)

        RETURN child
```

```
END
```

Como viu-se na Subseção 3.7.2, há uma variedade de possíveis formas de se realizar o *crossover* e inclusive uma combinação destas formas. Escolheu-se para este exemplo exatamente esta forma, por ter sido a que melhor se adaptou ao caso de estudo em questão.

### 4.1.9 A mutação

O processo de mutação definirá de forma meta-heurística [13] se um dado gene ou conjunto de genes irá ou não ser alterado.

Segundo Linden [14], o valor da probabilidade que decide se o operador de mutação será ou não aplicado é um dos parâmetros do algoritmo genético e só pode ser determinado pela experiência [15]

Implementou-se, neste caso de estudo, o operador de mutação real já explicado na subseção 3.7.3.

```
BEGIN:
Function Uniform_Mutation(Tau, mutationGene, t, y, r, gMax, b):
        factor = 0
        delta = y * (1 - r^((1-gMax)^b))

        IF Tau == 1, THEN:
                factor = delta

        ELSE

                IF Tau == 0, THEN:
                        factor = -delta
                END IF

        ENDI IF

        mutant = mutationGene + factor

        IF mutant is ComplexNumber, THEN:
                mutant = sqrt( (Real(mutant))^2 + (Imaginary(mutant))^2)
        END IF
```

[13] Que aqui se dá de acordo com uma probabilidade.
[14] (LINDEN, 2012),p.87
[15] Eis aqui mais um forte indício de que o projetista não pode, de forma alguma, ser dispensado no processo.

```
22
23          RETURN mutant
24  END
```

Onde:

- $t$ é a geração corrente
- $y$ é o máximo valor da mutação
- $gMax$ o número máximo de gerações que irão executar
- $r$ é um número entre 0 e 1
- $b$ é o parâmetro que controla o grau de dependência do valor da mutação com o número de gerações. Quanto maior este valor, mais rápido o valor de $\Delta$ tende a zero.

Neste caso, pode-se usar, como descreve Linden [16] uma versão do operador que concentre suas alterações em pequenos valores em torno do valor corrente utilizando ou uma distribuição normal ou ajustando os parâmetros à medida que o algoritmo vai sendo executado, segundo alguma estratégia evolucionária.

Algo importante no processo de mutação é a **taxa de mutação**. Este é um parâmetro definido pela inteligência do projetista que de forma empírica e por sua própria experiência poderá escolher o valor mais adequado para o problema em questão [17].

### 4.1.10 Execução do algoritmo

Tendo-se definido as condições iniciais e implementado o algoritmo buscou-se, para fim de experimentação, executá-lo a fim de se encontrar os valores *fittest* para as dimensões dos transistores CMOS a fim de se obter uma porta NAND de 5 entradas, com uma população de 10 indivíduos e apenas 20 gerações. Como se vê no Capítulo 5, o tamanho da população interfere exponencialmente no tempo de execução e no processamento. Logo, uma população menor implicará num algoritmo mais rápido. O número de gerações tem uma interferência linear.

Foram efetuadas 150 execuções para cada combinção, variando-se os parâmetros, e chegou-se a conclusão de que a taxa de mutação tem alguma influência nos algoritmos genéticos, embora não muita. Uma taxa de mutação de 80% (que não é nada realista) apresenta indivíduos que se perdem em mínimos locais e convergem a *scores* na média muito próximos aqueles com taxa de

[16] (LINDEN, 2012), p. 267.
[17] Novamente, torna-se necessária a inteligência do projetista, a qual transcende o universo do AG.

mutação de 50%, embora a melhor execução tenha sido obtida nestas configurações. Todavia, tende igualmente a ter uma maior taxa de extinção: 20% contra os 10% quando a taxa de mutação era de 50%. Populações com uma baixa taxa de natalidade tem uma maior taxa de extinção.

Aumentando-se a taxa de natalidade, aumenta-se linearmente o tempo de execução do algoritmo. Todavia, diminui-se consideravelmente o risco de extinção. Quando a taxa de natalidade é de 100% em relação a população inicial, o tempo de execução praticamente dobra. Todavia, nas 150 execuções apenas 1 levou à extinção - uma taxa de extinção de $0,008\%$. Entretanto, não se obteve melhoras significativas em relação a *fitness*.

Uma combinação entre alta taxa de mutação e alta taxa de natalidade deveria gerar indivíduos mais aptos e populações que praticamente não seriam extintas. Nas 150 execuções efetuadas, percebeu-se que, em média isso não se confirma. A taxa de natalidade predomina sob a taxa de mutação e o melhor resutado obtido é pior do que aquele obtido à taxa de mutação e natalidade de 50% [18].

A tabela 1 apresenta os dez resultados mais significativos das execuções feitas a uma taxa de mutação de 50%. A tabela 2 apresenta o comparativo dos melhores resultados de cada conjunto de execuções.

| Execução | Wn | Wp | Ln | Lp | Score |
|---|---|---|---|---|---|
| **1** | **1.9818066e-05** | **4.8265089e-05** | **1.1262250e-05** | **1.21863e-06** | **0.01323722** |
| **9** | 5.0807525e-05 | 0.000122 | 2.939997e-05 | 1.203778e-05 | 0.042435791 |
| **2** | 2.502797e-05 | 5.920165e-05 | 1.5035663e-05 | 5.8345346e-06 | 0.043344568 |
| **6** | 2.649407e-05 | 8.5567053e-05 | 1.948606e-05 | 9.8376891e-06 | 0.0496691 |
| **8** | 2.7954619e-05 | 6.5671674e-05 | 2.397312e-05 | 7.7896721e-06 | 0.05093009 |
| **5** | 4.7567209e-05 | 0.0001218873 | 3.11955e-05 | 1.4921395e-05 | 0.053171421 |
| **4** | 4.6191028e-05 | 0.00012170472 | 2.718575e-05 | 1.503605e-05 | 0.0542179 |
| **7** | 5.989916e-05 | 0.0001493682 | 3.749563e-05 | 2.051386e-05 | 0.06037676 |
| **3** | 2.971554e-05 | 2.893527e-05 | 2.018315e-06 | 3.2387336e-05 | 0.0809035 |
| **10** | 2.602338e-05 | 2.816931e-05 | 1.959668e-06 | 2.0907119e-05 | 0.1021992 |

Tabela 1 – Resultados mais relevantes das execuções do GA com 50% de mutação

A evolução dos indivíduos da melhor execução, ao longo das gerações, pode ser visualizada no gráfico da figura 27.

**Implementar, portanto, a porta NAND-5 com valores da primeira linha da Tabela 1, resultará em um circuito otimizado em dimensões, tempo de resposta e potência.**.

[18] Taxa de natalidade referente à população inicial.

| Execução | Wn | Wp | Ln | Lp | Score |
|---|---|---|---|---|---|
| AG 0.8 | 3.3511216e-05 | 8.3104845e-05 | 2.5037391e-05 | 1.8459396e-06 | 0.0098314823 |
| **AG 0.5** | **1.9818066e-05** | **4.8265089e-05** | **1.1262250e-05** | **1.218638e-06** | **0.013237221** |
| AG F10 | 5.4655030e-05 | 0.00013122511 | 3.1758401e-05 | 1.121834e-05 | 0.03659373 |
| AG F+M | 5.3557118e-05 | 0.00015015689 | 3.6834466e-05 | 1.4643784e-05 | 0.0415546 |

Tabela 2 – Comparativo dos melhores resultados obtidos nas execuções do AG variando-se os parâmetros de entrada

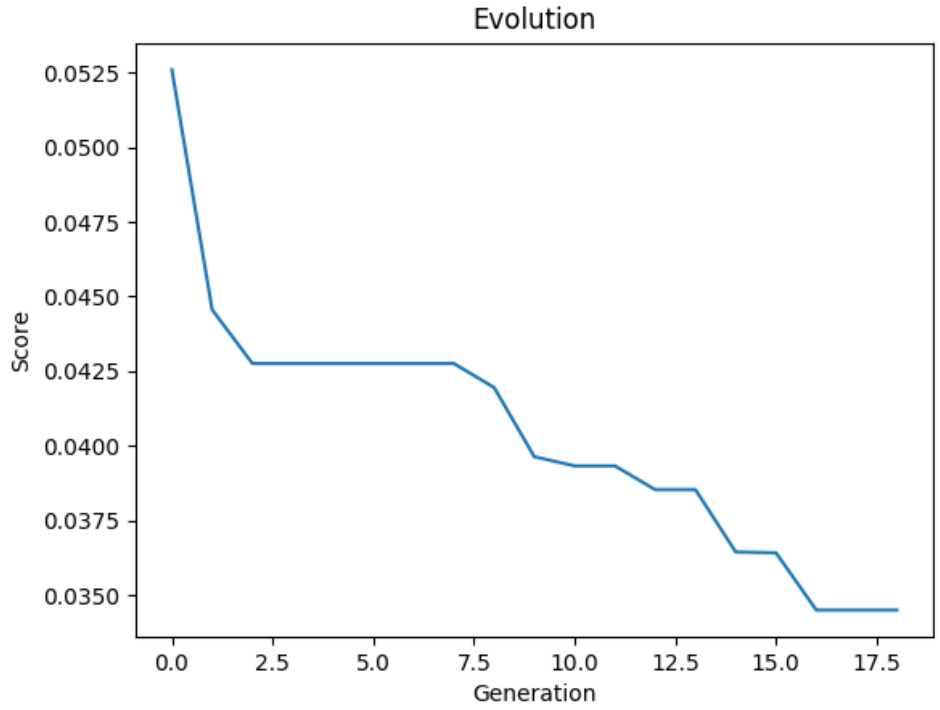



Figura 27 – Evolução dos indivíduos ao longo das gerações

### 4.1.11 A disponibilidade do Framework e do código-fonte

O framework encontra-se disponível em repositório público da internet [19] onde pode ser baixado ou *forked*. Um exemplo de uso encontra-se o arquivo *Runner/Main.py*.

Atualmente, a *branch* **master** contém o código utilizado neste trabalho. Todavia, código está sujeito à melhoria contínua e novas versões surgirão.

Testes automatizados estão sendo criados para o software. Todavia, ainda não estão disponíveis nessa versão acadêmica. Nem tampouco uma compatibilidade comercial, o que surgirá nas próximas versões.

### 4.1.12 Validação do Circuito

Para validação dos valores encontrados, gerou-se o arquivo NETLIST [20] contendo o esquemático do circuito para o qual o algoritmo fora executado: uma porta NAND-5, apresentada na figura 28 onde são considerados os valores ***fittest*** produzidos pelo algoritmo genético.

A seguir, realizou-se a simulação transiente a qual apresentou resultados satisfatórios para variações dos sinais de entrada (A,B,C,D,E). A figura 29 apresenta tais resultados para os valores apresentados no esquemático da figura 28.

No modelo de análise transiente é mostrada a amplitude de um sinal AC em função do tempo. A figura 29 mostra o resultado da simulação no OrCad PSPICE *Student Edition* [21] do circuito apresentado na figura 28.

Quando o OrCad realiza qualquer tipo de simulação AC, primeiramente ele faz uma simulação ***DC Bias*** na qual calcula os pontos de operação quiescentes (DC). Por este motivo, é importante evitar colocar em série impedâncias indutivas e resistivas. Porém, os transistores CMOS apresentam características resistivas e capacitivas, desconsiderando as interferências do substrato.

No caso de um modelo mais realista, devia-se pensar na utilização de um mesmo substrato, o que causaria economias ainda maiores, dado que a diminuição do tamanho dos canais permitiria tal integração.

O resultado desta integração, todavia, é que a mesma geraria ruídos. Estes, podem ser tratados de diversas formas, algumas das quais bem favoráveis (dado que este tipo de circuito é característico por ter rápido chaveamento). Por outro lado, pode ser bastante prejudicial em relação

[19] https://bitbucket.org/againstheretics/ga

[20] Este é um arquivo descritor do esquemático do circuito, ou seja, sua topologia. Neste arquivo, duas linhas são necessárias: a linha de título e a de ".end". As demais linhas são descritivas e seguem as regras que podem ser descritas em diversas bibliografias, tais quais o manual do NgSpice (VOGT; HENDRIX; NENZI, 2018) ou no livro de Roberts e Sedra (SEDRA; ROBERTS; SMITH, 1992), p. 553 em diante.

[21] Que pode ser obtida gratuitamente por qualquer estudante universitário no *web site* da Cadence.

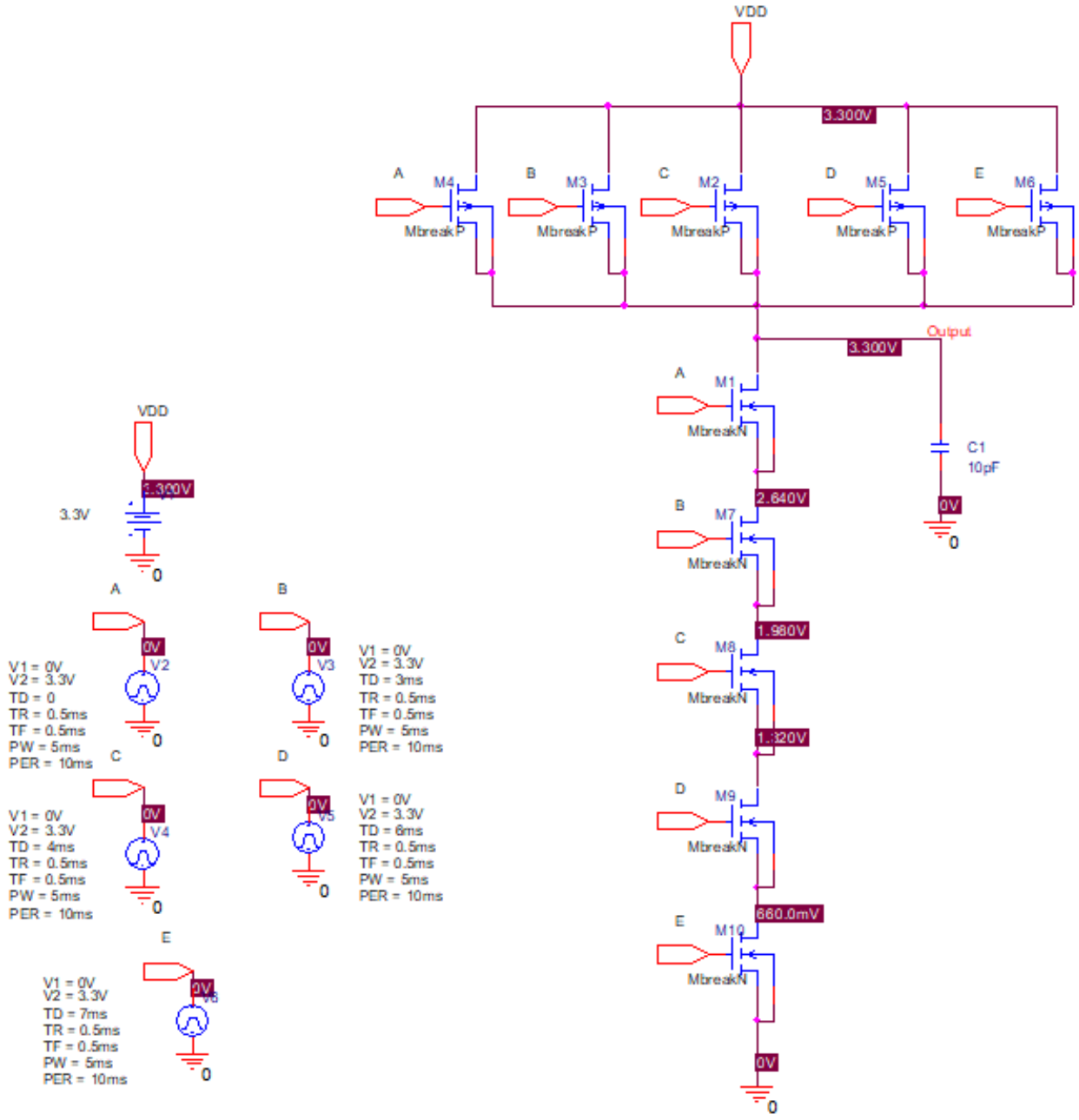



Figura 28 – Esquemático da porta NAND-5

aos patamares de tensão, além de poder ocasionar um acoplamento eletrostático entre as regiões de depleção e outros dielétricos, ou entre condutores paralelos.

A solução pala qual optou-se foi o uso de múltiplos substratos no mesmo circuito integrado.

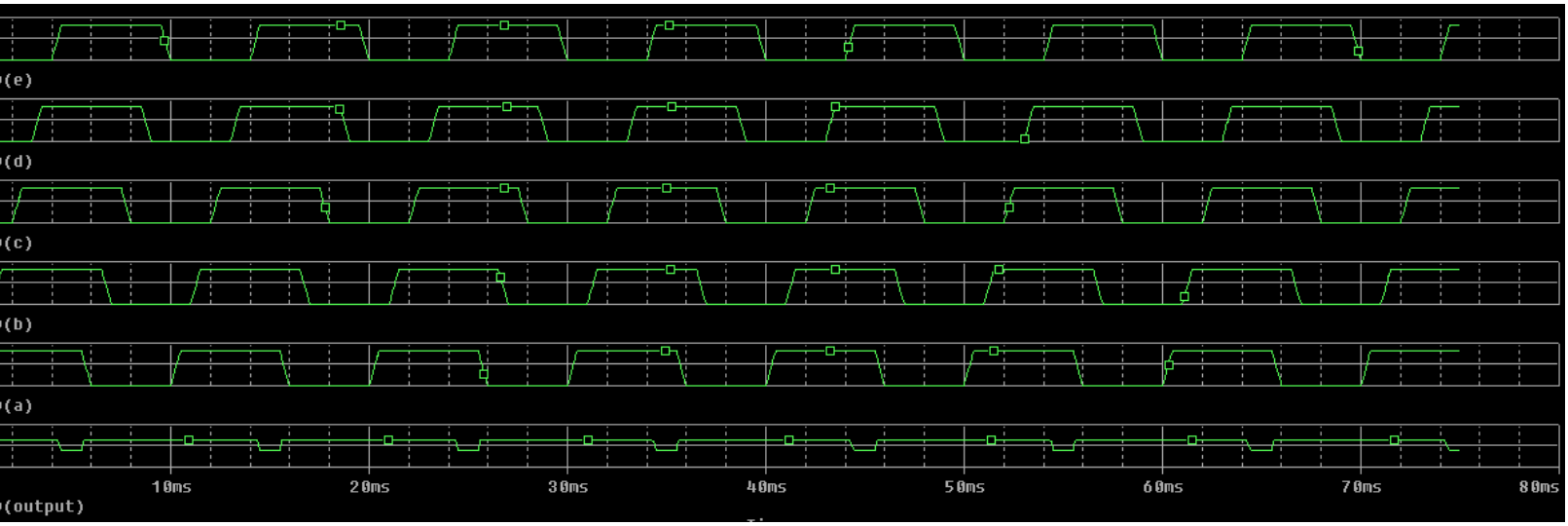



Figura 29 – Simulação para os valores apresentados no esquemático da figura 28.

# 5 Comparativo entre Diversas Implementações

Neste Capítulo implementa-se um comparativo entre os as três principais formas de Algoritmos Evolutivos e Técnicas Padrão de otimização para o caso de estudo deste trabalho: o algoritmo genético, o enxame de partículas e a evolução diferencial.

Fez-se, neste trabalho, dois comparativos entre estes três métodos: um qualitativo (envolvendo os três algoritmos), outro quantitativo (envolvendo apenas o algoritmo genético e o enxame de partículas [1]).

## 5.1 Comparação Qualitativa

Para a comparação qualitativa, consultou-se vasta bibliografia, tendo se baseado principalmente no estudo de (KACHITVICHYANUKUL, 2012). Para o modelo quantitativo, utilizou-se a implementação feita de algoritmo genético neste trabalho e implementou-se o PSO [2], também apresentadaa no repositório deste trabalho e, por fim, comparou-se os valores obtidos.

Por fim, comparou-se os resultados obtidos de nossa implementação 1 com os de (KACHITVICHYANUKUL, 2012), (ZHANG; GEN, 2005), (JARAMILLO; BHADURY; BATTA, 2002), (XIA; WU, 2006).

A comparação qualitativa pode ser vista na tabela 3.

| | **GA** | **PSO** | **DE** |
|---|---|---|---|
| Requer uma classificação das soluções | SIM | NÃO | NÃO |
| Influência do tamanho da população no tempo de solução | Exponencial | Linear | Linear |
| Influência da melhor solução na população | Média | Alta | Baixa |
| Média de fitness não seleciona o pior | FALSO | FALSO | VERDADEIRO |
| Tendência para convergência prematura | Média | Alta | Baixa |
| Continuidade (densidade) do espaço de busca | Baixa | Alta | Alta |
| Habilidade de alcançar uma boa solução sem uma busca local | Baixa | Alta | Alta |
| Subgrupos homogêneos aumentam a convergência | Sim | Sim | Não se aplica |

Tabela 3 – Comparação entre métodos evolutivos

Da tabela 3, infere-se que os GAs consomem mais tempo e mais processamento que o

[1] Escolheu-se esses dois por serem os representantes mais diferentes entre si, e a fim de simplificar também o trabalho.

[2] Enxame de partículas*(Particle Swarm Optimization)*.

PSO e o DE, dado que as soluções precisam ser classificadas, o que também se dá pela influência exponencial do tamenho da população no tempo de solução.

O algoritmo PSO é mais influenciado por um "superindivíduo" que os demais. Por este motivo, os PSO tão logo encontre máximos ou mínimos locais, tendem a convergir para eles rapidamente. No GA, embora esta influência exista, ela pode ser contornada pelas técnicas aplicadas neste trabalho e já discutidas no Capítulo 3.

A grande desvantagem dos GAs é, portanto, relacionada ao tempo e ao consumo de processamento. Atualmente, embora se tenha uma altíssima capacidade de processamento, muitas empresas fazem uso de *clusteres* de computadores na *núvem*, onde se cobra por processamento. Isso faz com que pensar em processamento volte a ser relavante em muitos casos.

Todavia, quando se trata de processos tão exigentes quanto a construção de semicondutores, a precisão costuma ser mais relevante que os gastos com processamento. Ao se supor uma linha de produção automatizada, a construção de um processador utilizando menos matéria prima que mantenha os padrões de qualidade compensa e muito os gastos com execuções de algoritmos na fase de projeto. Isso sem falar no tempo do projetista e tantos outros fatores que fazem ainda com que o processamento seja irrelevante.

Além disso, para os projetos reais, a diferença de tempo de execução de um algoritmo para outro é de poucos segundos, o que pode-se considerar insignificante.

## 5.2 Comparação Quantitativa

Para se comparar quantitativamente a execução do Algoritmo Genético ao PSO, foi necessário implementar o PSO.

Para tanto, utilizou-se a biblioteca ***pyswarms*** do Python.

Esta biblioteca foi desenvolvida por Lester James V. Miranda e seus contribuidores e publicada sob licensa MIT no *Git Hub* [3]. Esta biblioteca surgiu de uma disciplina do projeto de Mestrado onde ele percebeu que poderia contribuir com a comunidade de desenvolvedores criando uma biblioteca fácil para implementar soluções diversas utilizando PSO.

Nosso projeto necessita de uma otimização com fronteiras e deve percorrer os seguintes passos:

- Definir as fronteiras;
- Inicializar o enxame;

[3] <https://github.com/ljvmiranda921/pyswarms>

- Performar a otimização;

A função de avaliação, baseada no que fora apresentado na subseção 4.1.3, foi implementado como a seguir:

```
BEGIN Fitness:

    wn = uniform_value from minWn and maxWn
    wp = uniform_value from minWp and maxWp
    ln = uniform_value from minLn and maxLn
    lp = uniform_value from minLp and maxLp
    M =          number of inputs

    // Physical Conditions

    Cox = 6 * fanton
    unCox = 115 * micro
    upCox = 30 * micro
    Vth_n = 0.4
    Vth_p = - Vth_n
    Vdd = 2.5
    Cj = 0.2 * fanton
    Cjsw = 0.5 * fanton

         // Technology

    LdiffMargin = 0.2
    Ldiff = LdiffMargin * (ln+lp)

    // Expression Load Capacitance

    C = Cox*M*(wp*lp+wn*ln)  + Cj * Ldiff * (wn+wp) + Cjsw * (wn+wp+4*Ldiff
        )

    tphl = C * Vth_n / ((1 / 2) * unCox * (wn / ln) * (Vdd - Vth_n) ** 2)
    tplh = C * Vth_p / ((1 / 2) * upCox * (wp / lp) * (Vdd - Vth_p) ** 2)

    transientPerformanceCondition = abs((wn / ln) - (M / 2) * (wp / lp))
    timeEqualityPerformance = abs(tphl - tplh)

    y = abs(transientPerformanceCondition + timeEqualityPerformance)
```

```
36
37      surface = (wn * ln + wp * lp)*(1.2)  // Adds 20\% for the trails
38
39      score = y + surface*(10**(12))
40
41      results.TPC = transientPerformanceCondition
42      results.TEQ = timeEqualityPerformance
43      results.SRF = surface
44      results.BInd = [wn, ln, wp, lp]
45
46      return score
47
48  END
```

Onde chamou-se:

- $TPC$ à condição de performance transiente;
- $TEQ$ à condição de igualdade de tempo entre $tp_{HL}$ e $tp_{LH}$;
- $BInd$ ao indivíduo encontrado após o processo de otimização.

Diferente dos GAs, este algoritmo não tem "gerações" propriamente ditas, mas sim atualiza as posições durante um processo migratório. Fizeram-se 108 execuções deste algoritmo até a data em que este Capítulo foi escrito, como um comparativo de diversas migrações de aves ou cardumes feitas pelos biólogos. Destas, separou-se as 10 que apresentaram características mais interessantes para serem tratadas neste Capítulo e as disponibilizou nas tabelas 4 e 5.

| Execução | TPC | TEQ | SRF | Score |
|---|---|---|---|---|
| 1 | **0.06061992019577067** | **4.214331179444971e-17** | **1.6172349475082095e-11** | **1.8322153** |
| 2 | 6.7236279006116835 | 2.5798912807703233e-17 | 2.947141530866652e-11 | 2.0137 |
| 3 | 3.7702131653279514 | 1.4030305877708002e-17 | 2.5438264791412102e-11 | 1.5678 |
| 4 | 6.041660237561141 | 1.882036341734946e-17 | 3.872536568882719e-11 | 2.1805 |
| 5 | 40.01238048341742 | 1.4370845968674623e-17 | 7.733747525267992e-12 | 1.1067076 |
| 6 | 7.588953264145229 | 5.4040269725039764e-18 | 7.574951991147571e-12 | 1.6656 |
| 7 | **14.436243528268829** | **4.668397517960424e-18** | **5.671152747123853e-12** | **2.2408** |
| 8 | **2.4812139649235476** | **1.2438587872378096e-17** | **3.18322734821428e-11** | **0.9842** |
| 9 | 8.115864117674096 | 9.570870542321972e-18 | 1.103412688108483e-11 | 1.6686080 |
| 10 | 5.962305682531868 | 2.442139183095478e-17 | 3.5247911437482323e-11 | 2.6381 |

Tabela 4 – Resultados mais relevantes das execuções do PSO.

| Execução | Wn | Ln | Wp | Lp |
|---|---|---|---|---|
| 1 | 2.995175824018258e-06 | 1.5092433554461786e-06 | 1.9140336066364105e-06 | 4.679389460067373e-06 |
| 2 | 1.4704733195528766e-06 | 2.0766591854832166e-06 | 5.653768011029371e-06 | 3.8038067337059156e-06 |
| 3 | 1.6332074292436635e-06 | 4.838754131535741e-07 | 5.400498903009478e-06 | 3.7789629143114816e-06 |
| 4 | 2.332004960227789e-06 | 1.1742692638029825e-06 | 6.8858756031633555e-06 | 4.2888861240281196e-06 |
| 5 | 6.168229590635265e-07 | 1.0945242414885735e-06 | 6.842651843420202e-06 | 8.431909229075093e-07 |
| 6 | 3.5932871526841303e-06 | 4.649974655723253e-07 | 3.770753163997234e-06 | 1.2309452174882969e-06 |
| 7 | 3.3591916022534803e-06 | 1.0032456121800524e-06 | 2.19606455496087e-06 | 6.174073453840186e-07 |
| 8 | 1.497800214667055e-06 | 1.4649705942145042e-07 | 6.382721757190692e-06 | 4.121669758166277e-06 |
| 9 | 3.258808132724919e-06 | 1.4178935466716324e-06 | 3.0867267442682605e-06 | 1.4819785138091594e-06 |
| 10 | 1.500966404712225e-06 | 1.565872807609289e-06 | 6.115910899209084e-06 | 4.418464804066383e-06 |

Tabela 5 – Resultados mais relevantes das execuções do PSO

Observa-se da tabela que não necessariamente a que obteve o menor *score* obteve também as menores características individuais (SRF, TEQ, TPC, e valores dos componentes). Isso se dá pela característica do algoritmo PSO que calcula velocidade e deslocamento. Ao mesmo tempo, este tipo de algoritmo é interessante para a tomada de decisão a respeito do que o projetista deseja sacrificar. Pode ser que otimizar a área seja mais importante que o tempo ou até mesmo que todo o conjunto. Esta implementação permite este tipo de tratamento fino por parte do projetista ou até mesmo de outro algoritmo.

Sem dúvidas que esse tipo de abordagem também pode ser implementada em algoritmos genéticos. No caso específico de nossa implementação esta decisão pode ser tomada com base no tratamento do arquivo *output.txt* que loga todos os indivíduos das diversas gerações.

# 6 Conclusão do trabalho

## 6.1 Conclusões Gerais

Durante esta dissertação, diversas áreas do conhecimento foram tocadas. Primeiramente, foi estudado e resumido o estado da arte na disciplina de Microeletrônica, apresentando-se um material sólido que pode ser utilizado no ministério desta disciplina. Posteriormente, a área de Algoritmos Evolucionários forneceu sólido material de estudo do estado da arte da mesma, referenciando as bibliografias, das mais antigas às mais atuais sobre o tema [1]. Por fim, tocou-se na Biologia onde se tratou da questão a respeito da Origem das Espécies, apresentando indícios matemáticos, linguísticos e filosóficos de que uma das teorias mais icônicas da modernidade pode ser falseável. Fez-se por fim a implementação do que fora proposto em um caso de estudo: a otimização de um circuito NAND-5 em tecnologia CMOS, mostrando ser perfeitamente possível utilizar não apenas algoritmos genéticos, mas toda gama de algoritmos evolucionários para otimizar o processo construtivo de circuitos feitos nessa tecnologia. Sendo assim, o objetivo proposto fora atingido, conseguindo resultados práticos e teóricos auspiciosos em uma classe de problemas complexos.

Qualitativamente, este trabalho apresenta parâmetros suficientes no Capítulo 5 para que o projetista possa seguramente escolher entre tais e quais técnicas. Quantitavamente, todas elas oferecem resultados muito próximos.

A área de Algoritmos Genéticos, de maneira mais específica, não foge ao descrito neste trabalho. Há algumas outras técnicas, porém em sua vastíssima maioria, variações das aqui apresentadas. Estas mesmas técnicas que foram apresentadas são também utilizadas em processamento paralelo a fim de, com isso, acelerar ainda mais os cálculos.

Surpreendentemente, percebeu-se que a taxa de mutação tem influências interessantes sobre os indivíduos. Por um lado ela aumenta a diversificação de uma classe de indivíduos em uma dada geração, aumentando o espaço amostral dos processos de avaliação. Todavia, ao contrário do que se possa pensar [2], ela não é a responsável pelo aparecimento de superindivíduos. A recombinação (*crossover*) é a função vetorial responsável pela geração de superindivíduos, pois estes podem ser gerados mesmo com taxas baixíssimas de mutação, se não forem tomadas as precauções descritas na Seção 3.6. Além disso, conforme se observa na Subseção 4.1.10, uma alta taxa de mutação leva a população à extinção, o que corrobora com as pesquisas na Biologia de Behe, como dissertado no Apêndice D.

[1] O livro de Linden (LINDEN, 2012) ainda é uma das obras mais completas a este respeito em língua portuguesa.
[2] E diga-se de passagem, ao contrário do que afirma a Teoria Evolucionista na Biologia.

A taxa de natalidade é extremamente importante para diminuir o risco de extinção. Todavia, é diretamente proporcioanl ao tempo de execução em ordem polinomial, e também predomina sobre a taxa de mutação.

As discussões científicas a respeito dos temas que inspiraram o mesmo são acaloradas, envolvendo diversas correntes do pensamento humano e de suas crenças. Todavia, a Ciência [3] é capaz de trazer as luzes da razão aos debates, guiando o homem através das evidências tanto experimentais quanto dissertativas.

As evidências e discussões levantadas neste trabalho levantam questões aos modelos evolucionistas na Biologia e corroboram com as propostas de Behe (BEHE, 2006) em relação ao *Intelligent Design*.

A própria estrutura que modela o Indivíduo em uma classe que pode ser matematicamente representada pelo vetor $\vec{I} = (W_N, W_P, L_N, L_P)$ já contraria semânticamente a proposta da especialização darwinista. Na Teoria da Evolução, um indivíduo biológico primitivo é um vetor com um conjunto de características que o definem enquanto tal como um **"ente"** vivo. Valendo-nos da linguagem matemática, mostramos que, mantendo a natureza dos atributos, alterá-los por processos de mutação só tem um efeito degenerativo, jamais de especialização, semelhante ao que seria mudar a cor dos pelos, a altura, a densidade óssea ou muscular. Todavia, se a natureza dos atributos se alterar (por exemplo, $W_P$ for ***String***, o vetor $\vec{I_2} = (W_N, W_P, L_N, L_P)$ já não mais será da mesma espécie que antes, ainda que a letra (ou conjunto de letras) tenha a mesma representação binária de um dado número de $\vec{I}$ [4]. Isso, entreatanto, não se trata de maneira alguma de especialização, pois esta exigiria que *inteiro* se tornasse *string* aleatoriamente após execuções diversas desse algoritmo, ou seja, que os entes computacionais fossem capazes de modificar suas próprias estruturas, violando seu encapsulamento pelo simples fato de serem todos compostos de bits. Tal pressuposição é simplesmente absurda.

## 6.2 Comparação com outros trabalhos

O trabalho de Filgueiras (FILGUEIRAS, 2010) é um ótimo compemento a este trabalho. Este trabalho, todavia, utiliza um conjunto maior de técnicas como Pareto (BARRA et al., 2012) e um modelo mais enxuto na função de avaliação tal qual recomendado no trabalho de Zebulum (ZEBULUM; PACHECO; VELLASCO, 1998); também utiliza linguagem mais descritiva e levanta

[3] Novamente, no seu conceito mais amplo, como tratado, e não apenas no Cartesianismo (ou "'Ciência Moderna"').

[4] Se, pois, a Matemática e as Linguagens de Programação são linguagens autênticas, representam elas em seu microuniverso algo da realidade que descrevem. Sendo linguagens fundamentalmente lógicas, são superadas pelas linguagens mais elaboradas e certamente pelas línguas em sua capacidade de descrever a realidade, todavia, aquilo que tais linguagens inferiores consegue descrever não podem as línguas contrariar, dada que estas são subconjunto daquelas. Em outras palavras, ao se provar linguísticamente pela Matemática e pela Computação um determinado fato, este não pode ser contrariado racionalmente no mesmo universo por Arte ou Ciência que lhe seja superior.

discussões em várias áreas do conhecimento. É também um trabalho complementar, dado que é validado em simulador comercial (provavelmente o mais confiável), testa um conjunto diferente de circuitos (validando a tese de Filgueiras), transiente, e compara com a implementação em outras técnicas evolucionárias.

## 6.3 Aplicabilidade do trabalho

Tendo obtido resultados em diversas áreas do conhecimento humano, este trabalho pode ser aplicado tanto no ensino das disciplinas como para enriquecer o debate tanto sobre tecnologias evolucionárias, quanto sua aplicação em Microeletrônica, assim como pode ser utilizado para melhoria dos processos construtivos de circuitos integrados.

O *framework produzido* pode ser "baixado" via *internet* e utilizado em softwares privados para otimização de processos, assim como para o estudo das técnicas e testes.

## 6.4 Sugestão de trabalhos futuros

A seguir são apresentadas sugestões de possíveis trabalhos futuros:

- Melhorar a performance do algoritmo utilizando técnicas mais avançadas de programação paralela.
- Integrar outras técnicas evolucionárias ao *framework* produzido.
- Criar interface de integração com diversos simuladores comerciais e gratuitos, ao invés de apenas com OrCad.
- Criar interface WEB que receba um **NETLIST** e automaticamente crie o modelo, decida os principais valores e retorne um novo *NETLIST* com os valores atualizados.
- Integrar o *framework* com a criação de PCB.
- Implementar este algoritmo para sistemas biológicos a fim de validar num universo mais próximo as conclusões aqui alcançadas que apontam para as falhas do evolucionismo darwinista.
- Encontrar o perfeito equilíbrio entre taxa de mutação, de natalidade e de *crossover* que promoverá a maior convergência gênica e 0% de extinção.

# Apêndices

# APÊNDICE A – Noções Introdutórias de Microeletrônica

Neste Apêndice faz-se a revisão dos conceitos introdutórios de Microeletrônica. Ao seu termo é possível ter relativo grau de familiaridade com os circuitos MOS e com seu processo de fabricação, que servem de base para o capítulo 2.

O que se explica nas subseções seguintes a respeito dos transistores do tipo N, tem seu correspondente equivalente em relação aos transistores do tipo P. (FERREIRA, 2004)

## A.1 Transistor MOS

O conceito básico em torno dos semicondutores e da Microeletrônica é o conceito de dopagem. A dopagem consiste na adição de impurezas em um elemento semicondutor da Tabela Periódica com o objetivo de que ele se torne mais condutor. Geralmente o semicondutor é o silício (devido a possuir 4 elementos na camada de valência e outras propriedades) e duas impurezas lhe são adicionadas: adição de arsênico ou fósforo (**dopagem do tipo N**); e a adição de boro ou gálio (**dopagem do tipo P**). O fato de o silício ser tetravalente lhe permite formar estruturas cristalinas ou reticuladas. Na dopagem N, ligações covalentes entre o fósforo (ou arsênico), pentavalentes, e o silício - tetravalente - geram um **elétron livre**, ficando a estrutura cristalina negativa [1]. Na dopagem P, o boro (ou gálio), trivalentes, conseguem realizar apenas três ligações covalentes. Tal ligação exige destes átomos uma reorganização interna de suas estruturas físicas de forma que o elétron, ao se deslocar, deixa uma "lacuna". De fato, esta lacuna não existe fisicamente como ente portador de carga elétrica (como se fosse um próton, por exemplo), porém possui existência matemática devido ao fato de, na dinâmica dos fluxos dos portadores de carga, a ausência de elétrons naquela região comportar-se como se ali houvesse uma carga positiva (PEREIRA, Online. Acessado em 13/11/2017). Este conceito é o principal relevante no estudo dos semicondutores: a existência de lacunas e elétrons como portadores de cargas, os quais são transferíveis pela influência do campo elétrico. O circuito semicondutor mais simples é o diodo o qual consiste apenas de uma junção PN. Depois dos diodos [2], temos os transistores MOS. O transistor MOS é um circuito semicondutor que permite a passagem de corrente elétrica entre a **fonte** e o **dreno** [3] quando a tensão no ***gate***

[1] De onde provem o nome N.
[2] Há uma infinidade de diodos, porém não cabe a este trabalho explicá-los mais a fundo.
[3] *Source* e *Drain*

ultrapassa a tensão limiar [4] desde que exista uma diferença de potencial entre a fonte e o dreno. O transistor é fabricado em cima de uma lâmina de silício cristalino, chamada **substrato**, do tipo P, sob a qual é colocada um fina camada de dióxido de silício ($SiO_2$) de espessura $t_ox$ [5] Para formar o *gate*, é depositado um metal em cima da camada de $SiO_2$. Dois poços de dopagem N são formados em cima dos quais se acrescentam também contatos metálicos. Este processo será explicado de forma mais detalhada na próxima subseção. Como pode-se observar a relação Metal-Óxido-Semimetal, forma uma capacitância no *gate*. A região de depleção é formada mediante a aplicação de uma diferença de potencial entre o *gate* e a fonte e entre o *gate* e o dreno [6]. Ao aplicar uma tensão positiva no *gate* ($V_G$), devido a sua característica capacitiva, ele repele as lacunas livres da região do substrato [7]. "Estas lacunas são empurradas para baixo, deixando uma região depelada de portadores" de carga, ao mesmo tempo que atrai elétrons das regiões N do dreno e da fonte, gerando assim um **canal N** (SEDRA; SMITH, 2004). O valor da diferença de potencial entre o *gate* e o dreno para o qual um número suficiente de elétrons móveis se acumulam na região do canal para formar um canal de condução é chamada de **threshold** [8]. À medida que se aumenta a tensão da porta, essas cargas do tipo N vão sendo cada vez mais atraídas e a região, que era do tipo P do substrato e não condutora, vai assumindo uma característica condutora e se tornando uma zona do tipo N [9] A condutividade do canal é modulada pela tensão do *gate* [10]. O canal desaparece se a tensão $V_{GS}$ for inferior a $V_T$ e, neste caso, o transistor comporta-se como interruptor aberto. Conforme pode ser visto, o funcionamento do transistor é intimamente relacionado às suas características físicas, especialmente da zona de forte inversão: material de fabricação e dimensões de fabricação - **largura** e **comprimento** do canal. Adicionalmente, ao se aplicar uma tensão entre dreno e fonte ($V_{DS}$), ainda que pequena, se verá a passagem de corrente pelo canal formado proporcional à tensão aplicada.

$$I_D \propto V_{DS}$$

Assim, quanto maior a tensão $V_{DS}$, maior a corrente elétrica que passa pelo canal. Entretanto, chega um momento em que o canal satura e a corrente permanece essencialmente constante e, consequentemente, independente de posteriores aumentos de $V_{DS}$. Esse efeito é chamado **estrangulamento do canal** [11].

[4] Ou *treshold*, $V_T$, que é oriunda do processo de fabricação.
[5] geralmente 2-50nm. Usa-se o óxido de silício por ser um dos melhores isolantes existentes.
[6] Considere para esta primeira análise que fonte e dreno estão aterrados.
[7] Também chamado corpo, *body* em inglês, que dá origem a $V_B$ - tensão de substrato.
[8] Ou tensão de limiar.
[9] Este fenômeno se chama **forte inversão** (*strong inversion*).
[10] A medida que aumenta a tensão $V_{GS}$, menor é a resistência do canal e maior é a corrente.
[11] Também chamado *pinch-off*, ou tunelamento.

Depois, há quatro fatores que influencim na tensão limiar: o primeiro é a *work function* do material [12]; o potencial da superfície [13]; a carga de depleção [14]; e a carga originária das impurezas. Isso se o corpo e o substrato estiverem no mesmo potencial, geralmente aterrados. Em alguns circuitos, entretanto, isso não ocorre e uma diferença de potencial $V_{SB}$ aparece, o que modifica a carga de ionização e, consequentemente afeta também a tensão limiar (ZEGHBROECK, 2011). Assim, num caso mais geral, pode-se calcular a tensão limiar pela equação A.1 a seguir.

$$V_T = V_{T0} + \gamma(\sqrt{|-2\phi_F + V_{SB}|} - \sqrt{|-2\phi_F|}) \tag{A.1}$$

Em que $\gamma$ é o coeficiente de efeito de corpo e $\phi_F$ é o potencial de Fermi.

## A.2 Análise da operação do CMOS

Nesta subseção é analisada mais profundamente a operação do CMOS. Para isso a engenharia utiliza dois modelos baseados no comportamento do circuito em suas duas regiões de operação. É importante verificar que ambos modelos são aproximações, ainda que muito boas, do comportamento real dos semicondutores.

Para tensões $V_{DS}$ muito pequenas, o transistor comporta-se como um resistor e sua corrente cresce de forma linear, o que dá o nome desta região de operação [15]. Uma segunda aproximação se faz para tensões $V_{DS}$ mais elevadas onde o circuito atinge a saturação (ZEGHBROECK, 2011).

## A.3 Região de Triodo e Modelo linear

Nesta região o CMOS se comporta como um circuito linear, podendo ser modelado por uma resistência linear controlada pela tensão $V_{GS}$. Então ele pode ser utilizado como um *switch* para sinais digitais e analógicos ou um multiplicador. Desde que a tensão entre o *gate* e a fonte supere a tensão limiar até o *pinch-off* do canal, a corrente entre fonte e dreno $I_D$ no canal cresce proporcionalmente à tensão entre o gate e a fonte e pode ser calculada pela expressão A.7. (FERREIRA, 2004)

[12] O termo é frequentemente encontrado como **workfunction**, porém o termo em português ("função trabalho") não tem boa sonoridade e não é muito utilizado na bibliografia, embora se veja alguns autores fazendo uso.
Na Física do Estado Sólido é o mínimo trabalho termodinâmico (*ide est*, energia) necessário para remover um elétron de um sólido (do nível de Fermi) para um ponto no vácuo imediato à superfície do sólido. É, pois, uma medida da barreira potencial que os elétrons precisam vencer para abandonarem o material. Esta barreira potencial a ser vencida é diferente no material do gate e do semicondutor utilizado no substrato.

[13] Que possui valor absoluto igual ao dobro do potencial de Fermi

[14] Relacionada a ionização dos átomos dopantes na região de depleção antes de ocorrer a *strong-inversion*.

[15] Também chamada de **triodo**.

A corrente elétrica é diretamente proporcional à carga no canal e inversamente proporcional ao tempo necessário para que o fluxo de carga percorra o canal ($t_{Transit}$).

Desta forma, se $A_{Canal}$ é a área do canal, tem-se:

$$\Delta Q = -Q_{INV}.A_{Canal} = Q_{INV}WL \tag{A.2}$$

Onde $\Delta Q$ é a carga no canal.

$$v = \frac{L}{t_{Transit}} \rightarrow t_{Transit} = \frac{L}{v} \tag{A.3}$$

Onde $v$ é a velocidade que a carga percorre o canal. Esta velocidade, de acordo com a Eletrostática é produto da mobilidade elétrica das partículas no material pelo campo elétrico.

$$\vec{v} = \mu\vec{E} \rightarrow \|\vec{v}\| = \mu\left\|\vec{E}\right\| = \mu\frac{V_{DS}}{L} \tag{A.4}$$

A Eletrostática também nos permite dizer que a carga elétrica num capacitor é igual ao produto da capacitância pela diferença de potencial entre as paredes do mesmo. Assim sendo, pode-se escrever:

$$Q_{INV} = -C_{ox}(V_{GS} - V_T) \tag{A.5}$$

Para $V_{GS} > V_T$ [16].

Quando a tensão entre o dreno e a fonte ($V_{DS}$) não é suficientemente pequena, percebe-se o efeito da resistência do canal que provoca uma perda de tensão aproximadamente igual à metade do quadrado desta tensão. Valor este que pode ser desconsiderado caso a tensão $V_{DS}$ seja suficientemente pequena.

$$V_{Lost} = \frac{{V_{DS}}^2}{2} \tag{A.6}$$

Assim sendo, a corrente na região de triodo é representada neste modelo pela equação A.7.

$$I_{Dtriodo} = \mu_N C_{ox}\frac{W}{L}\left[(V_{GS} - V_T)V_{DS} - \frac{{V_{DS}}^2}{2}\right] \tag{A.7}$$

[16] Condição necessária para região de Triodo ou linear.

## A.4 Região de Saturação e Modelo Quadrático

À medida em que aumenta a tensão entre a fonte e o dreno, a diferença de potencial entre o *gate* e o dreno vai se reduzindo e a corrente $I_D$ da equação A.7 atinge seu valor máximo, de forma que o transistor se comporta como uma fonte de corrente controlada por $V_{GS}$. Isso ocorre devido ao fato de a tensão no dreno ($V_D$) já não ser mais suficiente para manter o canal, implicando em $V_{GD} \leqslant V_T$. Isso ocorre quando $V_{GS} - V_{DS} \leqslant V_T$ [17] (ZEGHBROECK, 2011).

Para modelar o transistor nessa região utiliza-se o modelo quadrático, o qual assume as mesmas premissas do modelo linear. Utiliza-se este modelo devido ao fato de, nesta região, ser necessário modelar a variação da inversão de carga entre a fonte e o dreno (ZEGHBROECK, 2011).

Considere uma pequena seção dentro do dispositivo com largura $dy$ e a tensão no canal $V_C + V_S$, onde $V_C$ é a tensão local do canal. Certamente, na região $dy$ a tensão do canal também será infinitesimal, portanto $dV_C$.

A lógica utilizada para inferir o modelo linear da equação A.7 ainda é válida, o que implicará na equação A.8.

$$I_D = \mu_n C_{ox} \frac{W}{dy}(V_G - V_S - V_C - V_T)dV_C \tag{A.8}$$

Sabe-se que a corrente $I_D$ é contínua através do canal e vê-se na equação que $V_{DS}$ fora substituído pela tensão do canal. Assim, para encontrar a equação da corrente de dreno, basta resolver a equação diferencial A.8.

$$\int_0^L I_D dy = \int_0^{V_{DC}} \mu_n C_{ox} W(V_G - V_S - V_C - V_T)dV_C \tag{A.9}$$

Como $I_D$ é constante em relação a $dy$, a equação A.9 resulta em:

- Para $|V_{DS}| < V_{GS} - V_T$:

$$I_{D,sat} = \mu_n C_{ox} \frac{W}{L}\left[(V_{GS} - V_T)V_{DS} - \frac{{V_{DS}}^2}{2}\right] \tag{A.10}$$

  O que confirma o modelo linear, para a região linear: A corrente de dreno, portanto, primeiro cresce linearmente até atingir o máximo onde vai para zero e invertendo sua carga total (lacunas). Uma vez que o dreno e o substrato formam um diodo reversamente polarizado, as lacunas não podem contribuir para a corrente $I_D$. Por esse motivo ocorre a saturação.

[17] O que é a condição para se estar na região de saturação.

- Para $V_{DS} > V_{GS} - V_T$:

$$I_{D,sat} = \frac{1}{2}\mu_n C_{ox}\frac{W}{L}(V_{GS} - V_T)^2 \tag{A.11}$$

Este modelo recebe o nome de modelo quadrático porque a saturação ocorre à direita da curva:

$$I_D = \mu_n C_{ox}\frac{W}{L}{V_{DS}}^2 \tag{A.12}$$

Todavia, a corrente de dreno medida na saturação não é constante como esperado por este modelo. De forma, não ser este o modelo ideal, ainda que muito bom. Experimentalmente percebe-se um fator multiplicador desta tensão que a faz crescer linearmente ainda que com coeficiente angular muito baixo. Assim sendo, o modelo empírico derivado deste modelo quadrático, para a região de saturação é [18]:

$$I_{D,sat} = \frac{1}{2}\mu_n C_{ox}\frac{W}{L}(V_{GS} - V_T)^2(1 + \lambda V_{DS}) \tag{A.13}$$

Onde $\lambda$ é o fator de correção linear.

## A.5 Processo de fabricação de um circuito CMOS

"Em sua maioria, a indústria moderna utiliza o *wafer* silício como matéria prima, os quais são cortados a partir de tarugos de cristal de silício, retirados de um cadinho com silício policristalino puro - processo conhecido como **Método de *Czochralsky***. Adiciona-se ao silício, ainda líquido, uma quantidade controlada de impurezas para que as propriedades elétricas desejadas sejam obtidas no cristal de silício resultante. Uma semente cristalina é mergulhada no silício líquido para determinar sua orientação cristalina. O silício fundido é mantido em um cadinho de quartzo envolvido por um radiador de grafite. O grafite é aquecido por indução de radiofrequência, e a temperatura é mantida em alguns graus acima da temperatura de fusão do silício ( $1425^oC$), tipicamente em uma atmosfera de hélio ou argônio"(PIMENTA, 2015).

Após a semente ter sido introduzida no silício líquido, ela é puxada gradualmente no sentido vertical e, ao mesmo tempo em que é rotacionada. O silício policristalino fundido derrete a ponta da semente e, à medida que a semente é puxada ocorre esfriamento [19]. Quando o silício líquido, em contato com a semente, esfria, e assume a forma e a orientação cristalina da semente. O diâmetro do tarugo é determinado pelas taxas de velocidade de tracionamento e de rotação. A formação do tarugo varia de 30 a $180mm/h$ (PIMENTA, 2015).

[18] (ZEGHBROECK, 2011)
[19] Solidificação.

O corte em fatias do tarugo é usualmente feito por meio de serras com diamantes nos dentes girando em alta rotação. Os *wafers* obtidos têm usualmente entre $0,25mm$ e $1mm$ de espessura, dependendo do seu diâmetro. Finalmente, os discos são polidos numa de suas faces até se obter um acabamento espelhado e sem imperfeições (PIMENTA, 2015).

Tendo o *wafer*, numa *Cleanroom* [20], faz-se o processo de oxidação térmica [21]. Este processo tem por objetivo a obtenção do óxido de silício ($SiO_2$), o qual será o dielétrico do *gate* e também uma máscara contra impurezas (PIMENTA, 2015).

O processo pode ser descrito pela equação A.14 [22].

$$Si + O_2 \xrightarrow{T} SiO_2, T > 1000^oC \tag{A.14}$$

A segunda parte do processo é a deposição de óxido. Esta etapa tem por objetivo a obtenção de óxido de silício ($SiO_2$) sobre o silício ou outra superfície qualquer. Sua principal função é o mascaramento de impurezas.

Tal processo é feito pela equação A.15.

$$SiH_4 + O_2 \xrightarrow{T} SiO_2 + 2H_2 \uparrow, T = 500^oC \tag{A.15}$$

Segue-se para a etapa de abertura de janelas através de fotogravação e corrosão química [23]. A fotolitografia é o processo pelo qual retira-se o óxido de silício, silício policristalino ou alumínio de certas regiões, determinadas pela fotomáscara. Esta etapa tem por objetivo a formação das regiões necessárias para o *source*, dreno e *gate*, assim como a definição das vias de interconexão. Nesta etapa, cobre-se o *wafer* com um filme de um material fotossensível, conhecido como *"photoresist"*. "Uma máscara com áreas claras e opacas, que representam o padrão a ser transferido para o *wafer*, é colocada sobre o material fotossensível e, por exposição à luz ultravioleta, este material será polimerizado nas regiões correspondentes às áreas claras da máscara. Retira-se a máscara e o *wafer* é então revelado usando-se produtos químicos (tais como tricloroetileno), os quais dissolvem as áreas não polimerizadas (PIMENTA, 2015).

A superfície apresentará, então, o padrão desejado. Este procedimento descrito corresponde ao ***photoresist* negativo**, sendo possível também o ***photoresist* positivo**, onde a área exposta à luz ultravioleta é removida"(PIMENTA, 2015).

[20] Uma sala com ambiente controlado, limpo de quaisquer impurezas - mínimas que sejam.
[21] A oxidação pode ser seca (*dry oxide*), usando oxigênio de alta pureza, ou molhada (*wet oxide*), usando vapor d'água, com oxigênio de alta pureza.
[22] Algumas bibliografias apontam $900^oC$ como uma temperatura suficiente para o processo.
[23] Etapa também chamada de fotolitografia.

O processo seguinte à fotolitografia é a remoção seletiva de material das áreas do *wafer* desprotegidas pelo *photoresist* - chamada ***etching***. Um único DI é submetido geralmente a vários *etchings* durante sua fabricação.

Segue-se então para o processo de deposição. "A deposição de camadas condutoras e isolantes constitui uma etapa importante do processo de construção do CI [24]. Este processo não consome silício do substrato, como a oxidação térmica e divide-se em dois modos: Deposição Física de Vapor [25] e Deposição Química de Vapor [26]. No PVD não ocorre reação química durante o processo, enquanto no CVD ocorre [27].

A próxima etapa é a difusão de impurezas. Esta etapa tem por objetivo introduzir na rede cristalina do $Si$ impurezs doadoras (fósforo, arsênio, etc.) ou aceitadoras (boro, etc.). Sua função é criar uma região com características doadora ou aceitadora.

Uma outra forma de se introduzir impurezas é a implantação iônica. Esta tem por objetivo introduzir na rede cristalina do $Si$ impurezas doadoras ou aceitadoreas por impacto.[28]

Pode-se resumir, portanto, o processo de fabricação do CI em dezessete passos (NICOLETT, 2017):

- Oxidação térmica;
- Fotogravação e Corrosão do $SiO_2$ para deposição das regiões tipo **N** [29];
- Implantação iônica de fósforo;
- Remoção total do $SiO_2$;
- Deposição de $SiO_2$;
- Fotogravação e Corrosão do $SiO_2$ para definição das regiões de difusão dos tipos **N** e **P** [30];

[24] Circuito Integrado
[25] Physical Vapor Deposition (PVD).
[26] Chemical Vapor Deposition (CVD).
[27] Um exemplo de CVD é a deposição de alumínio, no qual o alumínio é vaporizado. E um exemplo de CVD é o crescimento epitaxial" (PIMENTA, 2015)
[28] "Em um ambiente de baixa pressão (idealmente vácuo), íons apropriados (Boro para tipo **P** e Fósforo para tipo **N**) são acelerados de forma a adquirirem uma alta energia cinética em direção ao **wafer**. A profundidade de penetração está relacionada com a energia do feixe de íons, a qual pode ser controlada pela tensão do campo de aceleração. A quantidade de íons implantada pode ser controlada pela variação da corrente do feixe (fluxo de íons). Uma vez que a tensão e a corrente podem ser medidas e controladas com precisão, a implantação iônica resulta em perfis mais precisos e que podem ser reproduzidos facilmente. Como outra vantagem , a implantação iônica é feita a temperatura ambiente e não apresenta difusão lateral sendo recomendada para fabricação de CIs de alta densidade". (PIMENTA, 2015).
[29] Máscara NW
[30] Máscaras DN e DP

- Deposição de silício policristalino dopado;
- Fotogravação e corrosão do silício policristalino (*poly*) [31];
- Fotogravação do fotorresiste para definição das regiões **P+** [32];
- Implantação iônica de Boro;
- Remoção do fotorresiste;
- Fotogravação do fotorresiste para definição das regiões **N+** [33];
- Implatação iônica de fósforo;
- Deposição de $SiO_2$;
- Fotogravação e corrosão do $SiO_2$ [34] para definição dos contatos; Fotogravação e corrosão do alumínio [35] para definição do metal.

A figura 30 representa todas estas fases justapostas.

[31] Máscara PO
[32] Transistor PMOS e contato com substrado; Máscara DP
[33] Máscara DN
[34] Máscara CO
[35] Máscara ME

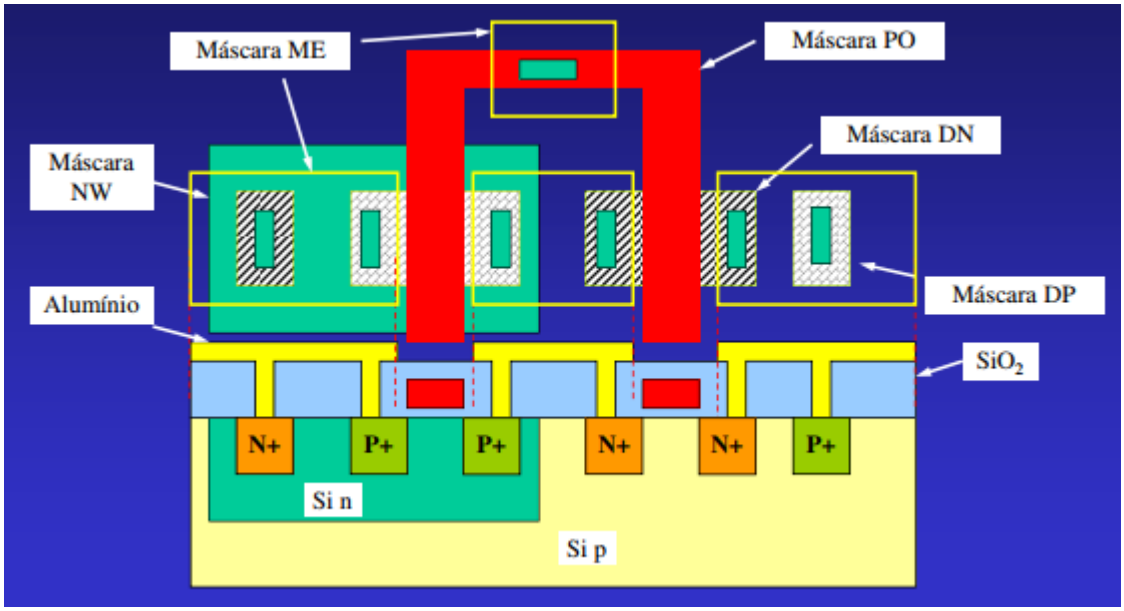



Figura 30 – Processo Construtivo CMOS (NICOLETT, 2017).

# APÊNDICE B – Modelos Não-Evolucionários

Neste apêndice expõe-se algums teoremas e métodos de otimização que não são evolucionários e que devem compor o ferramentário para a solução de problemas. Tais métodos são apresentados de forma superficial pois não constituem o objetivo deste trabalho.

## B.1 O Teorema de Pierre Fermat

Fermat nasceu em 1601 em Beaumont-de-Lomagne, França. Pouco se sabe sobre sua infância, exceto que ele era o filho de um próspero comerciante e o beneficiário de uma educação secundária clássica. Em 1631, Fermat foi agraciado com o grau de bacharel de leis civis pela Universidade de Orleans e estabeleceu sua prática em Toulouse, perto de sua terra natal. Nesse mesmo ano, casou-se com a prima da sua mãe. Fermat e sua esposa, Louise, tornaram-se os pais de cinco filhos. Um católico devoto, Fermat realizou um exercício de funções jurisdicional dentro da Igreja e serviu como parlementaire em Toulouse. A segurança financeira e boa saúde — exceto por um breve período durante o qual ele estava gravemente doente com a Praga —, enfeitaram a vida tranquila e convencional de Fermat (LARSON; EDWARDS, 2017).

A vida tranquila de Fermat lhe permitiu dedicar-se ao estudo da Matemática (para ele um *hobby*) e, em 1636, ele mostrou que o ponto crítico de uma função derivável ($f(x)$) é um ponto $x = c$ do domínio desta função no qual:

$$\left.\frac{dx}{dt}\right|_{x=c} = 0$$

Obviamente, utiliza-se neste trabalho já a notação de Leibniz. Todavia, quando Fermat anunciou seu teorema, fez uso de um simbolismo menos intuitivo - o qual não passaria de um pedante eruditismo matemático se aplicado aqui.

Pelo teorema, se $x = c$ é um ponto de extremo local para $f(x)$, a derivada de $f$ se anula e passa uma reta tangente horizontal à $y = f(x)\big|_{x=c}$. Esse ponto de extremo local pode ser **máximo** ou **mínimo** (SODRÉ, 2007).

Existem funções com um ponto em $x = c$, que não é ponto de máximo nem de mínimo local para $f(x)$, como a função $f(x) = x^n \forall n \in \Re$ definida sobre a reta, onde $x = 0$ é o ponto crítico mas esse não é um ponto de extremo (SODRÉ, 2007).

Além disso, se os pontos de extremo locais para $f(x)$ estiverem nas extremidades do domínio de $f$, as derivadas laterais de $f$ poderão existir e não serem nulas (SODRÉ, 2007).

Este teorema merece destaque neste trabalho dado ser um divisor de águas na solução dos problemas de otimização e, consequentemente, uma condição importante para se avaliar a necessidade ou não de utilizar computação probabilística para encontrar a solução de um dado problema de otimização (SODRÉ, 2007).

Além deste método simples, a Matemática apresenta os métodos do Cálculo numérico, dos quais podemos destacar os seguintes:

- Direções de Descida;
- Método da descida máxima;
- Métodos *Quasi-Newton*;
- Métodos de Segunda Ordem;
- Minimização com Restrições;
- Funções Restritivas de N variáveis por métodos indiretos;
- Multiplicador de Lagrange Aumentado (MMLA);
- Métodos para restrições com igualdades e desigualdades;

Dentre muitos outros.

# APÊNDICE C – Elementos Históricos da Computação

## C.1 A Máquina Universal de Turing

Em 1936, o matemático britânico Alan Turing (1912-1954) concebeu um dispositivo teórico como máquina de estados universal. De forma mais precisa, a máquina consiste de um **modelo abstrato de computador** restringindo-se apenas aos aspectos lógicos de seu funcionamento (memória, estados e trasições) e não à implementação física.

A Máquina de Turing não é um modelo real de computadores, de forma que se pode melhor dizer, que sob nenhum aspecto a Máquina de Turing é um modelo de **computador**, mas sim da **computação** (CHU-CARROL, 2013).

A Máquina de Turing consiste em:

- Uma "fita" abitrariamente extensível [1] em ambos sentidos, dividia em células adjascentes contendo cada uma um símbolo de um alfabeto [2].
- Um cabeçote capaz de ler símbolos da fina e nela escrevê-los.
- Um registrador de estados com estados finitos e um estado inicial.
- Uma função de transição que diz à máquina que símbolo escrever, como em qual direção deve mover o cabeçote.

Esta função de transição é chamada configuração da Máquina de Turing.

O funcionamento de uma Máquina de Turing se inicia num símbolo dado por inicial e segue a seqüência de passos seguinte:

- Observar o símbolo na fita em cima do qual o cabeçote está posicionado.
- Escrever um símbolo nesta célula. Se a célula estiver em branco ou não for da parte a ser processada, assinalar com símbolo "branco".

[1] Tão extensível quanto necessário for para a computação

[2] Alfabeto é um conjunto **finito** de símbolos ou caracteres que contém um caracter nulo, sendo que o conjunto vazio e o conjunto infinito não podem ser constituídos alfabeto (DIVERIO; MENEZES, 2011).

- Mover o cabeçote numa dada direção.
- Mudar o estado corrente;
- De acordo com a configuração, desempenhar uma ação.
- Caso não haja configuração prevista, ou a célula corrente esteja com estado final ativo, encerrar a máquina.

## C.2 O Teste de Turing

Em seu artigo [3], Turing propõe originalmente considerar a questão se as máquinas podem "pensar". Todavia, reconhece ser o temo "pensar" demasiado ambíguo [4] e muda a pergunta para "Há como imaginar um computador digital que faria bem o 'jogo da imitação"'? (TURING, 1950)

O teste consiste em um local colocar-se separadamente uma máquina que responda perguntas, um ser humano para responder as mesmas perguntas e outro ser humano para fazer as perguntas e julgar as respostas. Se a máquina conseguir confundir o juiz a ponto deste não saber distinguir se fora o ser humano ou ela a responder, então diz-se que a máquina passou no teste. É importante notar que não é necessário as respostas estarem corretas, apenas que o padrão de respostas faça o juiz não conseguir distinguir entre a máquina ou o ser humano (TURING, 1950).

## C.3 Otimização Global

A otimização global é um ramo da Matemática Aplicada e da Análise Numérica que lida com a otimização de um conjunto de funções que possuem determinados critérios. Basicamente resume-se a encontrar os máximos e mínimos globais ao longo dos valores fornecindos na entrada, ao invés dos máximos e mínimos locais.

[3] Computing Machinery and Intelligence
[4] Melhor seria "prepotente".

# APÊNDICE D – Discussão filosófica a respeito da fundamentação dos Algoritmos Genéticos e a Origem das Espécies

> "Em Ciência da Computação, nós reconhecemos o princípio algorítimico descrito por Darwin - a acumulação de pequenas mudanças através da variação randômica e seleção - como o escalar montanhas, mais especificamente o escalar montanhas da imitação randômica. Todavia, nós também reconhecemmos que o escalar de montanhas é a forma de organização mais simples de todas e sabemos que só funciona bem em uma limitada classe de problemas" (WATSON, 2006).

## D.1 Introdução à discussão

A Teoria dos Algoritmos Genéticos, para o seu surgimento, assim como todas as estratégias evolucionárias, tiveram - e ainda hoje têm - sua inspiração procedural no evolucionismo darwinista.

Em torno de 1850, Charles Darwin fez uma viagem a bordo do navio HMS Beagle na qual visitou vários lugares e observou que uma espécie de mechilões era ligeiramente diferente de outros animais em outros ecossistemas. Chegando ao fim desta viagem, começou a compilar suas anotações em seu livro "A Origem das Espécies" que fora publicado em 1859. Este livro traz a famigerada Teoria da Evolução das espécies e fora rejeitado e amplamente combatido pela sociedade acadêmica de sua época, dado que não respondia aos padrões científicos e violava de forma radical a crença da comunidade acadêmica de então a respeito do tema.

Isso mudou ao longo dos anos ao passo que movimentos acadêmicos evolucionistas foram surgindo, ganhando apoio principalmente das filosofias materialistas que levantaram a bandeira do naturalismo.

O naturalismo é uma doutrina filosófica que, negando a existência das esferas transcendentes ou metafísicas, integra as realidades anímicas, espirituais ou forças criadoras no interior da natureza, concebendo-as redutíveis nos termos das leis e fenômenos do mundo. E nada mais naturalista que o darwinismo! Darwin era um naturalista.

A Teoria da Evolução diz que na natureza todos os indivíduos dentro de um ecossistema competem entre si por recursos limitados. Aqueles indivíduos que obtem êxito armazenam em sua

estrutura biológica (através de mutações) características genéticas que passam aos seus descendentes. Já os que falham nesta interação com o meio, ou morrem, ou não tem essas características perpetuadas para a espécie. Este processo é denominado **seleção natural** (DARWIN, 1995).

Em decorrência disso, a combinação entre as características dos indivíduos que sobrevivem pode produzir um novo indivíduo adaptado às características positivas de cada um dos reprodutores, gerando descendentes mais aptos. Esta "**evolução natural**" é dirigida com o intuito de maximizar alguma característica da espécie.

Estes dois são os pontos fundamentais da teoria darwinista: por um lado a seleção natural, que promoveria a sobrevivência do mais apto ao meio (o que nada mais é que a velha filosofia do determinismo numa roupagem cartesiana) e, por outro, a mutação que promoveria não apenas as diversidades raciais quanto a diversidade das espécies (eis aí a grande proposta de Charles Darwin).

Com o crescimento da aceitação social pelo naturalismo, em grande parte dado pelos movimentos filosóficos que o sucederam, e pelo poder político-social que adiquiriram, o próprio conceito de ciência mudou e o darwinismo acabou por fechar como teoria uma lacuna para a qual o natrualismo não tinha nem mais longínqua resposta.

A Filosofia já refuta de forma clara há centenas de séculos o determinismo sendo um dos maiores nomes, sob o qual não há nenhuma refutação, Agostinho de Hipona (DOLÇ, 1989) (WEBER, 1998) [1]. Logo, a seleção natural surge não como uma teoria propriamente válida, mas como uma proposta que ignora as questões levantadas por Agostinho [2]. Simplesmente as desconsidera ao passo que se auto-afirma. Seleção natural nada mais é que determinismo biológico, ou seja, o velho determinismo filosófico na roupagem cartesiana.

Embora este trabalho rejeite de forma crassa a diferenciação entre Ciência e Filosofia, considerando tal fato um dos maiores erros da história (NOUGUÉ, 2018), primeiramente considera haver grandes diferenças entre a "ciência moderna" (de bases cartesianas) e a Ciência propriamente dita (ou Filosofia) e propõe **através do limitado modelo cartesiano** apresentar alguns indícios de sua falha, e evidências que, num modelo verdadeiramente científico (ou filosófico), [3] corroboram com certos elementos de doutrinas anti-darwinistas, sendo atualmente a principal delas a doutrina do *Design Inteligente* (BEHE, 2006). Contudo, não se pretende neste trabalho, demonstrar a inveracidade, ou impossibilidade do modelo darwinista (o que em boa parte já o faz a teoria do *Design Inteligente*), nem tampouco apresentar um tratado completo que justifique a Teoria do *Design Inteligente* - o que fugiria primeiramente ao escopo deste trabalho e também às competências tanto da Matemática e mais ainda da Computação - mas sim acrescentar aos trabalhos de Behe (BEHE,

[1] Santo Agostinho de Hipona.
[2] Assim como por toda Escolástica.
[3] Ou seja, que não é apenas experimentalista e cartesiano, mas como havia antes desta divisão, promovia uma investigação completa, com todas as ferramentas do intelecto humano, incluindo os experimentos.

2006) alguns elementos que a Matemática e a Computação lhe podem contribuir, mesmo através de uma implementação simples [4] de nosso caso de estudo.

De fato, do ponto de vista cartesiano é míster afirmar (sob suas premissas) que o evolucionismo é a teoria mais completa e que de fato oferece muitas explicações sobre a origem das espécies, embora ainda não seja unânime que ele seja suficiente para explicar esta origem completamente.

Todavia, torna-se aqui necessário apresentar tais premissas, ou ao menos, a mais importante delas: na "ciência" moderna, tal qual é chamada, é inaceitável que, frente à ausência de fatos empíricos, faça-se uma suposição com implicações metafísicas. Em outras palavras, a ciência moderna é necessariamente naturalista, ou seja, rejeita tudo aquilo que é imaterial ou que não possa ser explicado como consequência da matéria e da energia. Esta é uma premissa dogmática, princípio constituinte da dúvida metódica cartesiana.

Assim, não se pretende aqui ganhar-se para as teorias anti-darwinistas um suspiro cartesiano, como vem sendo a tentativa dos defensores do design-inteligente, dado que este modelo de fato implica numa contradição ao cartesianismo, mas outrossim, utilizá-las juntamente ao próprio cartesianismo, para apresentar uma pequena contribuição contra sua ifalseabilidade dogmática e ajudando a demonstrar que o modelo cartesiano não pode ser *per si* chamado "Ciência".

Neste sentido, não se poderia dar outro título a esta discussão que não "Filosófica", e também por este sentido coloca-se num Apêndice e não no corpo do trabalho de forma que fique bem claro ser um complemento ao trabalho e não parte constituinte da mesma argumentação.

Por outro lado é de suma importância desde já levantar que este Apêndice não irá de forma alguma ir para além da Física [5]. Isso se ressalta para elucidar que este trabalho não pretende dar saltos nas categorias das ciências [6], mas tão somente permanecer na Lógica (através da arte-ciência da Computação e da Matemática) e na Biologia (através da exposição de alguns pontos triviais).

Não se pressupõe de forma alguma, pois, a existência de Deus ou qualquer entidade não-física. Tal pressuposição na Física é (e aí há uma concordância com o cartesianismo) absurda [7].

[4] Se comparada às estruturas orgânicas.

[5] No seu sentido aristotélico-tomista, composta pela Cosmologia, Química, Biologia e Psicologia, supercampos nos quais pode-se englobar todo conhecimento humano sobre o mundo natural - hoje certamente com mais detalhes do que se faria quando o Trivium e o Quadrivium deixaram de ser ensinados. O que se chama hoje de Física é, apesar dos seus muitos e superevidentes avanços, um subcampo da Cosmologia aristotélico-tomista - que de fato era caduca, devido principalmente à falta de recursos.

[6] Que se iniciam na Lógica, como propedêutica; vão para a Física, que seria a própria sabedoria não houvesse o suprassensível; depois para as ciências do *agere*, ou práticas (Ética, Sociologia e Economia); depois para a Metafísica e findam-se na Teologia Sagrada (NOUGUÉ, 2018)

[7] E seria uma desonestidade intelectual, tal qual negar o suprassensível, ainda que fosse um erro menor.

## D.2 Apresentação do Design Inteligente

O conceito de *Intelligent Design* (ID) foi proposto em 1996 pelo bioquímico Miechael Behe no seu livro *Darwin's Black Box, the Biochemical Challenge to Evolution*. Behe clamou ter descoberto a prova incontestável de que a evolução darwinista não pode explicar muitas reações bioquímicas que ocorrem na célula. Levanta-se aqui os cinco principais pontos apresentados por (BEHE, 2006) a respeito do DI:

- O *design* não é místico, mas deduzido da estrutura física de um sistema.
- Todo mundo concorda que os sistemas tem um design (mesmo quem é contra o *design inteligente*)
- Existem obstáculos estruturais sobre a evolução darwinista.
- Grandes afirmações darwinistas se apoiam em uma imaginação descontrolada, dado que este não pode explicar as estruturas celulares.
- Forte evidência a favor do design inteligente e quase nenhuma a favor do darwinismo.

## D.3 Questões importantes e questões não-importantes

Grande parte das questões levantadas contra a proposta de Behe são irrelevantes e infelizmente ocupam a atenção da maioria dos envolvidos no debate sobre o *Design* (ou Projeto) Inteligente.

São irrelevantes questões como: ser ou não criacionismo e ser ou não Ciência [8].

Palavras como "*dangerous*" (perigoso) são frequentemente usadas contra o ID, o que é irônico dado que palavras semelhantes eram igualmente usadas contra a evolução darwiniana. Criacionistas levantam o incontestável ponto de que a lei darwiniana de "sobrevivência do mais forte" fora usada pelos nazistas para justificar o holocausto e sua expansão militar.

Kenneth Miller diz que Behe levanta um "argument from designer" (argumento do Projetista) que consiste numa tentativa pseudo-científica (ou melhor, pseudo-cartesiana) para provar a existência de Deus devido à complexidade do mundo natural. De fato, o *Design Inteligente* não se impõe como uma prova e como tal já fora refutado pois, embora extremamente improvável [9], é

[8] O que, para tratar, seria necessário mais que um pequeno Apêndice. E maiores interessados poderão aprender mais em (NOUGUÉ, 2018)

[9] E, consequentemente, não é sequer verossível e menos ainda verdadeiro.

possível que tais estruturas tenham se organizado desta forma ao longo dos séculos e, por permanecer no mecanicismo, o *Intelligent Design* conseguiu apenas o *status* de verossímil, uma vez que a realidade vai além das fronteiras do mecanicismo [10]. Todavia, tal condenação não se aplica às teses de Behe dado que ele assume não que as complexidades de uma forma geral necessitem da ação de um "Designer" (sobre o qual ele não faz nenhuma inferência, reconhecendo honestamente que é dever de outras ciências [11] como a Filosofia e a Teologia).

E sobre a evidência de tal *Design* concordam inclusive alguns dos mais famosos nomes do evolucionismo:

> Biologia é o estudo das coisas complexas que têm a aparência de ter um **design intencional**. (DAWKINS, 1986)

> Podemos dizer que um corpo ou órgão vivo tem um bom **design** quando possui atributos que um engenheiro inteligente e capaz teria inserido nele a fim que cumprisse algum propósito significativo, como voar, nadar, ver... [Mas] qualquer engenheiro é capaz de reconhecer um objeto que tenha sido estruturado (mesmo se mal estruturado) para um propósito determinado e até deduzir a natureza desse propósito a partir da organização do objeto. (DAWKINS, 1986)

> A seleção natural é o relojoeiro cego, cego porque não prevê, não planeja consequências, não tem propósito nem vista. Mas os resultados vivos da seleção natural nos deixam pasmos porque parecem ter sido estruturados por um relojoeiro magistral, dando a ilusão de um **desígnio** e **planejamento**. (DAWKINS, 1986)

Até mesmo um darwinista tão famoso quanto Richard Dawkins (provavelmente o quarto nome na área), acredita que exista uma **aparência de *design***, mas não **crêem** que essa aparência é real. Mas será esta uma crença baseada tão somente em fatos observados, ou uma crença que rompe a fronteira da Física e invade às Ciências do Ágere e, quiçá as mais superiores? Ou seria apenas uma superstição, uma crença cega e irracional que dá a determinado ente potências e características que não lhe pertencem? Seriam o darwinismo e o cartesianismo apenas superstições elevadas ao *status* de Ciência?

Em seu livro, *Darwin's Black Box* (BEHE, 2006), Behe demonstra a existência do que chamou **complexidade irredutível**, que, em poucas palavras trata-se de um sistema complexo

[10] Por esse motivo, não é resposta cabal. Esta, somente a Teologia poderá dar.
[11] Agora o termo fora usado no contexto correto: aquilo que busca conhecer e explicar as coisas

que não pode ser reduzido a sistemas mais simples formados através de mutação sem perder suas funcionalidades [12], o que representa uma grande dor de cabeça para o darwinismo:

> Se fosse possível demonstrar a existência de algum órgão complexo que não pudesse, de maneira alguma, ser formado através de modificações ligeiras, sucessivas e numerosas, minha teoria ruiria inteiramente por terra. Só que jamais consegui encontrar esse órgão. (DARWIN, 1995)

Ao se falar em questões importantes, sem dúvida alguma, a mais importante delas é a Metafísica. O que da Origem das Espécies deve-se propriamente a Charles Darwin são duas coisas: a primeira, uma conclusão obvia pela Genética de que os seres se modificam ao longo do tempo, o que a Ciência Moderna chama de "microevolução" [13]; em segundo, a pressuposição naturalista de que as **espécies** evoluem, ou seja a pressuposição da **especialização**.

Contra a primeira seria absurdo negar, porém se Darwin teve nela algum mérito este, sem dúvida foi o de ter estudado um pouco da Genética de Mendel. Já a segunda, é a que este trabalho vem firmar-se contrário levantando evidências e justificativas.

Para demonstrar efetivamente que a especialização é possível, seria antes necessário comprovar que os entes podem operar para além das potências radicadas em suas formas, de forma que, com isso fosse igualmente alterado o que, na ordem do ser os distingue como circunscritos numa espécie. Sem isso a Teoria da Evolução se posiciona tal qual quimera (SILVEIRA, 2012).

E é justamente este o ponto de condenação cabal do darwinismo.

## D.4 Em relação às observações feitas nos modelos algorítimicos

Abandonando momentaneamente a questão metafísica exposta, certamente que inferir com tão pouco [14] uma falseabilidade ao modelo darwinista é não menos desonesto que negar a Metafísica, ou a Teologia baseado em trabalhos da "ciência moderna". Entretanto, não se pode negar as questões que modelos tão simples já levantam às premissas darwinistas. Tais evidências matemáticas servem não para provar, mas para objetar, utilizando o próprio cartesianismo no processo, que a evolução é ao menos muitíssimo improvável.

[12] Em outras palavras, um sistema consiste de várias partes que devem trabalhar juntas para cumprir uma determinada função.
[13] Termo este que pode perfeitamente ser adotado na Ciência, ou seja, na Filosofia.
[14] "'Pouco"' aqui não se trata das objeções feitas no parágrafo último da seção anterior, mas das observações feitas nos modelos algoritmicos, para não cairmos no experimentalismo mecanicista. Pouco, porque exige a Física (onde se encontra a Biologia), e a Metafísica (onde se estuda o ente).

Como visto amplamente nos capítulos 3 e 4 todos os modelos computacionais baseados na evolução darwinista assumem algumas premissas inteligentes, que são deveres do criador do algoritmo:

- É assumida uma complexidade irredutível em todo algoritmo genético. Em nosso caso de estudo, tal complexidade seria o próprio Indivíduo, que é o transistor CMOS. Embora, aplicando o modelo de Orr D.5 possa-se logicamente reduzí-la às suas propriedades e supor que estas propriedades possam naturalmente mutar-se e agregar-se pelos processos evolucionários a fim de convergir resultando em indivíduos que se apliquem ao modelo (o qual conteria a inteligência infusa), ainda sim é míster salientar que tal possibilidade, embora matemática, não se valida na realidade, o que se confirma pela necessidade da classe **IndividualConditions**, a qual se criou para conformar ao máximo o modelo ao processo darwinista, como fosse o meio ambiente. Todavia é claro que nenhum meio ambiente teria em si propriedades dos indivíduos que os habitam. Tal modelo, pois, é sempre forçoso.

- O espaço de busca deve ser limitado por **possíveis** soluções. Essa possibilidade deve ser bem apresentada nas funções de seleção como **limites bem definidos**. Embora Linden afirme (e com razão) que um algortimo simples como o da roleta viciada (Ver Subseção3.7.1) seja capaz de representar a distribuição da população de forma mais natural, permitindo também que indivíduos menos aptos se reproduzam, nos seus exemplos [15] ele deixa bem claro que tal processo necessida de um intervalo definido. Se o espaço de busca tender à dimensões infinitas [16], o algoritmo não converge.

$$conv = \lim_{n \mapsto \pm \inf} \frac{1}{E(x_1, x_2, ..., x_n)} = 0 \tag{D.1}$$

Onde $n$ é o número de dimensões do espaço de busca $E$. Veja-se que, no algoritmo, assume-se certa aleatoriedade na formação da população inicial, embora se assuma que métodos inteligentes também sejam possíveis. A prática da área, como afirma Linden [17] é usar a estratégia mais simples - que é a escolha randômica devido ao fato desta gerar uma boa distribuição das soluções no espaço de busca e a mutação gerar diversidade suficiente para

[15] (LINDEN, 2012), p. 76
[16] Tiver limite inferior $-\inf$ e limite superior $+\inf$, o que ocorre no modelo darwinista, já que não há inteligência que possa definí-lo.
[17] (LINDEN, 2012), p.71.

garantir uma boa exploração. Todavia, como dito acima, o próprio espaço de busca fora devida e previamente limitado pelo projetista.

- Toda seleção é feita baseada na função de *fitness* que descreve a **engenharia** [18] do modelo que se deve otimizar. Essa engenharia deve ser [19] pelo *projetista*. Como afirma Linden (LINDEN, 2012), ela "calcula então um valor numérico que reflete quão bem os parâmetros representados no cromossomo resolvem o problema". Se certa aleatoriedade, pois, pudesse ser assumida na geração da população inicial (sem causar problema de convergência), neste ponto ela já necessitaria ser **avaliada** por um algoritmo, uma função constituinte do ente em questão que o caracterizasse. Linden ainda dirá que tanto melhor será o algoritmo genético quanto melhor for esse modelo expresso pela função *fitness*. Pergunta-se, portanto, de onde vem a *fitness* que rege os compostos orgânicos e, consequentemente, os seres vivos? Pois tão claras são algumas destas funções que são estudadas no Ensino Fundamental. Outras tão obscuras que nem toda tecnologia atual foi capaz de desvendar.

- A recombinação genética compartilha entre os pais, seguindo uma lei infusa (ou processo infuso) no algoritmo, características dos pais. Aqui o modelo orientado a objetos nos é bastante útil. Classes são entidades lógicas de objeto, distintas umas das outras pelas suas características e pelos procedimentos que são capazes de executar (os métodos), sem os quais: 1) Deixam de ser o que são; 2) Não são mais capazes de desempenhar suas funções. Pode-se fazer uma aproximação entre a classe na Computação e o ente filosófico, assim como - à complexidade irredutível deste paradigma computacional. Da mesma forma que na natureza, não existem evidências - a não ser forçosas, logo não-naturais e inviávies [20], também no modelo Computacional, o cruzamento de um indivíduo com outro será capaz de gerar apenas outro indivíduo, a menos que técnicas externas (como a criação de *Wrappers*) force outro comportamento. Todavia, seja na realidade, seja na Computação, ainda os métodos anti-naturais fazem uso da inteligência de projetistas (sejam Biólogos Geneticistas, sejam Cientistas da Computação) para criar anomalias ontológicas.

- O processo de mutação apresenta o fascinante mecanismo através do qual o algoritmo genético aponta para a diversidade populacional, possibilitando a conversão gênica. Aparentemente aqui dá-se a grande certeza do darwinismo por boa parte da chamada "Comunidade Científica". Todavia, parafraseando Richard Dawkins, pode-se dizer que a mutação dá-se a impressionante idéia de que, do caos, pode-se gerar a ordem e esta ordem pode ser perpetuada misteriosamente

[18] Seu projeto. Pode-se dizer perfeitamente aqui *design*.
[19] desenvolvida
[20] Dado que todas as tentativas feitas em laboratório geraram descendêntes defeituosos e inférteis, fadados à extinção natural em um ou poucas gerações (como no caso da bactéria e-coli) - de que uma dada espécie de indivíduos tenha se vertido n'outra por processos de recombinação e mutação (ou quaisquer outros).

por indivíduos distintos que nunca tiveram contato com aquele primeiro a se ordenar, por uma estranha aleatoriedade.

Enquanto o professor Richard Watson, evolucionista, apresentou em seu trabalho (WATSON, 2006) que as mutações não podem explicar a complexidade dos mecanismos biológicos e tenha oferecido um mecanismo (validado pela Ciência da Computação, num trabalho semelhante a este [21]) não-darwiniano, suas teorias foram desconsideradas pela "Comunidade Científica" [22]. Ainda que Watson tenha mostrado na simulação de computador que esta idéia de alcançar o pico é uma idéia que não funciona.

Não se nega aqui, todavia nem a existência das mutações (o que seria um absurdo), nem tampouco que, em alguns casos ela possa ser benéfica. As grandes evidências pró-evolucionismo são duas: as pesquisas feitas com a malária, tanto nas mudanças genéticas dos humanos contaminados, quanto dos próprios parasitas (*plasmodium falciparum*) e os 30 anos de pesquisa de Richard Lenski com a cultura de bactérias e-coli, no qual mais de 50.000 gerações, e agora um amplo investimento em tecnologia não o fizeram ir mais longe que as pesquisas feitas com a malária.

Segundo Behe (BEHE, 2006) observou, mais de 99% das mutações que afetam os indivíduos onde ocorrem [23] são detrimentais. Das que são benéficas (o 1% restante) quebram genes ou degradam sua função, ou seja, causam danos no sistema que modificam.

Assim sendo, se a seleção necessita de um projetista que lhe apresente os parâmetros e principalmente as regras de *fitness*, se os entes não se especializam naturalmente independente dos processos que sofram e do tempo, e se as mutações são em vasta maioria detrimentais, que resta ao darwinismo? Os fósseis?

> "'O conhecimento do registo fóssil expandiu-se muito nestes 120 anos depois de Darwin.... Ironicamente, nós temos hoje menos exemplos de transições evolucionarias do que tinhamos no tempo de Charles Darwin (RAUP, 1979)"'.

Esses são os indícios que os modelos computacionais evolucionários dão contra a própria teoria que os inspirou. Perguntas estas que não são respondidas pelo darwinismo e que, juntamente às diversas propostas sérias relacionadas ao *Intelligent Design* põem em cheque não apenas a teoria em si, quanto seus fundamentos: o materialismo, o naturalismo e o cartesinaismo. Não porém negam os avanços que vieram decorrente da quebra de preconceitos caducos por conta dessas cor-

[21] Embora de maior amplitude e complexidade na época.
[22] Não por apresentarem-se erradas, mas simplesmente não foram divulgadas e deixadas ao esquecimento.
[23] Pois há aquelas (mutações neutras) que não afetam, e que constituem a maioria das mutações (BEHE, 2006), (MASATOSHI, 2005), (MORGAN, 2009).

rentes filosóficas, mas clamam à chamada "Ciência Moderna" que rompa com certos dogmatismos naturalistas assumidos no Iluminismo, para que volte a compreender-se como parte de um todo [24].

## D.5 A refutação de Orr ao *Intelligent Desing*

A pressuposição inicial do *Inelligent Design* é de que é **impossível** (BEHE, 2006) que um sistema complexo seja construído sem elementos que sejam irredutíveis em sua complexidade.

> "Por um complexo irredutível, quero dizer, um único sisema composto por várias partes que combinam bem e que contribuem para a função básica, em que a remoção de qualquer uma das partes faz com que o sistema pare efetivamente de funcionar". (BEHE, 2006)

Para sua refutação, Orr diz que, num passado distante haviam sistemas simples, constituídos de apenas uma parte A, e que não funcinavam muito bem. Mutações genéticas ao longo de eras produziram uma parte B anexa a A de forma que o sistema AB fosse adaptado ao meio. De fato este sistema, concorda Orr, não seria irredutivelmente complexo, pois AB funcionaria ainda sem B. Todavia, uma segunda mutação genética ocorre no sistema AB, desta vez em A, produzindo A*. Este novo sistema A*B já não mais poderia funcionar sem B, logo seria irredutivelmente complexo. Com isso, ele assume ter mostrado ser possível que sistemas irredutivelmente complexos, poderiam ser reduzidos a um processo evolutivo no mais ortodoxo darwinismo, sendo considerada a refutação definitiva até o presente momento em que este trabalho é escrito, para a teoria da Complexidade Irredutível de Behe.

Vê-se que, de fato, Orr usa de uma lógica silogista muito simples para refutar Behe.

Todavia falta-se fazer à proposta de Orr, algumas perguntas:

- Dado que nunca foi encontrado na natureza, nem tampouco fabricada em laboratório, nenhuma parte A, simples, que sofresse tal processo, e que, muito pelo contrário, todos os experimentos se degeneraram as amostras a partir de um ponto complexo irredutível e irreversível (AVIEZER, 2010) pode-se afirmar que, embora haja a possibilidade, esta não condiza com a realidade observada?
- Dado não existir nenhuma evidência de A, poderia ser A um complexo irredutível ao invés de uma parte unifuncional? Ou, em outras palavras, de que modo A poderia desenvolver-se em B? Já que não se pode perguntar de que modo A verte-se em B na natureza, pois nunca foi encontrada tal estrutura A.

[24] A Filosofia, ou Ciência por excelência.

- Quando A produz B por mutação e, posteriormente, produz A*B, A*B*, (A*B*)*, onde cada $*$ implica em um processo de mutação, isso de fato demonstra ser possível o processo evolutivo, mas não que ele é provável, pois não há sequer um único caso observável deste tipo de agregação seguida de mutações sucessivas de coisas totalmente distintas resultando numa outra distinta de ambas [25]. Reafirmo, não se demonstra provável, mas possível. Todavia ainda que provável fosse o evolucionismo estaria explicando, apenas parte do processo da evolução biológica dos seres, dado que o indivíduo $(A * B*)^{n*}$ não se verteu noutro totalmente distinto de $A * B*$ mantendo em si suas características ontológicas [26].

- Talvez o mais importante fator relativo ao processo evolutivo é aquele já observado na pória lei de seleção e mutação darwinista e, por conseguinte, também no *design* inteligente: há uma "inteligência" infusa na formação dos seres: seja esta inteligência oriunda do processo adaptativo da seleção natural que, mesmo num dado momento, percebe ter atingido um pico na curva da evolução, seja na complexidade irredutível.

Mais a respeito da discussão científica entre Behe e Orr pode ser acompanhado no compilado feito por Richard Johns do departamento de filosofia da *The University of British Columbia* (JOHNS, 2012). Entretando, não é absurdo assumir racionalmente que tanto o darwinismo, quanto principalmente as objeções de Orr sejam meras falácias.

## D.6 Brevíssimas objeções metafísicas a respeito da evolução darwinista

Primeiramente, cabe-se aqui explicar o porquê de tal seção: ela se justifica e faz-se necessária dado que a Metafísica, justamente por ser *filosofia primeira* (como ensina Aristóteles), fornecer a todas as demais ciências os princípios sem os quais elas sequer poderiam ser chamadas propriamente de "ciências". Desta forma, a Metafísica pode e deve imiscuir-se nos problemas de todas as demais ciências, quado estas contrariam os princípios indemonstráveis dela, que lhes servem de esteio. E tal "intromissão" [27] lhe cabe de direito, em virtude da universalidade de seu objeto formal que é o superior a todas as demais (a exceção da Teologia): o ente enquanto ente.

Todo ente é composto por substância e forma. A substância se divide em duas: primeira e segunda. Na substância primeira estão contidas a essência, que é o que responde à questão "'*Quid est*"' [28] e obtém em resposta a quididade, o próprio ente; e os acidentes. A substância segunda refere-se aos universais abstraídos dos indivíduos, ou seja, obtidas deles pelo intelecto. A substância

[25] Que seria o processo de especialização, o qual depende de se encontrar os elos perdidos.
[26] Que fazem dele ser o que é.
[27] As aspas se devem ao fato de não ser de fato uma intromissão, uma vez que toda ciência - a exceção da Teologia - é subconjunto da Metafísica, e ela não faz mais um senhor ao acessar os porões da própria casa quando entra nas demais ciências e as julga, pois lhe é superior.
[28] Verbo **êsse**.

constitui o substrato pelo qual o sujeito (matéria para Aristóteles) se constitui em algo organizado segundo uma forma. Para que se possa, entretanto, atribuir algo a um sujeito é necessária existência de predicados (ou categorias) que dizem o ser de vários modos.

Assim sendo, em todo ente material (composto de matéria e forma), a forma é o ato primeiro de organização da matéria [29](sendo-lhe fundamental) e a operação o ato segundo (SILVEIRA, 2012).

Outra característica da forma é o **princípio de especificação**. E isto por uma razão simples: é pela forma que um ente se enquadra numa espécie. Todavia, espécie aqui distingue-se daquela da ciência biológica. Esta classificação a priori já vê-se perturbada ao ponto de não poder-se justificar-se de forma universal sequer entre os próprios biólogos (embora amplamente aceita), menos ainda para as ciências elevadas. *Ipso facto*, algumas das especializações da biologia se justificam na Metafísica, por se diferenciarem quanto ao *modus operandi*, já outras são meras diferenciações na matéria. Na Metafísica, espécie é o que radica na forma. Não se trata, pois, de classificar os entes quanto a meros comportamentos, mas algo que lhe é anterior: o estatuto ontológico das espécies que pressupõem todos os seus predicamentos (SILVEIRA, 2012).

A especialização darwinista, como visto ao longo deste trabalho, assume que a matéria possa violar primeiramente a forma (através de determinações do ambiente) [30] e, através de sucessivas mutações, de ações externas e por um longo período de tempo [31], mudar a própria quididade.

Fugindo das questões epistemológicas (e de outras) [32]. , notório se faz que, para não sucumbir ante a primeira objeção filosófica, a hipótese da evolução precisaria provar antes de tudo a possibilidade da *forma entis* transcender as potências que a circunscrevem (SILVEIRA, 2012) [33].

As premissas de especialização darwinistas são, portanto, desde o início, uma confusão na própria compreensão da matéria dado que, somente um ente sem composição de matéria em sua forma poderia não ser limitado por ela em seu ser e operar. No darwinismo, a matéria é tomada como realidade ativa ao invés de passiva. A suposição de que as operações gênicas aplicadas

[29] A matéria é o que, em sentido lato, está em potência tanto para o ser substancial (*materia ex qua*) quanto para o ser acidental (*materia in qua*), sendo a primeira um dos coprincípios da **natureza** (junto à forma e à privação - meramente potenciais), que demarca os limites de seu modo de operar na ordem do ser (*operatio sequitur esse*)

[30] Como visto, o velho determinismo filosófico em uma roupagem cartesiana.

[31] Novamente o determinismo.

[32] Sequer vamos tratar do erro epistemológico das óbvias inversão de limites e violação das barreiras do universo que circunscreve cada ciência, mas vale salientar que há um sério problema quando uma ciência não é aparelhada sequer para conhecer a origem da série causal em seu próprio âmbito de investigação ser utilizada para arrogar a uma hipótese o estatuto de verdade universal. Quando cada um dos subconjuntos da Ciência Moderna tenta fazer tal pressuposição viola suas próprias fronteiras decretando a própria falência.

Como ensina Duns Scotus, quando, ainda que sob pretexto de investigar uma suposta origem da vida, a Biologia viola suas fronteiras, já não mais pode-se dizer Ciência Natural, mas sim arroga-se características metafísicas, forjando-se numa pseudo-metafísica, numa má-metafísica, o que estaria introduzindo um *argumentum ad absurdum* dado que, fundamentada no materialismo e no naturalismo, caminha pelos jardins da Metafísica para pressupor algo puramente metafísico.

[33] Dado que, na falácia de Orr, deria necessário provar que A pode desenvolver B sem que seja uma complexidade irredutível, de forma que AB fosse outro ente toalmene distinto de A

sucessivamente por um período longo de tempo é capaz de tornar um ente n'outro totalmente diverso, tal qual defende Orr, traz consigo a premissa oculta de que a substância segunda, e a matéria, têm potência ativa infinita - o que não passa de uma deificação da matéria *O darwinismo é a derrocada materialista do cartesianismo.*

Coisa uma é pressupor as evoluções dentro de uma própria espécie (através dos mecanismos que forem), outra bem distinta todavia, e inaceitável na Metafísica (que rege a biologia conquanto lhe é superior), é pressupor que uma espécie supere as inalienáveis contigências metafísicas em que está arrojada.

A chamada microevolução, entretanto, não é impecílio para a Metafísica. Todavia, a especialização sim. Negar a especialização signfica negar que somos frutos de um processo evolutivo de um descendente comum.

## D.7 Algumas Conclusões

As conclusões que se chegam aqui, depois de Orr e também pelas observações feitas em ambos os lados desta disputa são:

1. Por um lado Teoria da Evolução não é falseável no âmbito da Lógica, o que se deriva de Orr. Nem tampouco possa-se afirmar de forma contundente que todo sistema complexo se origina de sistemas complexos irredutíveis, neste âmbito da Lógica.

2. Dado que a Ciência desconhece tanto os elos perdidos quanto também os seres mais simples, tendo apenas hipóteses a respeito de sua forma, também é impossível negar que tais seres possam, ao invés de simples, terem uma complexidade irredutível, a qual possam ter se combinado para desencadear o processo evolucionário, caso tenha ocorrido.

3. Nem a Teoria da Evolução, nem a Teoria do Design Inteligente servem para explicar *perfeitamente* a origem das espécies no mundo, embora ambas forneçam elementos complementares para esta compreensão. Tais elementos complementares podem ser observados inclusive em sistemas verdadeiramente simples como o caso de estudo deste trabalho, onde de fato os processos evolucionários darwinistas levam a pontos de mínimo local e os mecanismos da evolução, aliados à inteligência do projetista trabalham juntos para atingir sistemas melhores, sem todavia conseguir-se resolver os saltos lógicos do evolucionismo, nem as brechas do *design* inteligente.

4. As filosofias naturalistas e materialistas que perpassam pela mente de muitos evolucionistas, e as mecanicistas e humanistas, que passam pela mente dos adeptos de muitos defensores do *Intelligent Design*, os quais usam para justificar ou a implicação direta da existência de Deus

a partir das meras observações naturais, ou sua negação, são em última instância falaciosas vez que a Física - e menos ainda a Biologia, que lhe é ainda inferior - não podem mais que fornecer algumas evidências para a Metafísica (a qual somente ela, com suas ferramentas, consegue inferir a respeito de tais temas, os quais só poderão ser mais perfeitamente tratados na Teologia), menos ainda poderia levar o homem, por seus próprios meios a **provar** Deus e compreendê-lo em sua plenitude. Toda prova cabal por pressuposições puramente racionais pressupõe ser o ente que faz a prova maior que o que ele observa. Sem isso é possível provar tão somente caminhar nas dimensões do intelecto passo-a-passo, sem tocar na certeza: julgamento autônomo (opinião), estado de dúvida, conjecturação do possível, verossimilhança, probabilidade. O evolucionismo tange o estado da dúvida [34], enquanto o *intelligent design*, desconsiderando as premissas dogmaticas do cartesianismo (mas fazendo uso de sua metodologia experimental) e assumindo o paradigma mecanicista tange a esfera da verossimilhança. Todavia, certeza, só se pode tê-la na Metafísica.

5. A respeito da cientificidade das Teorias, em última análise, nem Darwinismo (ou o Cartesianismo no qual ele pretende se solidificar) nem o *Intelligent Design* são propriamente científicas, mas se por um lado o darwinismo é cartesiano, por outro o *Intelligent Design* transcende o cartesianismo aproximando-se dos limites da Física, ainda que mecanicista.

6. É inegável que os modelos computacionais evolutivos, juntamente com os modelos computacionais de inteligência artificial tem sido muito úteis neste debate e tais contribuições trazem para a lógica matemática - a mais simples subdivisão da Lógica possível - assuntos que antes só se era possível aproximar a compreensão após muitos anos de estudos complexos. As abstrações computacionais popularizaram relativamente tais assuntos. Todavia, esta popularização provocou certa vulgarização dos temas e consquente violação de fronteiras, gerando por vezes, teorias absurdas.

[34] Sendo, como afirmara João Paulo II (*Último Papa canonizado.* mais que uma mera hipótese.

# Referências


AL., A. B. et. *Molecular Biology of the Cell*. [S.l.: s.n.], 2002.

ALAM, M. Particle swarm optimization: Algorithm and its codes in matlab. In: . [S.l.: s.n.], 2016.

ARORA, S. Polynomial time approximation schemes for euclidean traveling salesman and other geometric problems. *J. ACM*, ACM, New York, NY, USA, v. 45, n. 5, p. 753–782, set. 1998. ISSN 0004-5411. Disponível em: <http://doi.acm.org/10.1145/290179.290180>.

ARROYO, J. E. C. *Heurísticas e metaheurísticas para otimizaćão combinatória multiobjetivo.* Dissertação (Mestrado) — Universidade Estadual de Campinas, Campinas, 2002.

AVIEZER, N. Intelligent design versus evolution. *Rambam Maimonides Medical Journal, 1(1), e0007*, 2010. Disponível em: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3721655/>.

BARR, N. *Economics of the Welfare State*. 4. ed. Oxford University Press, 2004. Disponível em: <https://EconPapers.repec.org/RePEc:oxp:obooks:9780199264971>.

BARRA, S. et al. Multi-objective genetic algorithm of cmos operational amplifiers. In: . [S.l.: s.n.], 2012.

BEHE, M. J. *Darwin's Black Box: the biochemical challenge to evolution*. [S.l.]: Free Press, 2006.

BENI, G.; WANG, J. Swarm intelligence in cellular robotic systems. *Robots and Biological Systems: Towards a New Bionics?*, 1993. Series F: Computer and System Sciences, vol 102. Springer, Berlin, Heidelberg.

CHU-CARROL, M. C. [S.l.]: PragmaticBookshelf, 2013.

DARWIN, C. *The Origin of Species*. Gramercy, 1995. Hardcover. ISBN 0517123207. Disponível em: <http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/0517123207>.

DAS, S.; ABRAHAM, A.; KONAR, A. Particle swarm optimization and differential evolution algorithms: Technical analysis, applications and hybridization perspectives. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

DAWKINS, R. *The blind watchmaker*. [S.l.]: Penguin, 1986.

DIVERIO, T. A.; MENEZES, P. B. *Teoria da Computação: máquinas universais e computabilidade*. [S.l.]: bookman, 2011. ISBN 978-85-7780-831-1.

DOLÇ, M. *Confessions*. Proa, 1989. (Clàssics del cristianisme). ISBN 9788477391036. Disponível em: <https://books.google.com.br/books?id=b0jtRQAACAAJ>.

EIBEN, A. E.; SMITH, J. E. *Introduction to Evolutionary Computing*. 2nd. ed. [S.l.]: Springer Publishing Company, Incorporated, 2015. ISBN 3662448734, 9783662448731.

FERREIRA, J. ao C. *Circuitos CMOS: Um resumo*. 2004. Projetos de Circuitos VLSI. Aula.

FILGUEIRAS, I. F. R. da S. *Otimização de circuitos CMOS por algorítimo genético*. Dissertação (Mestrado) — Universidade de São Paulo, 2010.

FONSECA, C. M.; FLEMING, P. J. Genetic algorithms for multiobjective optimization: Formulationdiscussion and generalization. In: *Proceedings of the 5th International Conference on Genetic Algorithms*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993. p. 416–423. ISBN 1-55860-299-2. Disponível em: <http://dl.acm.org/citation.cfm?id=645513.657757>.

FOWLER, M. et al. *Agile Manifesto*. 2001. Disponível em: <http://agilemanifesto.org>.

HANCOCK, H. *Theory of Maxima and Minima*. [S.l.]: University of Cincinnati, 1917.

INSTITUTE, L. *Lean Principles*. 2018. Disponível em: <https://www.lean.org/WhatsLean/Principles.cfm>.

JARAMILLO, J. H.; BHADURY, J.; BATTA, R. On the use of genetic algorithms to solve location problems. *Comput. Oper. Res.*, Elsevier Science Ltd., Oxford, UK, UK, v. 29, n. 6, p. 761–779, maio 2002. ISSN 0305-0548. Disponível em: <http://dx.doi.org/10.1016/S0305-0548(01)00021-1>.

JOHNS, R. *Design Arguments - Behe vs. Orr*. 2012. Acessada em 20 de janeiro de 2018. "http://faculty.arts.ubc.ca/rjohns/design.pdf".

KACHITVICHYANUKUL, V. Comparison of three evolutionary algorithms: Ga, pso and de. v. 11, n. 3, p. 215–233, 2012. ISSN 1598-7248.

KENNEDY, J.; EBERHART, R. Particle swarm optimization. *Proceedings of IEEE International Conference on Neural Networks*, 1995.

KENNEY, J. M. et al. *Literacy Strategies for Improving Mathematics Instructions*. [S.l.: s.n.], 2005. I.

KOFUJI, P. D. S. T.; ZUFFO, P. D. J. ao A.; SOARES, P. D. J. ao N. *Circuitos Integrados CMOS*. 2004.

KOZA, J. R. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. 6th. ed. [S.l.]: Massachusetts Institute of Technology, 1992. ISBN 0-262-11170-5.

LARSON, R.; EDWARDS, B. *Pierre de Fermat*. 2017.

LINDEN, R. *Algoritimos Genéticos*. [S.l.: s.n.], 2012. ISBN 978-85-399-0195-1.

MACHADO, A. *"Descartes e a Psicologia da Dúvida"*. 2018. "<http://www.olavodecarvalho.org/descartes-e-a-psicologia-da-duvida/>".

MARTIN, R. C. *Clean Code Video Series*. 1. ed. Upper Saddle River, NJ, USA: Addison Wesley Professional, 2017. Safari Books. ISBN 0134843800.

MASATOSHI, N. electionism and neutralism in molecular evolution. *Molecular biology and evolution.*, 2005. Disponível em: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1513187/>.

MIRANDA, V.; FONSECA, N. Epso-evolutionary particle swarm optimization, a new algorithm with applications in power systems. In: *IEEE/PES Transmission and Distribution Conference and Exhibition*. [S.l.: s.n.], 2002. v. 2, p. 745–750 vol.2.

MONTFORT, A. ao C. *"As Três Revolućões na Arte"*. 2018. "<http://www.montfort.org.br/bra/cadernos/arte/3revolucoes/",26/02/2018.>

MONTFORT, A. ao C. *"As Três Revolućões na Arte"*. 2018. "<http://www.montfort.org.br/bra/cadernos/religiao/iluminismo/>".

MORGAN, L. A. Human y chromosome mutation rates. 2009. Acessado a 14 de agosto de 2017. Disponível em: <http://sandwalk.blogspot.com.br/2009/08/human-y-chromosome-mutation-rates.html>.

NICOLETT, A. S. *Processo de Fabricaćão de Circuitos Integrados*. 2017. Curso de Engenharia Elétrica. Aula.

NOUGUÉ, C. *Do Papa Hertico*. [S.l.]: Edićões Santo Tomás, 2017. v. 1.

NOUGUÉ, C. *A Ciência Moderna à luz do Tomismo*. 2018. Palestra conferida em 28 de agosto de 2017, pelo Centro Dom Bosco em Curitiba. Pode ser acessada em "https://www.youtube.com/watch?v=ejpJ0NC8JEA".

ONWUBOLU, G. C.; DAVENDRA, D. *Differential Evolution: A Handbook for Global Permutation-Based Combinatorial Optimization*. [S.l.]: Springer-Verlag Berlin Heidelberg, 2009. ISSN 1860949X. ISBN 9783-540-92151-6.

PANTUZA, G. *Métodos de Otimizaćão Multiobjetivo e de Simulaćão Aplicados ao Problema de Planejamento Operacional de Lavra em Minas a Céu Aberto*. Dissertação (Mestrado) — Universidade Federal de Ouro Preto, Ouro Preto, 2011.

PARETO, V. *Manual of political economy (manuale di economia politica)*. New York: Kelley, 1971 (1906). xii, 504 p. p. Translated by Ann S. Schwier and Alfred N. Page.

PELETIER, M. *Newton's Problem of the Body of Minimal Resistance*. 2010.

PEREIRA, L. S. *Dopagem Eletrônica*. Online. Acessado em 13/11/2017. Https://www.infoescola.com/quimica/dopagem-eletronica/.

PIMENTA, T. C. *Processo de Fabricaćão de CI*. 2015. Curso de Microeletrônica, PRPPG - Mestrado em Ciência e Tecnologias da Informaćão. Aula.

POPPER, K. *A Sociedade Aberta e seus Inimigos*. [S.l.: s.n.], 1874. II.

RABAEY, J.; CHANDRAKASAN, A.; NIKOLIC, B. *Digital Integrated Circuits*. 2004.

RAUP, D. Conflicts between darwin and paleontology. Field Museum of Natural History Bulletin, v. 50, 1979.

SC.D., T. L. H. C. *The Thirteen Books of Euclid's Elements*. [S.l.]: Cambridge University Press Warehouse, 1908.

SEDRA, A.; ROBERTS, G.; SMITH, K. *SPICE for Microelectronic Circuits*. 3rd. ed. [S.l.]: Saunders College Pub., 1992. ISBN 0-03-052617-5.

SEDRA, A. S.; SMITH, K. C. *Microelectronic Circuits*. fifth. [S.l.]: Oxford University Press, 2004.

SILVEIRA, S. *A metafísica contra a Teoria da Evolução*. 2012. <http://contraimpugnantes.blogspot.com.br/2012/02/metafisica-contra-teoria-da-evolucao.html>.

SODINI, C. G. *Microelectronics Devices and Circuits*. 2017. Acessado em http://web.mit.edu/6.012/www à 8 de dezembro de 2017.

SODRÉ, U. Matemática essencial: Ensino fundamental, médio e superior. In: . [S.l.: s.n.], 2007.

STORN, R.; PRICE, K. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, v. 11, 1997. ISSN 1573-2916.

TANG, P.-H.; TSENG, M.-H. Adaptive directed mutation for real-coded genetic algorithms. *Applied Soft Computing*, 2013. ISSN 1568-4946.

Thomas Aquinas. *Summa theologiae*: pars prima, a quaestione i ad quaestionem il. [S.l.]: Typographia Polyglotta. v. 4.

TURING, A. Computing machinery and intelligence. 1950.

VIRGILIUS. *The Aeneada*. [S.l.]: Classics MIT, 19 a.C.

VOGT, H.; HENDRIX, M.; NENZI, P. *Ngspice User Manual*. 2018. <http://ngspice.sourceforge.net/>.

WATSON, R. A. Compositional evolution: The impact of sex, symbiosis and modularity on the gradualist framework of evolution. 2006.

WEBER, D. *Augustinus: De Genesi Contra Manichaeos*. Academiae Scientiarum Austriacae, 1998. (Corpus scriptorum ecclesiasticorum Latinorum). ISBN 9783700127130. Disponível em: <https://www.augustinus.it/latino/genesi_dcm/index2.htm>.

WEISSTEIN, E. W. Dido's problem. *MathWorld*, 2017. Http://mathworld.wolfram.com/DidosProblem.html.

WESTE, N. H. E.; HARRIS, D. M. *CMOS VLSI Design: A Circuits and Systems Perspective*. [S.l.]: Pearson, 2011. ISBN 78-0-321-54774-3.

WHITLEY, D. The genitor algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. In: . [S.l.]: Morgan Kaufmann, 1989.

WIKIPEDIA. Secretary problem. *en.wikipedia.org*, 09 2017.

XIA, W.-j.; WU, Z.-m. A hybrid particle swarm optimization approach for the job-shop scheduling problem. *The International Journal of Advanced Manufacturing Technology*, v. 29, n. 3, p. 360–366, Jun 2006. ISSN 1433-3015. Disponível em: <https://doi.org/10.1007/s00170-005-2513-4>.

YILMAZ, Ö. F.; DURMUŞOĞLU, M. B. *An Integrated Mehtodology for Ordere Release and Scheduling in Hybrid Manufacturing Systems*. [S.l.]: IGI Global, 2018. Applied Optimization Methodologies in Manufacturing System.

ZEBULUM, R. S.; PACHECO, M. A.; VELLASCO, M. Synthesis of cmos operational amplifiers through genetic algorithms. *SBCCI*, 1998.

ZEGHBROECK, B. V. *Principles of Semiconductor Devices*. [S.l.: s.n.], 2011. Http://ece-www.colorado.edu/ bart/book/.

ZENG, M. jia; CHENG, Z. lin. A method using genetic algorithm to optimize neural networks applied in sustainable development ability appraisal. *International Conference on Electronic Computer Technology*, 2009.

ZHANG, H.; GEN, M. Multistage-based genetic algorithm for flexible job-shop scheduling problem. *Journal of Complexity International*, 2005.