Algoritmos evolutivos como estimadores de
frequência e fase de sinais elétricos:
métodos multiobjetivos e paralelização em
FPGAs

Tiago Vieira da Silva

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura:

Algoritmos evolutivos como estimadores de frequência e fase de sinais elétricos: métodos multiobjetivos e paralelização em FPGAs

Tiago Vieira da Silva

Orientador: *Prof. Dr. Alexandre Cláudio Botazzo Delbem*

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação - ICMC-USP, como parte dos requisitos para obtenção do título de Mestre em Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

USP - São Carlos
Novembro/2013

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados fornecidos pelo(a) autor(a)

V586a Vieira da Silva, Tiago
Algoritmos evolutivos como estimadores de
frequência e fase de sinais elétricos: métodos
multiobjetivos e paralelização em FPGAs / Tiago
Vieira da Silva; orientador Alexandre Cláudio
Botazzo Delbem. -- São Carlos, 2013.
88 p.

Dissertação (Mestrado - Programa de Pós-Graduação
em Ciências de Computação e Matemática
Computacional) -- Instituto de Ciências Matemáticas
e de Computação, Universidade de São Paulo, 2013.

1. Algoritmos evolutivos. 2. Algoritmos
multiobjetivos. 3. FPGA. 4. Estimção de frequência.
5. Sinal elétrico. I. Botazzo Delbem, Alexandre
Cláudio, orient. II. Título.

Dedico aos meus pais.

Agradecimentos

Agradeço a Deus por ter criado os algoritmos evolutivos e as garotas ruivas.

Agradeço aos meus pais por tudo que fizeram por mim.

Agradeço ao meu orientador pela orientação e pela paciência dos dias que eu desaparecia para escalar o Monte Fuji e não dava notícias.

Agradeço aos meus amigos pela companhia e pela diversão.

Resumo

Este trabalho propõe o desenvolvimento de Algoritmos Evolutivos (AEs) para estimação dos parâmetros que modelam sinais elétricos (frequência, fase e amplitude) em tempo-real. A abordagem proposta deve ser robusta a ruídos e harmônicos em sinais distorcidos, por exemplo devido à presença de faltas na rede elétrica. AEs mostram vantagens para lidar com tais tipos de sinais. Por outro lado, esses algoritmos quando implementados em *software* não possibilitam respostas em tempo-real para uso da estimação como relé de frequência ou Unidade de Medição Fasorial. O desenvolvimento em FPGA apresentado nesse trabalho torna possível paralelizar o cálculo da estimação em *hardware*, viabilizando AEs para análise de sinal elétrico em tempo real. Além disso, mostra-se que AEs multiobjetivos podem extrair informações não evidentes das três fases do sistema e estimar os parâmetros adequadamente mesmo em casos em que as estimativas por fase diverjam entre si. Em outras palavras, as duas principais contribuições computacionais são: a paralelização do AE em *hardware* por meio de seu desenvolvimento em um circuito de FPGA otimizado a nível de operações lógicas básicas e a modelagem multiobjetiva do problema possibilitando análises dos sinais de cada fase, tanto independentemente quanto de forma agregada. Resultados experimentais mostram superioridade do método proposto em relação ao estimador baseado em transformada de Fourier para determinação de frequência e fase.

Abstract

This work proposes the development of Evolutionary Algorithms (EAs) for the estimation of the basic parameters from electrical signals (frequency, phase and amplitude) in real time. The proposed approach must be robust to noise and harmonics in signals distorted, for example, due to the presence of faults in the electrical network. EAs show advantages for dealing with these types of signals. On the other hand, these algorithms when implemented in software can't produce real-time responses in order to use their estimations as frequency relay or Phasor Measurement Unit. The approach developed on FPGA proposed in this work parallelizes in hardware the process of estimation, enabling analyses of electrical signals in real time. Furthermore, it is shown that multi-objective EAs can extract non-evident information from the three phases of the system and properly estimate parameters even when the phase estimates diverge from each other. This research proposes: the parallelization of an EA in hardware through its design on FPGA circuit optimized at level of basic logic operations and the modeling of the problem enabling multi-objective analyses of the signals from each phase in both independent and aggregate ways. Experimental results show the superiority of the proposed method compared to an estimator based on Fourier transform for determining frequency and phase.

Lista de Figuras

1.1	Representação gráfica de sinal trifásico.	2
1.2	Sistema Integrado Nacional.	3
2.1	Amostras de sinal.	12
3.1	Representação de um indivíduo com valores binários.	16
3.2	Cruzamento por corte.	18
3.3	Fronteira de pareto.	19
3.4	Criação de indivíduo no AEMT.	21
3.5	Inserção de novo indivíduo na tabela.	21
3.6	Modelo de programação paralela: a) Troca de mensagem b) Memória compartilhada.	23
4.1	Relação entre desempenho e flexibilidade.	29
4.2	Diagrama de componentes de um FPGA.	30
4.3	Componentes de um bloco lógico.	31
4.4	Disposição dos pinos no bloco lógico.	31
4.5	Chave de interconexão.	32
4.6	Diagrama de iteração do algoritmo Cordic.	34
5.1	Fluxograma do algoritmo.	40
5.2	Processamento paralelo.	40
5.3	Geração do indivíduo.	41
5.4	Cruzamento por característica.	42
5.5	Blocos principais do relé.	47
5.6	Máquina de estados finita do AG.	48
6.1	População do AG: (a) Geração 1 (b) Geração 20.	54
6.2	Variação da amplitude simulada.	55
6.3	Variação da frequência simulada.	56
6.4	Variação do ângulo de fase simulada.	56
6.5	Variação da amplitude em VHDL.	57
6.6	Variação da frequência em VHDL.	58
6.7	Variação do ângulo de fase em VHDL.	58
6.8	AG Software x AG Hardware convergindo para o valor real.	59
6.9	Janela com 8 amostras de tempo.	60
6.10	Janela com 15 amostras de tempo.	61

6.11	Janela com 8 amostras de tempo e memória do resultado anterior.	62
6.12	Janela com 15 amostras de tempo e memória do resultado anterior.	63
6.13	Modificação na representação da amplitude.	64
6.14	Modificação na representação da frequência.	65
6.15	Comparação entre o AG e um relé comercial.	66
6.16	Comparação entre o AG e um relé comercial ampliada.	66
6.17	Sinal regularizado.	67
6.18	Sinal regularizado para amostras de 16 <i>bits</i>	68
6.19	Resultado para o sistema trifásico.	69
6.20	Caso B: Resultado hipotético para o sistema trifásico utilizando oráculo.	70
6.21	AG Multipopulacional simples.	71
6.22	AG Multipopulacional com amplitude ponderada.	72
6.23	AG com função objetivo composta.	73
6.24	AG com múltiplas tabelas seguindo o modelo AEMT.	74
6.25	AEMT com amplitude ponderada.	75
6.26	TVE para variação de fase.	77
6.27	TVE para variação de amplitude.	77
6.28	TVE para variação de frequência.	78

Lista de Tabelas

5.1	Codificação de parâmetros proposta inicialmente no projeto.	42
5.2	Codificação final de parâmetros.	45
5.3	Requisitos mínimos do AG.	45
5.4	Características dos FPGAs.	46
6.1	Parâmetros testados.	54
6.2	Tempo de recuperação dos algoritmos	76
6.3	Tempo de computação do AE e AEMT para $1 \leq M \leq 3$ and $1 \leq T - M \leq 2$	79

Lista de Abreviaturas e Siglas

ADC	<i>Analog-to-Digital Converter</i>
AE	Algoritmo Evolutivo
AEMO	Algoritmo Evolutivo Multiobjetivo
AEMT	Algoritmo Evolutivo Multiobjetivo por Tabelas
AG	Algoritmo Genético
AGMO	Algoritmo Genético Multiobjetivo
ATP	<i>Alternative Transients Program</i>
ASIC	<i>Application-Specific Integrated Circuit</i>
CAG	Controle Automático de Geração
CMOS	<i>Complementary Metal-Oxide-Semiconductor</i>
CORDIC	<i>COordinate Rotation DIgital Computer</i>
CPLD	<i>Complex Programmable Logic Device</i>
DSP	<i>Digital Signal Processor</i>
DFT	<i>Discrete Fourier Transform</i>
EEPROM	<i>Electrically Erasable Programmable Read-Only Memory</i>
EPROM	<i>Erasable Programmable Read Only Memory</i>
FDP	Função de Distribuição de Probabilidade
FFT	<i>Fast Fourier Transformer</i>
FPGA	<i>Field-Programmable Gate Array</i>
IEEE	<i>Institute of Electrical and Electronics Engineers</i>
LFC	<i>Load Frequency Control</i>
LFSR	<i>Linear Feedback Shift Register</i>
MIMD	<i>Multiple Instruction Multiple Data</i>
MPI	<i>Message Passing Interface</i>
ONS	Operador Nacional do Sistema Elétrico
PLD	<i>Programmable Logic Device</i>

PMU	<i>Phasor Measurement Unit</i>
PVM	<i>Parallel Virtual Machine</i>
SDE	Sistemas de Distribuição de Energia elétrica
SEP	Sistema Elétrico de Potência
SIMD	<i>Single Instruction Multiple Data</i>
SRAM	<i>Static Random-Access Memory</i>
TVE	<i>Total Vector Error</i>
TW	<i>Transformada Wavelet</i>
VHDL	<i>VHSIC Hardware Description Language</i>
VHSIC	<i>Very High Speed Integrated Circuit</i>

Sumário

Resumo	i
Abstract	iii
Lista de Figuras	vi
Lista de Tabelas	vii
Lista de Abreviaturas e Siglas	ix
1 Introdução	1
1.1 Sistemas Elétricos de Potência	1
1.1.1 Operação do sistema	3
1.1.2 Tratamento de falhas	4
1.1.3 Estimação de frequência e fase	4
1.2 Objetivo	5
1.3 Organização	5
2 Estimação de Parâmetros de Sinais Elétricos	7
2.1 Apresentação do problema	7
2.2 Métodos de estimação de frequência	8
2.2.1 Transformada Discreta de Fourier	9
2.2.2 Método dos Mínimos Quadrados	10
2.2.3 Filtros Ortogonais	10
2.2.4 Transformada Wavelet	10
2.2.5 Redes Neurais Artificiais	11
2.3 Medição de Amplitude e Fase	11
2.4 Estimação por Algoritmos Evolutivos	12
2.5 Considerações	13
3 Algoritmos Evolutivos	15
3.1 Estrutura de um Algoritmo Genético	15
3.1.1 Indivíduos	16
3.1.2 Função de avaliação	16
3.2 Operadores genéticos	17

3.3	Problemas de Otimização Multiobjetivo	18
3.3.1	Abordagens de resolução	19
3.3.2	Algoritmos Evolutivos Multiobjetivos	20
3.3.3	Algoritmo Evolutivo Multiobjetivo por Tabelas	21
3.4	Aumento de Eficiência de Algoritmos Evolutivos por paralelização	22
3.4.1	Fundamentos de métodos de paralelização	22
3.4.2	Algoritmos Evolutivos Paralelos	25
3.4.3	Aspectos de paralelização em hardware	26
3.5	Considerações	27
4	Hardware Reconfigurável	29
4.1	<i>Field-Programmable Gate Array</i>	30
4.2	Gerador de números aleatórios	32
4.2.1	Entrada aleatória	32
4.2.2	Tabela pré-calculada de números aleatórios	33
4.2.3	LFSR - <i>Linear Feedback Shift Register</i>	33
4.3	Funções trigonométricas	33
4.3.1	Polinômio de Taylor	33
4.3.2	Valores tabelados de seno	34
4.3.3	Cordic	34
4.4	Linguagem de descrição de <i>hardware</i>	35
4.5	Considerações	35
5	Desenvolvimento da proposta de estimador de Amplitude, Frequência e Fase	37
5.1	Estrutura de dados	37
5.1.1	Entrada de dados	38
5.1.2	Representação dos indivíduos	38
5.2	Projeto de Algoritmo Genético paralelizado em FPGA	39
5.2.1	Operadores utilizados	41
5.2.2	Função objetivo	42
5.3	Paralelização em FPGA	45
5.3.1	Componentes	46
5.3.2	Processos	47
5.4	Método de Regularização	48
5.5	Algoritmos Evolutivos Multiobjetivos Implementados	49
5.5.1	Relé usando Método de Agregação de Funções	50
5.5.2	Relé usando o AEMT	50
5.6	Considerações	51
6	Resultados Experimentais	53
6.1	Avaliação de janela independente	53
6.1.1	Resultados em <i>software</i>	55
6.1.2	Resultados em <i>hardware</i>	57
6.2	Avaliação em tempo real	59
6.3	Efeito da Regularização	67
6.4	Contribuição de Modelos dos Multiobjetivos	68
6.4.1	Heurística de Troca das Fases	71
6.4.2	AG Multiobjetivo por Função Agregação	72
6.4.3	AEMT para Relé de Frequência	74

6.5	TVE usando o AEMT	76
6.6	Eficiência computacional do AE e AEMT	78
6.7	Considerações	79
7	Conclusão	81

Introdução

Sistemas Elétricos de Potência - SEP são sistemas de energia que englobam a geração, transmissão e distribuição de eletricidade. Com o aumento contínuo no tamanho e complexidade deles surgem problemas de sincronia e falhas de fornecimento de energia. É necessário o acompanhamento da frequência de operação para coibir falhas no sistema e, caso ocorram, detectá-las rapidamente, para que o fornecimento possa ser restaurado sem grandes danos.

A correta estimação de frequência e fase é importante para manter sistemas sincronizados e evitar falhas de distribuição (Kundur et al., 1994). Um relé deve estimar com precisão a frequência e comutar o circuito em caso de problemas com o sinal recebido. Dessa forma, o sistema deve possibilitar um monitoramento constante da qualidade do sinal, viabilizando a detecção de onde e como os problemas e falhas na rede elétrica ocorrem. A estimação da fase do sinal em determinado instante de tempo, permite detectar interações entre sinais vindos de diferentes pontos em um sistema de distribuição.

1.1 Sistemas Elétricos de Potência

Um Sistema Elétrico de Potência, SEP, é formado por equipamentos de geração, transmissão e distribuição de energia elétrica. Há várias pesquisas sobre problemas relacionados ao sistema trifásico, bem como à conversão entre corrente alternada e corrente contínua, além de pesquisas voltadas a campos de estudo específicos, como a distribuição de carga, ou garantia de fornecimento (Bollen, 2000).

O sistema trifásico é uma forma de transmissão de corrente alternada composta por três tensões sincronizadas, defasadas em $1/3$ de ciclo. Dessa forma o sistema se mantém balanceado e a transmissão é mais eficiente do que seria se fossem três sistemas isolados (Santoso et al., 2002).

Figura 1.1: Representação gráfica de sinal trifásico.

Um SEP é uma rede de dispositivos conectados que vão desde a produção até o consumo final de energia elétrica. É comum haver subestações que aumentam a amplitude do sinal, para que este seja enviado através de longas distâncias, da mesma forma que outras subestações convertem o sinal para uma tensão menor que será usada pelo consumidor final. Os componentes do SEP são divididos, conforme suas funções em: Geração, Transmissão e Distribuição (Kundur et al., 1994).

Geração

Geração de eletricidade é o processo em que energia obtida de uma fonte externa é transformada em energia elétrica. Existem diversos meios de transformação podendo ser química, fotovoltaica e a mais comum, eletromecânica. A conversão eletromecânica baseia-se em obter energia cinética, através de uma turbina, e converter o movimento em eletricidade através de um gerador. O sinal trifásico é criado em sincronia com o campo magnético gerado pela turbina, assim a frequência do sinal é proporcional à frequência de rotação. Esse é o método mais usado devido a sua eficiência na conversão de energia.

Transmissão

A eletricidade deve ser transportada da estação de geração até as unidades de distribuição. Para isso ela passa por um complexo sistema de transmissão, que inicia com transformadores aumentando a tensão, o que facilita a distribuição em longas distâncias. Por exemplo, o caso do Sistema Integrado Nacional conecta os pontos de geração aos locais de consumo de energia no Brasil conforme mostra a Figura 1.2, fornecida pelo ONS (Operador Nacional do Sistema Elétrico).

Figura 1.2: Sistema Integrado Nacional.

Linhas de transmissão podem se estender por quilômetros. Sendo sujeitas a fatores externos, é necessário que exista redundância na distribuição, o que cria uma complexa malha para transmitir o sinal. Ao longo da rede de transmissão, existem diversas subestações responsáveis por gerenciar as ligações entre diferentes redes, monitorar o sinal e alterar a tensão conforme a necessidade do sistema (Bollen, 2000).

Distribuição

O sistema de distribuição é responsável por levar a energia do sistema de transmissão para o consumidor final. O equipamento inclui transformadores conectados ao sistema de transmissão, para reduzir a tensão, linhas de distribuição dos transformadores para os consumidores e sistemas de proteção e controle do equipamento entre o consumidor e o transformador. O equipamento de proteção inclui para-raios, disjuntores e fusíveis. O equipamento de controle inclui reguladores de tensão, capacitores, relés e gerenciadores de demanda.

1.1.1 Operação do sistema

Em um SEP, as quedas de frequência resultantes, por exemplo, da perda parcial da geração, não podem ser toleradas por longo tempo, pois podem danificar o gerador. No caso de geradores acionados por turbinas a vapor, se a frequência cai abaixo de 5% do valor nominal, corre-se o risco de quebra das palhetas devido à rotação na faixa de ressonância mecânica (Stevenson, 1982).

Tendo em vista o tamanho que os SEPs desenvolveram e as situações transitórias pelas quais eles passam, é preciso constante monitoramento para garantir qualidade, confiabilidade e continuidade na distribuição da energia elétrica (Lee, 2008). Equipamentos de monitoração avaliam constantemente a situação da rede de transmissão, enviando dados para centrais de processamento. Estes dados pode ser enviados diretamente via rede elétrica em forma de sinal digital, formando assim uma Rede Elétrica Inteligente, em inglês *Smart Power Grid*. Com base nos dados é possível desativar linhas com sinal ruído ou controlar a carga dos geradores. É possível também detectar e reestabelecer quedas de energia locais evitando grandes períodos sem suprimento de energia elétrica (Chen et al., 2009).

1.1.2 Tratamento de falhas

Inúmeras interferências podem ocorrer em um SEP, é possível citar o acionamento indevido de relés, mau funcionamento de equipamentos sensíveis e distorções em equipamentos de medição, podendo chegar até interrupção do fornecimento de energia. Equipamentos e maquinários são sensíveis a variações de carga, uma vez que possuem controles microprocessados que podem queimar com picos de tensão. Essas características os tornam mais sensíveis a variações na qualidade do sinal, podendo resultar em falhas de operação (Emanuel et al., 1991).

Em sistemas cada vez mais integrados, perturbações no sinal causadas pela partida de motores e utilização de sistemas eletrônicos passam a ser fatores que interferem na estabilidade do sinal. As perturbações mais comuns podem ser divididas em quatro tipos (Phadke & Thorp, 1988):

- Perturbações na amplitude,
- Na frequência,
- Na forma de onda ou
- No desequilíbrio da corrente em sistemas trifásicos.

Para avaliar a estabilidade do sistema têm sido mais comumente utilizados a amplitude e frequência do sinal. A amplitude está associada à forma de onda, à simetria do sistema e à tensão aceitável para a operação. A frequência está diretamente associada à velocidade de rotação dos geradores (Fuchs et al., 2008).

1.1.3 Estimação de frequência e fase

Em um sistema em equilíbrio, a frequência do sinal de corrente alternada, deveria permanecer constante, porém este está sujeito a variações de carga no sistema. Com o aumento e diminuição de demanda em diferentes pontos da rede, a mudança na rotação de motores e geradores altera a frequência do sinal. Assim a variação de frequência age como um indicador de desequilíbrio no sistema, uma vez que ela deveria ser constante em todo o SEP.

A fase representa o estado da onda em determinado instante de tempo. Através dela é possível calcular o comportamento da soma de dois sinais distintos, prevendo como o sinal se comportará em uma rede interligada. Uma Rede Elétrica Inteligente com sensores espalhados em diversos pontos poderia monitorar, prever e controlar o comportamento do sinal elétrico. Com isso seria possível obter maior rendimento na transmissão da energia gerada com menos perdas devido a falhas (Christie & Bose, 1996).

Algoritmos Evolutivos (AEs) são adequados a tratar esse tipo de problema com múltiplas informações dos vários sensores da Rede Elétrica Inteligente. Além disso, os AEs apresentam grande tolerância a ruídos, produzindo estimações relevantes mesmo quando os dados de entrada são significativamente distorcidos. No caso do SEP, um dado distorcido é na verdade um sinal defeituoso devido a problemas na rede. Outra vantagem dos AEs para processamento de sinais elétricos é que eles possuem uma estrutura adequada para serem paralelizados, permitindo aumentar a eficiência do AE a um ponto de garantir resultados em tempo real, tornando-os adequados para o desenvolvimento das Redes Elétricas Inteligentes (Blickle, 1997).

1.2 Objetivo

Este projeto busca desenvolver, como prova de conceito, um estimador de frequência e fase através de um Algoritmo Evolutivo projetado de forma paralela em um FPGA (*Field Programmable Gate Array* - Arranjo de Portas Programável em Campo) (Bobda, 2007).

Para aumento da eficiência de AEs, além da paralelização em FPGA, são investigadas técnicas de regularização da variação de frequência (Golub et al., 1999), (Monaco et al., 2009) (Seção 5.4), assim como o uso de Algoritmos Evolutivos Multiobjetivos (AEMOs), para modelar o sistema trifásico (Dias & De Vasconcelos, 2002) e melhor aproveitar as informações medidas das três fases para o processo de estimação.

O sucesso dessas técnicas, no problema de processamento do sinal elétrico, aumenta a eficiência da estimação, permitindo um tempo menor de processamento e também a simplificação do hardware projetado por código VHDL. Dessa forma, desenvolve-se um novo relé computacionalmente eficiente e robusto, uma vez que é capaz de estimar corretamente a partir de sinais significativamente distorcidos.

1.3 Organização

Os capítulos seguintes tratam das áreas de conhecimento abordadas e do desenvolvimento do projeto em si. No Capítulo 2 são mostradas as técnicas já utilizadas no problema de estimação de frequência em sinal elétrico. O Capítulo 3 apresenta o funcionamento dos AEs e suas variações utilizando técnicas compactas, com poucos indivíduos e multiobjetivas e aplicações paralelas. O

Capítulo 4 descreve a utilização de *hardware* reconfigurável e a linguagem de programação utilizada.

O Capítulo 5 mostra as escolhas tomadas no desenvolvimento do projeto: o tipo de dados utilizado, a modelagem e variações do algoritmo, a forma de paralelização, regularização e uso de métodos multiobjetivos. O Capítulo 6 mostra os resultados obtidos com os diversos AEs desenvolvidos até se projetar o Algoritmo Evolutivo Multiobjetivo por Tabelas (AEMT) para estimação de frequência e fase. O Capítulo 7 apresenta considerações finais sobre o projeto, destacando os objetivos atingidos.

Estimação de Parâmetros de Sinais Elétricos

Neste capítulo é feita uma revisão do problema de estimação da frequência, amplitude e ângulo de fase do sinal transmitido na rede elétrica. Na Seção 2.1, são apresentados os desafios e motivações em estimação de frequência. A Seção 2.2 descreve os métodos mais comuns para a análise do sinal. A Seção 2.3 explica a representação de amplitude e fase utilizando fasores, bem como a norma do IEEE de avaliação da estimação dessa grandeza. Na Seção 2.4, é apresentada a abordagem proposta ao problema, que será detalhada nos capítulos seguintes. A Seção 2.5 apresenta considerações finais sobre o problema apresentado.

2.1 Apresentação do problema

Para operação satisfatória de um SEP, a frequência deve permanecer aproximadamente constante. Um eficiente controle da frequência assegura constância da velocidade dos motores de indução e motores síncronos. Em uma rede de energia, um decréscimo na frequência do sistema pode resultar em altas correntes de magnetização em transformadores e motores de indução (Kundur et al., 1994).

Em um SEP, a frequência é dependente do balanço de potência ativa, isto é, a relação entre potência gerada e potência consumida. Como a frequência é um fator comum por toda a parte do sistema, uma mudança na demanda de potência ativa, em um dado ponto, é refletida em todo o conjunto através da mudança na frequência. Devido à existência de muitos geradores suprindo

energia ao SEP, devem ser providos meios para manejar a demanda de potência nos geradores conforme a necessidade.

Um regulador de velocidade em cada unidade geradora, fornece uma função de controle primário da velocidade, cujo objetivo é fazer o balanço de potência ativa do sistema, e com isso, estabilizar a frequência do sistema, quando observado um valor diferente da nominal. Desta forma, o controle primário de velocidade provoca um pequeno desvio na frequência do sistema. A correção de tal desvio é feita pelo controle secundário de velocidade, conhecido como Controle Automático de Geração (CAG) (Kundur et al., 1994).

Na ausência de um regulador de velocidade, a resposta do sistema para uma mudança de carga é determinada pela constante de inércia e pela constante *Dumping* - D (expressa como porcentagem de mudança na carga para 1% de mudança na frequência). Valores típicos de D variam de 1 a 2, um valor de $D = 2$ significa que 1% de mudança na frequência causaria 2% de mudança na carga. O desvio na velocidade, em relação ao seu estado estacionário, é tal que a mudança de carga é exatamente compensada pela variação da carga, devido à sensibilidade da frequência.

Em um sistema interligado, com duas ou mais áreas controladas de forma independente, além do controle de frequência, a geração dentro de cada área tem que ser controlada para que seja mantido o intercâmbio de energia planejado. O controle de geração e frequência é geralmente referenciado como Controle de Carga Frequência (LFC - *Load Frequency Control*) (Rerkpreedapong et al., 2003).

Em geral, a carga do SEP é uma composição de uma variedade de dispositivos elétricos. Para cargas resistivas, tais como iluminação incandescente e aquecimento, a potência elétrica independe da frequência. No caso de cargas motoras, tais como ventiladores e bombas, a potência elétrica muda com a frequência devido à mudança na velocidade do motor.

2.2 Métodos de estimação de frequência

Tipos diferentes de relés de frequência têm sido usados ao longo dos anos. Os relés de disco de indução, precursores de todos os relés de frequência, têm cedido espaço a outros tipos de relés com maior grau de precisão e versatilidade (Kundur et al., 1994).

Em virtude do desenvolvimento de dispositivos semicondutores estáticos com alto grau de confiabilidade, o projeto de relés de proteção utiliza desses componentes para obter respostas mais rápidas e exatas. Alguns relés microprocessados baseiam-se em uma técnica em que o período da forma de onda é observado, até que o próximo zero da tensão seja detectado. Dessa forma, através do período da forma de onda se obtêm a frequência da mesma.

Hoje em dia, os relés microprocessados são capazes de retirar amostras dos valores de tensão e corrente com alta velocidade, manipular estes dados, efetuar medições, informar casos de falta e fazer a sua autochecagem. Do avanço tecnológico evidenciado, obtiveram-se algumas melhoras

no sistema de proteção das redes elétricas, assim como outros benefícios. É possível citar (Phadke & Thorp, 1988):

- **Custo** - em decorrência de avanços significativos no campo computacional, os custos na obtenção de um relé digital estão cada vez mais acessíveis;
- **Autochecagem e confiabilidade** - os relés digitais podem ser programados para monitorarem seu próprio software e hardware, aumentando a confiabilidade;
- **Integração do sistema em ambiente digital** - permite a comunicação entre o relé e o sistema de gerenciamento computadorizado;
- **Flexibilidade** - como dispositivo programável, torna-se mais prático a mudança das características do relé, frente a mudanças eventuais no sistema.

O sinal digital pode ser lido diretamente ou passar antes por um filtro que vise minimizar o ruído inerente a leitura e ao próprio sinal. Das técnicas conhecidas é possível citar Transformada Discreta de Fourier, Método dos Mínimos Quadrados, Filtro de Kalman, Filtros Ortogonais. Além de técnicas iterativas, baseadas em Redes Neurais Artificiais e AEs. Nas Subseções a seguir serão listadas as técnicas de estimação conhecidas na literatura com ênfase nos métodos on-line que servirão de comparação ao método proposto neste trabalho.

2.2.1 Transformada Discreta de Fourier

Em (Girgis & Ham, 1982), é proposto um algoritmo para a detecção de cargas utilizando a Transformada Rápida de Fourier (FFT - *Fast Fourier Transformer*). Estes enfatizam que a técnica calcula o desvio da frequência observada e prevê a porcentagem da geração ou da carga, em desequilíbrio.

A Transformada Discreta de Fourier converte valores recebidos na dimensão da amplitude para valores na dimensão da frequência. Baseada na fórmula

$$f_j = \sum_{k=0}^{n-1} x_k e^{-\frac{2\pi i}{n} jk}, \quad (2.1)$$

as variáveis f_j representam os coeficientes de frequência após a aplicação da transformada. Sendo n o número de amostras e x_k a amostra obtida no instante k . O procedimento evidenciado atua apenas nas cargas excedentes, sem o comprometimento da estabilidade do processo. Possui alta precisão, a velocidade na estimação e a sua viabilidade de desenvolvimento em circuitos impressos, sendo usado como base para diferentes implementações. Em (Lobos & Rezmer, 1997) é apresentado um método com boa estimativa para formas de onda com grandes harmônicos porém o tempo de processamento fica em torno de três a quatro períodos da componente fundamental.

2.2.2 Método dos Mínimos Quadrados

Em (Sachdev & Giray, 1985), é aplicado o método dos mínimos quadrados em amostras de tensão digitalizadas, obtidas no ponto de localização do relé. O desenvolvimento matemático do algoritmo proposto considera que a frequência da rede elétrica é estimada em condições normais de operação, ou seja, são descartadas situações nas quais existe uma descaracterização da forma de onda, como por exemplo, situações de queda brusca de amplitude.

Desta forma, pode-se representar a forma de onda por:

$$v(t) = V_{max} \cdot \sin(2\pi ft + \theta), \quad (2.2)$$

em que V_{max} é o valor máximo da tensão, f é o valor da frequência e θ é o ângulo de fase. A expressão que determina a frequência do sinal é alcançada ao aplicar identidades trigonométricas conhecidas e a série de Taylor na Equação 2.2. Os autores evidenciam que os resultados obtidos nos testes são satisfatórios, embora o desempenho da técnica seja muito influenciado pelo tamanho da janela de dados, a frequência de amostragem e a aproximação realizada na expansão da série de Taylor.

2.2.3 Filtros Ortogonais

Aplicando a teoria de componentes ortogonais, os autores (Szafran & Rebizant, 1998) discorrem sobre um novo algoritmo que utiliza dois filtros ortogonais, do tipo de resposta ao impulso, para a estimação da frequência. A propriedade essencial da técnica, segundo os pesquisadores, é a excelente imunidade à amplitude das componentes ortogonais e os ganhos variáveis do filtro, os quais asseguram resultados com precisão de 1,5 a 20 MHz. Uma importante vantagem desse procedimento é a precisão, independente da taxa de amostragem do sinal de entrada, além da simplicidade e estabilidade, o que possibilita sua aplicação a equipamentos de proteção.

2.2.4 Transformada Wavelet

O autor (Lin, 2001) propôs uma nova aproximação recursiva de estimação da frequência, utilizando a Transformada Wavelet (TW). O autor evidencia que o algoritmo pode ser aplicado em uma larga faixa de desvio da frequência, com alta robustez e precisão na estimação, mesmo sobre condições dinâmicas e sinais com componentes harmônicos em corrente contínua. O que o torna apropriado ao desenvolvimento em tempo real. Ainda neste trabalho, o pesquisador faz uma breve análise das características e limitações do algoritmo de filtragem por Fourier em comparação à técnica proposta.

2.2.5 Redes Neurais Artificiais

Em (Dash et al., 1997), é apresentada uma arquitetura de rede neural artificial (RNA) para a estimação de frequência. A aproximação usa neurônios artificiais lineares adaptativos, conhecidos como Adaline, para identificação dos parâmetros que descrevem o modelo discretizado, do sinal de tensão, presente na rede elétrica. Tais variáveis são ajustadas, para forçar a diminuição do erro entre o sinal atual e o calculado com as novas amostras adquiridas, satisfazendo a equação de estabilidade da diferença, minimizando a função de erro.

2.3 Medição de Amplitude e Fase

A norma IEEE C37.118 define os requisitos para implementação de unidades de medição fasorial. Um fasor, também chamado de vetor de rotação, é uma representação bidimensional de uma onda em movimento harmônico simples:

$$x(t) = A \cdot \cos(\omega t + \phi) \quad (2.3)$$

que na representação fasorial seria dada, em coordenadas polares, em função da amplitude da onda, A , e do ângulo de fase no instante em questão, $\omega t + \phi$, para o instante $t = 0$:

$$\vec{X} = A(\cos(\phi)\vec{i} + \sin(\phi)\vec{j}) \quad (2.4)$$

também representada como:

$$\vec{X} = A e^{i\phi}. \quad (2.5)$$

A utilização de fasores para representação de amplitude e ângulo de fase, permitem a simplificação de várias operações necessárias no gerenciamento do sinal elétrico, simplificando operações com ondas senoídeais que passam a ser operações entre vetores (Martin et al., 2008).

Para verificar a qualidade da estimação de amplitude e fase, utiliza-se uma métrica chamada *Total Vector Error*, TVE, definida como

$$TVE = \frac{|\vec{X}_{est} - \vec{X}|}{|\vec{X}|}, \quad (2.6)$$

em que $\vec{X}$ é o valor real do vetor e $\vec{X}_{est}$ é o valor estimado por diferentes métodos. A Seção 6.5 apresenta resultados da estimação de fasores, utilizando o TVE, comparando os métodos desenvolvidos neste projeto com a Transformada Discreta de Fourier.

2.4 Estimação por Algoritmos Evolutivos

AEs vem sendo utilizados na estimação de frequência e fase devido sua facilidade de adaptação, eficiência e tolerância a ruídos. O grupo de pesquisa vem utilizando os AEs como ferramenta, em diversos problemas que necessitam de uma estimação de frequência rápida e confiável, podendo citar: (Macêdo et al., 2002), (Vargas et al., 2005), (de Souza et al., 2006), (Souza et al., 2007a), (Souza et al., 2007b), (Souza et al., 2008), (de Carvalho et al., 2009), (Coury et al., 2009a), (Coury et al., 2009b), (Usida et al., 2010).

Valores são obtidos com uma frequência de amostragem definida, e um conjunto de n valores, aqui chamado de janela, representando o sinal em um dado intervalo de tempo. Conforme o tempo passa, novos dados são obtidos e a janela desliza, fazendo com que os valores mais antigos sejam substituídos.

Figura 2.1: Amostras de sinal.

A cada janela, o algoritmo deverá buscar por parâmetros, que ao serem calculados gerem um sinal semelhante ao amostrado. Cabendo ao algoritmo minimizar a diferença entre o sinal obtido e o sinal gerado. Ao contrário do método dos mínimos quadrados, que usa um método determinístico para minimizar essa diferença, o AE fará uma busca sem informações do domínio em questão. Apenas minimizando a diferença, enquanto melhora a qualidade da população de indivíduos gerados.

Os AEs serão mais detalhados no Capítulo 3 e aspectos relevantes do método proposto serão discutidos no Capítulo 5. Neste contexto, esta pesquisa apresenta um método alternativo e eficiente, para medição do desvio da frequência, assim como da magnitude e do ângulo de fase para um dado sinal de tensão. Nesta abordagem, o sistema é formulado como um problema de otimização,

cujo objetivo é minimizar o erro de estimação. Cabe adiantar que uma das vantagens evidenciadas quando do emprego de AEs ao problema delineado, é que o mesmo oferece resultados satisfatórios na estimação, mesmo diante de sinais ruidosos.

Investiga-se também uma estrutura de programação para a técnica dos AEs, a qual se aproxime da estrutura requerida para o desenvolvimento em FPGA. A proposta AE + FPGA deve ser capaz de estimar satisfatoriamente a frequência, a magnitude e o ângulo de fase para uma dada forma de onda, evidenciando um tempo de processamento adequado a uma aplicação on-line.

2.5 Considerações

Neste capítulo, foram apresentados os principais pontos referentes à estimação de frequência do sinal, dando as bases matemáticas, os métodos de resolução de problemas e suas aplicações práticas. Com base nos conceitos apresentados foi possível conhecer o estado da arte da avaliação de frequência e fase para sistemas de tempo real, além de estabelecer adequada comparação do método aqui apresentado com outras soluções já presentes na literatura.

O método proposto possibilita calcular rapidamente, em uma fração do período da onda, com qualidade e garantia de retorno do valor sem atraso. O AE torna o sistema tolerante a ruídos do sinal, sendo flexível o bastante para serem efetuadas diversas melhorias, como o uso de técnicas de regularização, Seção 5.4, e abordagem multiobjetiva do sistema trifásico, Seção 5.5. O desenvolvimento em FPGA permite explorar o paralelismo implícito do AE e paralelizar o cálculo da função objetivo (Equação 5.2), que avalia cada potencial modelo de onda, viabilizando AEs para estimação em tempo real.

Algoritmos Evolutivos

Os Algoritmos Genéticos (AGs) são uma classe de Algoritmos Evolutivos (AEs). Nos AEs, deve-se otimizar uma quantidade de 'recursos' para resolver determinado problema. Os recursos são avaliados paralelamente e evoluem em populações. Os AGs seguem o mesmo princípio, mas são definidos por codificar esses recursos como sendo genes em um cromossomo. Para evoluir a população são aplicados operadores nos genes (Holland, 1975). Em geral, AGs são utilizados em problemas, os quais não se possui muita informação sobre o domínio em questão, e que o espaço de busca é muito grande, necessitando apenas de uma forma de quantificar quão boa é uma determinada solução.

Este capítulo visa apresentar os principais conceitos sobre AEs, suas variações e seus operadores. A Seção 3.1 apresenta a estrutura de funcionamento de um AG e suas bases teóricas. A Seção 3.2 mostra os principais operadores genéticos conhecidos. Na Seção 3.3, são apresentados os Algoritmos Evolutivos Multiobjetivos (AEMOs) e suas variações. A Seção 3.4 trata do uso de métodos paralelos na criação de AGs. Por fim na Seção 3.5, são feitas considerações sobre o uso de AGs e como eles são utilizados no desenvolvimento do projeto.

3.1 Estrutura de um Algoritmo Genético

Os AGs foram propostos por Holland ao estudar o processo evolutivo dos seres vivos e como a natureza se adapta para lidar com os recursos disponíveis. O processo foi adaptado para resolver problemas computacionais. Nos AGs, as possíveis soluções do problema se tornam indivíduos em uma população. Assim como na biologia, os indivíduos sofrem seleção, cruzamento e mutação, evoluindo até atingir melhores soluções (Holland, 1975).

Dessa forma, para que a população evolua, assim como no processo biológico, é necessário que exista competição por recursos, variabilidade nas soluções e que os melhores indivíduos sejam unidos para formar soluções melhores. A hereditariedade se dá pela presença de genes que atuam como parâmetros da solução, mapeando características que podem ser compartilhadas entre os pais, ou modificadas aleatoriamente.

Ao longo do processo evolutivo podem ocorrer falhas, a população pode convergir para uma solução que não represente o ótimo, ou simplesmente perder variabilidade e convergir à deriva. Para isso podem ser aplicados outros métodos, como predação dos indivíduos mais fracos, destruir a população iniciando com valores aleatórios, simulando desastres que podem ocorrer (Goldberg, 2002).

3.1.1 Indivíduos

A busca pela solução é feita simultaneamente em diversas instâncias do domínio do problema, cada instância é chamada indivíduo e armazena valores de uma possível solução. O conjunto de indivíduos em uma mesma iteração do algoritmo é chamado população. A população normalmente começa com valores aleatórios, porém podem ser inseridos indivíduos na população, gerados previamente, que possuam bons valores na função de avaliação, de modo que os mesmos conduzam a população para áreas de busca mais promissoras.

11...0	00...1	10...1
V_m	f	ϕ

Figura 3.1: Representação de um indivíduo com valores binários.

Uma representação comum dos indivíduos é feita através de vetores binários que armazenam os dados do problema. Assim um valor 0 ou 1 pode representar se determinado parâmetro pertence ou não à solução. Na Figura 3.1, são representados valores de amplitude (V_m), frequência (f), e ângulo de fase (ϕ). Em outros problemas, podem ser usados números inteiros para quantificar uma variável, ou um agrupamento de *bits* com significado próprio para o problema. A solução representada pelo indivíduo é também chamada de genótipo, pois armazena as características intrínsecas do mesmo (Haupt et al., 1998).

3.1.2 Função de avaliação

Cada indivíduo possui um valor associado que o qualifica, em comparação aos demais, como melhor ou pior dentro do problema em questão. Esse valor é calculado usando uma função de avaliação, também chamada função *fitness*, e pode ser desde uma expressão matemática relacionada a um problema, um valor obtido experimentalmente, ou também uma nota associada por um avaliador externo. O valor da função para determinado indivíduo é também chamado aptidão ou fenótipo, pois representa como o indivíduo se adaptou ao problema.

De um modo geral, a função de avaliação é o gargalo do AG, pois ela costuma necessitar a realização de um experimento, ou mesmo um tempo de processamento para que o indivíduo seja corretamente avaliado. Assim, abordagens paralelas vem sendo utilizadas para diminuir o tempo de geração dos indivíduos, além de técnicas de redução de espaço de busca, reduzindo o número de avaliações ao partir do princípio que indivíduos semelhantes possuem aptidões semelhantes (Sastry & Goldberg, 2003).

3.2 Operadores genéticos

Os operadores genéticos devem fazer com que os indivíduos de melhor aptidão se sobressaiam, gerando melhores soluções a cada geração do algoritmo. Os operadores são baseados na biologia e visam gerar competição e variabilidade na população. A seguir os operadores genéticos mais utilizados (Goldberg, 2002):

Seleção

A seleção consiste na escolha dos melhores indivíduos para continuar existindo na população. Existem diversas formas em que ela é utilizada. O método da roleta consiste em selecionar os indivíduos aleatoriamente de forma proporcional à suas respectivas aptidões, dando assim mais chances de selecionar os melhores. Na seleção por torneio dois, ou mais, indivíduos aleatórios são retirados da população e o de melhor aptidão é selecionado. A seleção por truncamento é feita ordenando os indivíduos de acordo com sua aptidão e selecionando uma quantidade fixa dos melhores indivíduos. Existe também a seleção em que o melhor indivíduo é selecionado e é usado para cruzar com toda a população.

Elitismo

O elitismo consiste em manter inalterado o melhor indivíduo da população, mantendo ele imune aos outros operadores. Dessa forma, evita-se que a melhor solução seja perdida e aumenta-se a velocidade de convergência do algoritmo, diminuindo o número de gerações necessárias.

Cruzamento

O cruzamento é usado para combinar soluções de dois ou mais indivíduos previamente selecionados. O que possibilita que boas soluções possam ser unidas, gerando soluções melhores. Quando a representação do indivíduo se dá por vetor de números binários, é comum que o cruzamento faça um corte no vetor e junte os valores dos pais gerando novos indivíduos.

Essa técnica sofre com o fato de que parâmetros que poderiam estar relacionados, podem ser separados, o que acaba 'quebrando' uma solução que estaria sendo formada. Quando os parâmetros relacionados são conhecidos, ou quando o indivíduo possui parâmetros inteiros é mais

Figura 3.2: Cruzamento por corte.

vantajoso que o cruzamento retorne um valor médio ao valor dos pais, ou ainda um valor dado por uma distribuição de probabilidade em função dos valores dos pais (Sastry & Goldberg, 2003).

Mutação

A mutação é aplicada nos indivíduos de forma aleatória, fazendo pequenas alterações no genótipo. Isso evita que os indivíduos fiquem limitados a um número de soluções dadas pela população inicial. Assim pequenas variações no genótipo, que seriam difíceis de conseguir por meio do cruzamento, podem ser obtidas através de mutação. A mutação serve para manter a variabilidade genética da população, atuando como um ajuste fino aos genes do indivíduo.

Predação

A predação consiste em periodicamente eliminar os indivíduos de menor aptidão, em seu lugar podem ser introduzidos indivíduos com genes aleatórios ou uma variação de um indivíduo de melhor aptidão. Além de aumentar a velocidade de convergência da população a predação aumenta a variabilidade, no caso do indivíduo aleatório introduzido, o que pode fazer com que a população encontre um novo espaço de busca que retorne um resultado melhor.

Existem outros operadores, além de variações dos mesmos aqui apresentados, a seção atual visou apresentar os métodos mais usados para AGs simples. A Seção 3.3 apresenta os algoritmos evolutivos multiobjetivos em suas principais variações.

3.3 Problemas de Otimização Multiobjetivo

Modelos multiobjetivos são representações realísticas para complexos problemas de otimização. Em muitos problemas da vida real, os objetivos considerados estão em conflito uns com os outros. Uma solução aceitável para um problema multiobjetivo é investigar um conjunto de soluções, cada uma satisfazendo um objetivo sem deixar que uma solução domine as demais.

Neste caso, o critério de otimalidade precisa ser reavaliado para considerar os múltiplos objetivos, para tal, é comum a utilização da Dominância de Pareto. Ótimo de Pareto é um termo

derivado da economia em que um determinado valor não pode ser melhorado sem prejudicar outros. Em um conjunto de soluções ótimo de Pareto é possível percorrer a fronteira entre as soluções considerando a perda e o ganho entre diferentes objetivos Figura 3.3 (Steuer, 1986).

Figura 3.3: Fronteira de pareto.

Na Figura 3.3, os círculos representam possíveis soluções para o problema de minimizar as funções $f1$ e $f2$, na fronteira estão as soluções em que não se pode minimizar o valor de uma função sem maximizar o da outra. Cabe ao algoritmo em questão escolher qual função terá preferência (Deb, 2001).

3.3.1 Abordagens de resolução

Os algoritmos para resolução de problemas multiobjetivos podem ser separados em três categorias principais (Abraham & Jain, 2005), conforme descrito na sequência.

Agregação de funções

Uma abordagem mais simples, consiste em transformar as diversas funções em uma única, e resolver o problema usando métodos monoobjetivos tradicionais. Como desvantagem ela apresenta apenas uma solução de cada vez, o que dificulta a obter diversidade no Conjunto de Pareto. Esta abordagem mostra bons resultados para alguns tipos de problemas, porém quando há objetivos conflitantes, é preciso considerar os pesos das funções em questão.

População

Uma população de soluções é utilizada para oferecer diversidade à busca, permitindo que um número maior de soluções próximas à Fronteira de Pareto sejam obtidas. De um modo geral, este tipo algoritmos não utiliza a Dominância de Pareto como critério de resolução, buscando encontrar a solução que mais se adapta a todos objetivos. Um dos algoritmos mais tradicionais foi o *Vector Evaluated Genetic Algorithm (VEGA)*, que inspirou novos algoritmos multiobjetivos (Schaffer, 1985).

Dominância de Pareto

Visando sanar problemas dos métodos anteriores, esta abordagem utiliza critérios de seleção baseados na Dominância de Pareto. Ordenando as soluções baseadas na proximidade com a fronteira de Pareto e a variedade da população. Proposto inicialmente em (Goldberg, 1989) foram surgindo diversas variações conforme o tempo, podendo citar o NSGA-II (Horn et al., 1994) e SPEA-2 (Zitzler et al., 2001) como as mais utilizadas.

3.3.2 Algoritmos Evolutivos Multiobjetivos

Para solucionar esse tipo de problema é possível modelar AGs usando múltiplas funções objetivo com operadores específicos para garantir a diversidade da população. Há duas abordagens mais utilizadas para AEMOs. A primeira é combinar as funções de aptidão para cada objetivo em uma função composta, assim o problema passa a ser calcular pesos das funções para obter a melhor solução. A segunda abordagem é determinar um conjunto de soluções ótimo de Pareto e nesse espaço escolher a que apresenta melhores resultados ao problema em questão (Dias & De Vasconcelos, 2002).

AGs são adequados para lidar com problemas multiobjetivos, uma vez que podem lidar com diferentes objetivos em paralelo e encontrar diferentes soluções de ótimo de Pareto (Goldberg, 1989). A constante geração de novos indivíduos a cada população permite mapear diferentes espaços o que facilita encontrar diferentes soluções. Usando-se técnicas que evitem a convergência de todos os indivíduos para uma mesma solução, aumenta-se a eficiência do algoritmo quando comparado a abordagens determinísticas (Schaffer, 1985).

Para o cálculo do sistema trifásico, avaliando as três componentes de onda do sistema foi proposta uma abordagem multiobjetiva considerando as componentes como funções independentes e também considerando as mesmas como parte de uma função objetivo mais complexa, mais detalhes do AEMO desenvolvido são dados na Seção 5.5.

3.3.3 Algoritmo Evolutivo Multiobjetivo por Tabelas

No AEMT, um indivíduo gerado é avaliado para diferentes funções objetivo. Ao apresentar bons resultados essa solução é salva em uma tabela relativa ao objetivo específico. Ao longo do processo, indivíduos de diferentes tabelas são utilizados na geração de novas soluções, até serem obtidos os melhores indivíduos para todos os objetivos em questão, sendo escolhida solução que melhor se adaptar às diferentes funções (Coello et al., 2007). Além disso, o AEMT pode encontrar soluções não-dominadas que os métodos mais conhecidos não conseguem (Santos et al., 2010).

Figura 3.4: Criação de indivíduo no AEMT.

O número de tabelas pode ser expandido conforme o número de combinações possíveis dos objetivos em questão, sendo o mais comum adicionar uma tabela referente a soma de todos objetivos. A Figura 3.4 ilustra as tabelas utilizadas no AEMT proposto na Seção 5.5.2 e o fluxo de criação e avaliação de um novo indivíduo. Pode se procurar funções mais simples, com apenas os objetivos essenciais à resolução do problema.

Figura 3.5: Inserção de novo indivíduo na tabela.

Ao contrário de abordagens apresentadas anteriormente, no AEMT não há separação das populações em diferentes gerações conforme o algoritmo avança. No AEMT, um indivíduo gerado irá ser avaliado por todas as funções objetivo, caso apresente boa aptidão ele será incluído na população adequada, substituindo o de pior aptidão, Figura 3.5, isso torna as populações sempre atualizadas, garantindo ao algoritmo uma conversão mais rápida.

O método foi escolhido também devido a sua simplicidade e facilidade de executar no FPGA, uma vez que requer poucas alterações em relação a um algoritmo monoobjetivo. Além do baixo custo computacional adicional, dado que ao calcular um objetivo, este pode ser usado nas diferentes tabelas que dependem da mesma função. A Seção 5.5.2 apresenta como foi desenvolvido o AEMT para o Relé de Frequência, e a Seção 6.4.3 apresenta os resultados obtidos.

3.4 Aumento de Eficiência de Algoritmos Evolutivos por paralelização

O modelo de programação tradicional trata os algoritmos como estruturas sequenciais. A forma de desenvolvimento de algoritmos limita-se ao paradigma de programação. Nela o algoritmo é um conjunto de instruções armazenadas em memória, que devem ser seguidas sequencialmente para obter o resultado esperado.

Algoritmos sequenciais são limitados pela velocidade de processamento, esta é limitada fisicamente pelo consumo de energia e superaquecimento dos processadores. Para contornar essa barreira novas arquiteturas são desenvolvidas, permitindo que instruções não conflitantes sejam processadas simultaneamente, além de processadores com diversos núcleos que podem executar mais de uma tarefa em paralelo.

Mas a arquitetura não muda o fato, de que os algoritmos codificados são feitos pensando de forma sequencial. Por mais paralela que seja a arquitetura projetada a partir de um algoritmo sequencial, há aspectos inerentes do próprio algoritmo que limitam o nível de paralelização. Para ter um ganho de desempenho mais impactante é preciso que o algoritmo seja reprojetoado desde seus elementos mais básicos de forma paralela. Para isso, deve-se identificar tarefas independentes que quando unidas têm a mesma função do algoritmo (Grama et al., 2003).

3.4.1 Fundamentos de métodos de paralelização

Para a criação de um algoritmo paralelo, pode ser usada como base uma versão sequencial ou pensar o problema desde o princípio de forma paralela. O ideal é pensar no paralelismo que se pode produzir em relação ao problema, compreendendo o inteiramente e identificando as partes que podem ser paralelizadas (Scott et al., 2011).

Divisão de Tarefas

A divisão da computação em pequenos conjuntos é chamada de decomposição. Existem vários meios de efetuar a decomposição, citando os principais: decomposição por dados, decomposição recursiva, decomposição exploratória e decomposição especulativa. As decomposições exploratória e especulativa são geralmente utilizadas em problemas mais específicos, enquanto que as decomposições recursiva e por dados são amplamente utilizadas na maioria dos problemas.

A granularidade é determinada pelo número e tamanho das tarefas pelo qual o problema é decomposto. Ela é fina quando as tarefas são decompostas em muitas tarefas de baixa complexidade e é grossa quando a decomposição é feita por meio de poucas tarefas com complexidade alta. Na prática, quanto mais uma tarefa aproxima-se de uma expressão simples, mais fina é sua granularidade e quanto maior o conjunto de operações contidas dentro de uma tarefa, mais grossa será (Cormen et al., 2001).

Técnicas de controle

Modelos de programação paralela podem ser divididos em dois grandes grupos, de acordo com a forma em que os processos interagem. A comunicação pode ser feita por troca de mensagens ou utilizando memória compartilhada (Figura 3.6).

Figura 3.6: Modelo de programação paralela: a) Troca de mensagem b) Memória compartilhada.

Troca de mensagens

No paradigma por troca de mensagens, também chamado *message-passing*, cada processo possui um espaço de memória dedicado, em que os demais processos não possuem acesso, Figura 3.6 (a). Isso evita acessos e escritas inválidas na memória e demais problemas de concorrência, uma vez que o processo original está ciente de todos os seus dados e das requisições feitas pelos demais processos.

As requisições de dados entre processos distintos são feitas por mensagens solicitando dados, chamadas de *send* e *receive*, podendo ser ou não bloqueantes de acordo com o método utilizado. Esse paradigma é de relativamente fácil de ser utilizado em diferentes arquiteturas, resultando, na maioria dos casos, em bom desempenho independente do número de processadores. Existem várias bibliotecas que permitem a abordagem de passagem de mensagens, como P4, PVM (*Parallel*

Virtual Machine), *Express*, *Parmacs* e *MPI (Message Passing Interface)* (Scott et al., 2011) (Foster & Karonis, 1998).

Memória compartilhada

Na paralelização com memória compartilhada, a comunicação entre os processos é implícita, uma vez que a mesma memória pode ser acessada por todos processadores, como mostrado na Figura 3.6 (b), não são necessárias funções específicas para comunicação.

É um modelo voltado para aplicações de alto desempenho, em que é possível ter vários processos simultâneos sincronizados. Em diferentes arquiteturas pode ser necessário uso de semáforos e outras técnicas de controle para evitar acessos indevidos. Na implementação do AE, diferentes indivíduos são criados simultaneamente e todos com acesso a memória que armazena a população, o controle de escrita é feito pelo próprio processo do indivíduo que é criado com área de escrita já definida.

Métricas de desempenho

A eficiência de um algoritmo paralelo é medida em função do desempenho de um algoritmo sequencial para a solução do mesmo problema dada a mesma entrada. A avaliação deve considerar o número de processos em que foi dividido o algoritmo, o tempo perdido em funções que não puderam ser paralelizadas e no gerenciamento do paralelismo.

Na Equação 3.1, o *speedup*, S é dado pela razão entre o tempo de execução do algoritmo sequencial, T_s , e o tempo de execução do algoritmo paralelo, T_p , e mede o desempenho do algoritmo paralelo.

$$S = \frac{T_s}{T_p} \quad (3.1)$$

O *speedup* pode ser maior que o número de processadores, super-linear, em casos que o uso de mais memória simplifique etapas do processamento, evitando o acesso a memória de menor velocidade. A Equação 3.2 define a eficiência E de um algoritmo paralelo como sendo a razão do *speedup* pelo número de processadores P

$$E = \frac{S}{P}. \quad (3.2)$$

O ideal é atingir 100% ($E = 1$) de eficiência. No entanto isso dificilmente é possível, os processadores não podem dedicar-se inteiramente ao problema, pois estes realizam também o gerenciamento das interações entre processos, o que não ocorre no paradigma sequencial (Scott et al., 2011).

3.4.2 Algoritmos Evolutivos Paralelos

Pela forma como são construídos, os AEs são altamente paralelizáveis, o que permite reduzir significativamente o tempo de execução do algoritmo. Diferentes soluções podem ser avaliadas de forma independente, e a complexidade de gerenciamento é muito baixa, uma vez que cada solução contém a informação relacionada há uma diferente instância do problema (CantuPaz, 1997).

De um modo geral, a função de avaliação toma maior parte do tempo de processamento em um AE. Partindo do princípio de que cada solução é independente, é possível dividir o AE de acordo com o número de indivíduos por geração, o que tornará o paralelismo efetivo visto que toda a parte de avaliação será paralelizada. Podem ser adicionados parâmetros extras para melhor aproveitar a arquitetura paralela, dividindo os indivíduos em diferentes populações é preciso levar em conta a quantidade e o tamanho de populações criadas, além de como será feita a interação entre populações de processos distintos (Alba, 2005).

Se o AE vai ser executado em um FPGA ou em uma rede computadores deve se levar em conta as características de cada arquitetura. Considerando o FPGA com unidades de processamento mais básicas e com baixo custo para a troca de informação entre processos, ao contrário de uma arquitetura em rede, na qual os processos podem ser bastante complexos, porém possuem grande custo de comunicação (Le & Rejeb, 2006).

AEs paralelos são classificados de acordo com a divisão das subpopulações, pela forma com que elas interagem e como são feitas as migrações entre diferentes instâncias do problema. Das principais classificações de AEs paralelos é possível citar (Alba, 2005):

População simples *boss-worker*

A implementação mais básica de um AE paralelo, é distribuir o cálculo de função objetivo em diferentes processos, enquanto um processo principal coordena a população e criação de novos indivíduos. Assim o espaço de busca do AE é o mesmo de um método serial, porém o *speedup* conseguido ao paralelizar a função objetivo é significativo gerando grande diminuição do tempo de processamento.

O desempenho de um AE paralelo *boss-worker* é relacionado ao tempo de cálculo da função objetivo. O processo *boss* deve ser capaz de suprir os demais com novos indivíduos, evitando que fiquem ociosos, o mesmo cuidado deve ser tomado com o *boss* para que ele não perca tempo esperando os resultados. O tempo de comunicação entre os processos é essencial para isso, o que torna essa classificação ideal para FPGAs uma vez que os processos executam todos no mesmo circuito integrado.

Múltiplas populações

O AE paralelo de múltiplas populações é a implementação mais comum, utilizada em arquiteturas mais sofisticadas com alto poder de processamento, porém com maior custo de comunicação.

As interações entre populações, chamadas de migrações, podem ser controladas por parâmetros como frequência, número de imigrantes e método de seleção (Alba, 2005).

Os parâmetros que mais influenciam a eficiência do método são o tamanho de cada população e os efeitos da migração. Uma abordagem com pouca migração não seria, em termos de resultados, muito diferente de uma implementação serial com o mesmo número de indivíduos. Assim, compensa aumentar o número de migrações conferindo um melhor variedade de soluções, mesmo ao custo do aumento de comunicação e gerenciamento do AE (Alba, 2005).

O uso de AEs paralelos multipopulacionais, permite a redução do tamanho da população, quando comparada ao AG serial. O conhecimento prévio de genes que possuem relações entre si, também chamados *building blocks* (Goldberg, 2002), ajuda na criação de populações iniciais. Deixando cada população se especializando em uma parte do problema, garantindo convergência rápida, mesmo com taxas de migração baixas. Porém, em problemas mais complexos ou de larga escala, não costuma ser ter o conhecimento prévio de *building blocks*.

A escolha de quais indivíduos devem migrar deve ser levada em conta. Em populações com muita diversidade, escolher os imigrantes e como eles serão incorporados na população receptora é um fator importante. A pressão de seleção de um AG influencia na velocidade de convergência de uma população. Uma convergência muito lenta ou rápida pode fazer o algoritmo falhar. Visto que, as migrações influenciam na pressão de seleção (Alba, 2005) (Goldberg, 2002), a escolha dos imigrantes pode ser aleatória ou baseada na aptidão deles.

AG Paralelo Hierárquico Híbrido

Os sistemas híbridos podem ser construídos combinando características de AGs paralelos. Também chamados de hierárquicos, pois um nível gerencia o nível logo abaixo. O AG Paralelo Hierárquico combina as qualidades dos modelos para obter melhor desempenho ao custo do aumento da complexidade na modelagem e controle do algoritmo (Alba, 2005).

Com essa hierarquia, os indivíduos podem migrar para diferentes populações, aumentando o espaço amostral destas, visando melhorar a qualidade das soluções. Na combinação do mestre-escravo com o multipopulacional, a migração é feita entre ilhas de indivíduos e a avaliação dos mesmos é feita em paralelo, estratégia adequada nos casos do cálculo da função objetivo ser muito custoso (CantuPaz, 1997).

3.4.3 Aspectos de paralelização em hardware

AEs demandam tempo para que seja encontrada a solução adequada ao sistema, até que a população convirja para uma valor ótimo, são necessárias diversas gerações. No algoritmo desenvolvido, são avaliados 25000 indivíduos a cada nova janela do sinal, considerando a versão com 500 gerações de 50 indivíduos cada.

Uma vez que os AEs são altamente paralelizáveis, com operações simples que são executadas repetidas vezes, o processo de criação e avaliação do indivíduo, é possível executá-las de forma simultânea sem restrições ao funcionamento do método. Teoricamente, é possível avaliar ao mesmo tempo todos os indivíduos da mesma população, porém devem ser considerados os limites do hardware em questão.

O funcionamento do hardware configurável é explicado no Capítulo 4. A Seção 5.3 trata do funcionamento do algoritmo em hardware, explicando as divisões de tarefas e o controle do paralelismo.

3.5 Considerações

AEs permitem uma busca exploratória ampla mesmo que o conjunto de soluções seja variado. O AE possibilita a evolução de soluções promissoras em direção à melhor solução. No método proposto (Seção 5.2), a avaliação de frequência é feita a intervalos constantes, uma nova iteração do AE pode utilizar o resultado anterior como ponto de partida para a nova estimativa. Os indivíduos podem ser avaliados de forma independente, o que facilita o desenvolvimento do algoritmo de forma paralela, já visando o desenvolvimento para FPGAs.

O uso de modelos multiobjetivos, como o AEMT, aumentam a eficiência e flexibilidade do relé, tornando-o mais tolerante em caso de quedas de fase. É possível implementar o AEMT de forma paralela no FPGA, com os diferentes objetivos calculados de forma simultânea. O Capítulo 4 apresenta os FPGAs como uma opção eficiente e flexível de desenvolvimento e paralelização de AEs em aplicações de tempo real.

Hardware Reconfigurável

É possível definir as arquiteturas de computação de acordo com seu grau de flexibilidade, partindo de algoritmos funcionando totalmente em *hardware* (ASIC), passando por *hardware* reconfigurável, soluções mistas de *hardware* e *software*, processadores de sinal digital (do inglês *Digital Signal Processor* - DSP), até chegar a sistemas em *software* desenvolvidos para arquiteturas físicas de propósito geral (arquitetura de von Neumann), variando em flexibilidade e desempenho como mostra a Figura 4.1 (Gois, 2011).

Figura 4.1: Relação entre desempenho e flexibilidade.

O conceito de *hardware* reconfigurável surgiu inicialmente com os PLD e CPLD, eram matrizes de portas lógicas que podiam ser configuradas para gerar diferentes equações lógicas. Porém estes não permitiam criar operações complexas, pois faltava praticidade nas conexões, nesse contexto surgiram os FPGAs.

4.1 Field-Programmable Gate Array

Um FPGA é um circuito integrado cuja funcionalidade pode ser programada pelo usuário final. A configuração, normalmente, é feita usando uma linguagem de descrição de *hardware*, tais como Verilog ou VHDL, similares às usadas na programação de ASICs. FPGAs podem ser configurados para emular qualquer função lógica, sendo limitados apenas pelo número de blocos lógicos e conexões disponíveis, podendo ser reprogramados pelo usuário conforme a sua necessidade. A possibilidade de reprogramação torna o desenvolvimento em FPGAs mais barato quando comparado ao desenvolvimento em ASIC (Bobda, 2007).

O desenvolvimento de um algoritmo diretamente em *hardware* permite um maior grau de paralelismo, pois operações repetitivas podem ser replicadas no *hardware* e processadas simultaneamente, gerando um aumento considerável no desempenho do algoritmo, desde que essas operações sejam independentes.

Um FPGA é composto por blocos de entrada e saída, que se ligam a blocos lógicos. As conexões são gerenciadas por chaves de interconexão que servem de rota entre blocos (Kilts, 2007).

Figura 4.2: Diagrama de componentes de um FPGA.

Blocos lógicos

Um bloco lógico é uma memória que armazena os valores de saída para cada possível combinação de entradas, o tipo de memória e quantidade de entradas são parâmetros usados para diferenciar os FPGAs. Normalmente se usam quatro entradas com memória SRAM, EEPROM ou Flash, há também versões que são programáveis apenas uma vez usando EPROM ou CMOS. A Figura 4.3 esquematiza a arquitetura de um bloco lógico.

Figura 4.3: Componentes de um bloco lógico.

As entradas servem de endereçamento para a tabela de resposta, também chamada *look-up table*. Os blocos são organizados formando uma matriz bidimensional. A Figura 4.4 ilustra a disposição dos pinos no bloco lógico.

Figura 4.4: Disposição dos pinos no bloco lógico.

Chaves de interconexão

Nos pontos de encontro entre canais verticais e horizontais ficam as chaves de interconexão. Numa arquitetura padrão a chave possui três conexões programáveis que permitem que um canal se conecte com os outros três adjacentes. Nessa topologia apenas canais adjacentes podem se conectar. A Figura 4.5 ilustra a chave de interconexão.

Normalmente, o roteamento em um FPGA é feito sem segmentação, cada fio se conecta somente a um bloco lógico antes de terminar em uma chave de interconexão. Ligando algumas chaves programáveis é possível criar longos caminhos que se conectam em diversos blocos lógicos.

Figura 4.5: Chave de interconexão.

Blocos de entrada e saída

Responsáveis pela comunicação com o mundo exterior localizam-se na periferia do chip e são conectados aos blocos lógicos através das chaves de interconexão. Eles podem receber dados vindos de pinos e botões na placa do FPGA, e enviar mensagens através de *displays*, como também podem enviar e receber dados para periféricos via conexão serial, USB, ethernet, adicionando flexibilidade ao FPGA.

4.2 Gerador de números aleatórios

Devido a sua natureza não determinística, o AG necessita constantemente de números aleatórios para efetuar suas operações. Dado a quantidade de números utilizados e a frequência de atualização, é necessário um método que seja simples e possa fornecer novas entradas pseudoaleatórias de forma constante, evitando atraso no processamento.

Os principais métodos de se obter números aleatórios em *hardware* são usando dados provenientes de fontes externas, dados pré-armazenados em memória, e também através de relações de recorrência como o LFSR - *Linear Feedback Shift Register* (L'Ecuyer, 1996). Para essa aplicação os três métodos foram testados obtendo-se uma combinação dos dois primeiros métodos.

4.2.1 Entrada aleatória

O sinal obtido em tempo real pode ser usado como semente ou como o próprio valor aleatório. Deve-se considerar, dada uma distribuição de valores, se é possível gerar uma Função de Distribuição de Probabilidade próxima à distribuição uniforme. Também é necessário verificar se a taxa de atualização desse valor torna possível a atualização dos valores de acordo com a frequência desejada.

São comumente usados como entrada o som, luminosidade ambiente, ou a própria tensão variável na corrente alternada. O que acaba gerando a necessidade de adicionar *hardware* especial para captar essas entradas, e torna o método mais interessante quando se deseja uma entrada realmente aleatória para fins estatísticos (L'Ecuyer, 1996).

4.2.2 Tabela pré-calculada de números aleatórios

Armazenam-se em memória valores aleatórios pré-calculados, esses valores são lidos e conforme o tempo passa voltam a se repetir. Para obter boa eficiência e qualidade é necessário que a quantidade de valores armazenados seja grande o suficiente para que os mesmos valores não se repitam em uma mesma operação.

Para aumentar a eficiência um mesmo vetor pode ser usado para alimentar diferentes circuitos consumidores. Assim, a cada vez que o vetor for consumido por completo, os dados passaram a se repetir, mas cada valor será usado por um consumidor diferente. Essa estratégia foi usada para alimentar os diferentes operadores do AG desenvolvido.

4.2.3 LFSR - *Linear Feedback Shift Register*

LFSR consiste em um gerador de números pseudoaleatórios muito utilizados em sistemas digitais. Em geral, o LFSR é composto por duas partes: um registrador de deslocamento e uma função de realimentação. O bit de entrada do registrador de deslocamento é uma função linear com os seus estados prévios (Murase, 1992).

4.3 Funções trigonométricas

No problema em questão, é necessário o uso da função seno para, em função do tempo e ângulo de fase, calcular o sinal gerado juntamente com a amplitude máxima. Funções trigonométricas possuem alto custo computacional, por isso uma abordagem própria para FPGAs deve ser utilizada para permitir maior desempenho.

Dos métodos existentes é possível citar a aproximação por polinômio de Taylor, o uso de uma tabela com valores pré-calculados além do método de Cordic (Volder, 1959). Devido ao custo computacional e simplicidade dos dados, tendo sempre precisão constante foi utilizada uma tabela de senos salva em memória (Edwards & Penney, 2008).

4.3.1 Polinômio de Taylor

Baseado no Teorema de Taylor é possível obter uma aproximação de qualquer função utilizando-se apenas de operações polinomiais simples. O polinômio de Taylor é formado pelas funções derivadas da função original e os valores da função para determinado ponto fixo.

Para se calcular o valor do seno como aproximação do cálculo de uma função senoidal, usa-se a série:

$$\sin(x) \approx x - \frac{1}{6}x^3 + \frac{1}{120}x^5 - \frac{1}{840}x^7 \quad (4.1)$$

calculando os quatro primeiros termos da série tem-se um erro da ordem de 10^{-6} , mas o custo computacional é alto visto que é preciso calcular termos da ordem de x^7 , além de efetuar divisões para cada termo do somatório. Dado que o cálculo do seno seria feito para cada amostra do sinal, repetidas para cada indivíduo, isso geraria uma grande sobrecarga no processamento (Edwards & Penney, 2008).

4.3.2 Valores tabelados de seno

Salvando valores de seno em memória tem-se a vantagem de conseguir rapidamente os valores necessários, a um custo computacional fixo, porém com grande consumo de espaço. O seno é usado na função de avaliação do AG, sendo esse o gargalo do sistema, o uso de memória se justifica. Dado também que a precisão é fixa o que simplifica a utilização de apenas uma tabela para a função.

4.3.3 Cordic

COordinate Rotation DIgital Computer (coordenar a rotação digital do computador) é um método voltado para sistemas que não possuem *hardware* dedicado à multiplicação. Utilizando uma tabela de arco-tangentes é possível através de adições e subtrações calcular com precisão os valores de seno e cosseno para todo ângulo, com precisão variando conforme a necessidade e tamanho da tabela. É um método iterativo e conforme o número de passos se obtém uma melhor aproximação, como mostra a Figura 4.6.

Figura 4.6: Diagrama de iteração do algoritmo Cordic.

O método é muito eficiente, usado em diversos sistemas microprocessados e utiliza menos memória que a solução por valores tabelados, porém não fornece o resultado em tempo constante, necessitando vários ciclos. Isto levou a optar pelo uso de uma tabela completa (Volder, 1959).

4.4 Linguagem de descrição de *hardware*

As técnicas tradicionais para descrever *hardware*, por meio de especificação de interconexões dos elementos lógicos, tornam difícil o projeto do *hardware* se este envolve grande número portas lógicas. Nesse sentido, para auxiliar a modelagem de circuitos lógicos digitais complexos, têm se utilizado linguagens de descrição de *hardware* (HDLs, do inglês, *Hardware Description Language*). Essas linguagens realizam a descrição dos sistemas no nível físico e comportamental, proporcionando alto nível de abstração, possibilitando ao projetista a facilidade de testar o funcionamento dos circuitos digitais por meio de simulações, para sintetizá-lo após a descrição. Existem várias linguagens de descrição de *hardware*, entre as quais podem ser citadas: Verilog (Palnitkar, 2003), VHDL (Brown & Vranesic, 2008), SystemC (Müller et al., 2003), Handel-C (Martin, 2001) e *Bluespec System Verilog* (Dave et al., 2007).

VHDL, Linguagem de descrição de *hardware* para circuitos integrados de alta velocidade, é uma linguagem de descrição de *hardware* usada tanto para a programação de ASICs quanto para FPGAs. Originalmente desenvolvida por militares para facilitar a troca de informação sobre circuitos eletrônicos. Derivada da Linguagem de programação Ada, permite a criação de entidades com funcionalidade definida e descrição de entradas e saídas. O processamento ocorre dentro da unidade e é normalmente iniciado por sinais, que são gerados devido a mudanças nas entradas. Sendo uma linguagem para *hardware*, permite que instruções sejam executadas simultaneamente dentro de um processo definido (Ashenden, 2008).

A linguagem permite aos projetistas descrever o comportamento do *hardware* e definir as portas de entrada e saída dos componentes, de maneira similar às linguagens de alto nível. Um projeto em VHDL é baseado em entidades, que podem representar desde uma simples porta lógica até circuitos lógicos mais complexos, com funções aritméticas e de controle de dados.

VHDL foi desenvolvida com o objetivo de simular, modelar e documentar projetos de circuitos integrados de alta velocidade. Posteriormente, recebeu a possibilidade de síntese para automatizar projetos de circuitos lógicos digitais. Contudo, para o desenvolvimento de projetos com grande número de elementos lógicos, a linguagem apresenta significativa demora na realização, uma vez que, os projetistas devem tratar de detalhes que são removidos das linguagens de alto nível.

4.5 Considerações

Nesse capítulo foi apresentado o conceito de hardware reconfigurável enfatizando-se aspectos importantes para o projeto de um novo relé de frequência e fase por AEs. No projeto foram utilizados FPGAs da família Stratix II da Altera, apesar do relé não ser desenhado especificamente para essa família, esta foi usada como exemplo para avaliar diferentes versões do projeto. Foi utilizada a interface de desenvolvimento Quartus II e a linguagem VHDL, junto com o diagrama esquemático que permite visualizar as entidades geradas. O uso de memórias para números ale-

at6rios e c6culo da fun76o seno foi feito de maneira a se adequar a quantidade de mem6ria no FPGA. A Se76o 5.3 apresenta mais detalhes da utiliza76o do FPGA e do desenvolvimento do AE paralelizado em FPGAs.

Desenvolvimento da proposta de estimador de Amplitude, Frequência e Fase

Neste capítulo são apresentados detalhes do desenvolvimento do algoritmo, buscando apontar as razões dadas para cada opção de desenvolvimento utilizada. Na Seção 5.1 é detalhada a precisão dos dados e a forma como são calculados. A Seção 5.2 refere-se ao algoritmo e como foi feito para se conseguir eficiência nas operações. A Seção 5.3 trata da utilização do FPGA quanto ao número de portas utilizadas além de hardware dedicado. A Seção 5.4 trata do método de redução de espaço de busca utilizado. A Seção 5.5 diz respeito à adição de um componente multiobjetivo ao sistema tratando as três fases de onda de forma simultânea. Por fim, a Seção 5.6 resume as técnicas utilizadas para o desenvolvimento do projeto.

5.1 Estrutura de dados

A estrutura de dados criada foi pensada para maximizar a eficiência das operações quando aplicadas em um FPGA, visando limitar as operações matemáticas a funções de soma, subtração e deslocamento de *bits*. A precisão de cada valor representado, dado pelo número de *bits* utilizados, foi obtida com base na precisão necessária, e adaptada conforme a obtenção de resultados no decorrer do projeto.

5.1.1 Entrada de dados

O sinal trifásico obtido da rede elétrica, na frequência nominal de 60 Hz, é recebido por um conversor analógico-digital com 16 *bits* de precisão para cada fase do sinal. Não é do escopo desse projeto o tratamento do sinal analógico, sendo que o algoritmo passa a considerar apenas após a sua digitalização.

Um valor do sinal digitalizado é salvo a cada 1,3 ms, em uma frequência de amostragem de cerca de 770 Hz é possível conseguir 15 amostras para cada ciclo completo do sinal. Os dados são salvos em um vetor e atualizados a cada nova amostra. Possuindo 15 valores, com 16 *bits* cada, estes serão usados na comparação com o valor gerado pelo AG.

5.1.2 Representação dos indivíduos

No AG, cada indivíduo representa uma possível solução. A complexidade da avaliação de cada indivíduo afeta diretamente o desempenho do algoritmo e a utilização de área do FPGA. Assim, os valores representados devem ser pensados de modo a maximizar a qualidade do resultado e minimizar a complexidade da avaliação. Cada indivíduo da população possui as seguintes características:

Frequência

A frequência é o parâmetro que apresenta maior variabilidade durante a execução do algoritmo. O parâmetro opera consigo mesmo, definindo diferentes valores para cada instante de tempo, juntamente com o ângulo de fase para determinar o seno da onda no momento em questão.

São utilizados 24 *bits* para representar a frequência, sendo que ao operar com o ângulo de fase apenas os 16 *bits* mais significativos de cada instante são considerados. Os *bits* menos significativos são utilizados para diminuir erros de aproximação, que acontecem devido a somas sucessivas, para se conseguir valores de tempo mais altos.

O valor representado no cromossomo, f_i equivale à frequência do sinal no instante $t = 8$, com $f_i = [0..2^{24} - 1]$ representando frequências entre 0 Hz até a frequência de amostragem, $F_A = 769,23\text{Hz}$, para se obter o valor real em função do cromossomo temos

$$\begin{aligned} \frac{8f}{F_A} &= \frac{f_i}{2^{n_b}} \\ f &= \frac{f_i}{8} \cdot \frac{F_A}{2^{n_b}}, \end{aligned} \quad (5.1)$$

em que f_i é o valor representado no cromossomo, referente ao instante da oitava amostra, f é o valor de frequência real, com base na frequência de amostragem, $F_A = 769,23$, e no número de

bits da representação $n_b = 24$. O uso da frequência no instante $t = 8$ é detalhado na Seção 5.2.2 referente ao cálculo da função objetivo.

Ângulo de fase

Representa o ângulo do sinal no instante inicial da janela amostrada, $T = 0$. Ao operar com a frequência retorna o ângulo do instante em questão, $T > 0$, que será usado no cálculo do seno. A precisão do ângulo de fase é limitada pela precisão do sinal de entrada, sendo que não ocorre ganho a partir do momento em que o número de *bits* é maior que a precisão da entrada. Também é limitado pela quantidade de memória, uma vez que a cada bit adicionado o tamanho da tabela de senos dobra para representar mais valores. No relé padrão são usados 16 *bits* de precisão.

Amplitude

Multiplicada pelo seno, obtido dos valores anteriores, retorna um dos valores a ser comparados com a amostra real de sinal. Assim como o ângulo de fase, é limitada pela precisão da entrada. Sendo que a partir do momento que a precisão da amplitude é maior que a do sinal, o cálculo passa a ser desnecessário, pois não há como garantir precisão maior que a da entrada. São usados 16 *bits* de precisão assim como a precisão do seno, uma vez que aumentar essa precisão geraria uma maior complexidade dos multiplicadores utilizados.

Aptidão

O valor que define a qualidade de um indivíduo quando comparado ao sinal real. É obtido com a soma das diferenças entre as amostras reais e as geradas pelos parâmetros anteriores. Matematicamente não há como esse valor passar de cinco *bits* a mais da precisão do sinal de entrada, sendo assim são usados 24 *bits*, visto que as operações de soma são simples e não apresentam grande consumo de área do FPGA.

5.2 Projeto de Algoritmo Genético paralelizado em FPGA

O AG proposto inicia com uma população aleatória, com indivíduos espalhados pelo espaço amostral. O processamento reinicia a cada nova janela, com a obtenção de uma nova amostra. O Algoritmo segue conforme o fluxograma da Figura 5.1, seguindo as etapas de Inicialização, Processamento e Resultado.

Dado que o AG é voltado para um sistema de tempo real, não se pode ter a condição de parada dependente das condições de entrada ou de processamento, assim para garantir a exibição do resultado, a condição de parada é fixa sendo atingida ao início de uma nova janela, independente da qualidade da resposta alcançada.

Figura 5.1: Fluxograma do algoritmo.

Figura 5.2: Processamento paralelo.

A Figura 5.2 explicita o processamento paralelo de indivíduos, baseado na arquitetura mestre-escravo com o processamento sendo controlado por uma instância superior e a avaliação de indivíduos em instâncias menores. No FPGA foram testadas versões com 1, 2 e 4 indivíduos gerados e avaliados simultaneamente. Novos indivíduos são gerados usando os operadores genéticos destacados na Figura 5.3 e o algoritmo continua até atingir a condição de parada.

Figura 5.3: Gerao do indivduo.

5.2.1 Operadores utilizados

Os operadores genticos so responsveis por gerar a nova populao, com base na populao anterior. Eles foram alterados de forma a simplificar operaes executadas no FPGA. So eles:

Elitismo

O elitismo garante que o melhor indivduo da populao atual continue na prxima, evitando que uma boa soluo se perca durante o processamento. Funciona armazenando seus dados em um elemento fixo da populao, no caso o primeiro indivduo, e fazendo com que os prximos operadores substituam apenas os outros indivduos da populao. ativado sempre que a funo objetivo encontra um indivduo com aptido melhor que a do indivduo atualmente tido como o melhor. Quando se inicia uma nova populao (quadro "Nova Populao" na Figura 5.1), o melhor indivduo da populao anterior mantido, da mesma forma em que quando uma iterao do AG concluída e inicia-se uma nova janela o melhor indivduo da janela anterior usado como base para a nova populao.

Torneio

Utilizado para selecionar indivduos da populao atual, que sero usados no prximo operador. O torneio seleciona dois pares aleatoriamente e fica com o melhor de cada par. Os dois melhores indivduos so passados para o prximo operador.

Cruzamento

Responsvel por criar um indivduo para a prxima gerao com base nos vencedores do torneio. Ao contrrio do cruzamento por corte (Figura 3.2), os componentes do indivduo, frequncia, amplitude e fase, so mesclados de forma independente, de modo que *bits* referentes amplitude, no se misturem com *bits* relacionados frequncia ou ngulo de fase no novo indivduo. Cada componente criado com base nos componentes dos pais podendo gerar cinco valores distintos com igual probabilidade.

A Figura 5.4 ilustra a os possveis dos valores do novo cromossomo, em relao aos cromossomos dos pais, x_i^t e x_j^t . Os valores possveis, alm dos prprios x_i^t e x_j^t , so a mdia entre os dois

Figura 5.4: Cruzamento por característica.

valores, $x_i^t + \delta^t$, e os valores distantes dos pais $x_i^t - \delta^t$ e $x_j^t + \delta^t$, com $\delta^t = \frac{x_j^t - x_i^t}{2}$, sendo um dos cinco valores escolhido de forma aleatória para cada componente.

Mutação

Após a criação do novo indivíduo pelo operador de cruzamento, acontece a operação de mutação. Baseada em uma função de probabilidade são feitas alterações no indivíduo gerado. No AG é usada uma probabilidade de 20% de mutação para cada um dos componentes básicos do indivíduo (frequência, ângulo de fase e amplitude) de forma independente. Caso um valor seja escolhido para mutação é feita uma soma ou subtração de uma constante ao valor atual do componente. A mutação é focada apenas em alterações pontuais evitando grandes alterações para que o indivíduo não divirja muito dos indivíduos geradores.

5.2.2 Função objetivo

A função objetivo deve avaliar os indivíduos de forma prática e rápida. A função

$$Aptidao = \sum_{t=0}^{N-1} |Sinal(t) - V_m \cdot \sin(2\pi f \frac{t}{F} + \phi)|, \quad (5.2)$$

retorna exatamente a aptidão do indivíduo, servindo como estimador de quão próximo o indivíduo está do estado atual do sinal. Sendo: *Sinal* o vetor contendo as ultimas N amostras da rede elétrica; V_m , f e ϕ respectivamente o valor de amplitude, frequência e ângulo de fase que o indivíduo representa e F a frequência de amostragem dos dados.

Tabela 5.1: Codificação de parâmetros proposta inicialmente no projeto.

Parâmetro	No. de <i>bits</i>	Valores Inteiros	Valores Reais
V_m	8	[0; 255]	[2, 5; 12, 5]
f	8	[0; 255]	[58, 0; 62, 0]
ϕ	9	[0; 511]	[0; 2π]

Num primeiro momento foi usada uma abordagem em que os parâmetros do indivíduo eram usados como índice em tabelas, estas retornariam os valores já calculados. Cada operação era pré-calculada e armazenada em memória no momento de programação do FPGA. Ao invés de executar a operação os parâmetros eram usados como endereçamento para conseguir o resultado na memória.

No caso, a operação $2\pi f \frac{t}{F} + \phi$ contida na Equação 5.2 seria calculada passando os parâmetros de frequência, tempo, e fase para a memória, a fim de se chegar ao valor do seno. Os valores de seno e amplitude eram concatenados para gerar o endereço que retornaria o valor do sinal. Essa abordagem, usando parâmetros como endereços, gerou um atraso referente ao acesso aos dados da memória, além de muitas combinações repetidas que aumentaram o tamanho da memória em função da precisão dos parâmetros usados.

Em uma nova abordagem, os dados são representações discretas de valores reais, ao invés do valor em si, eles são representados em escalas para serem usados no cálculo da função, é o caso dos valores de frequência, tempo e ângulo de fase que quando operados entre si retornam um índice da memória de senos e da amplitude que ao multiplicar pelo valor de seno retorna um valor na escala recebida do conversor analógico digital.

O modelo do sinal gerado, a função senóide na Equação 5.2, pode ser calculado usando as funções

$$\omega(f, t) = 2\pi f \frac{t}{F}, \quad (5.3)$$

$$\theta(\omega, \phi) = \omega + \phi, \quad (5.4)$$

$$S(\theta) = \text{sen}(\theta), \quad (5.5)$$

$$V(V_m, S) = V_m S. \quad (5.6)$$

A Equação 5.3 visa calcular os valores da frequência angular para cada instante de tempo. A representação de ω se dá em função da frequência de amostragem F , fixada em 769,23 Hz. O cromossomo f equivale ao valor do parâmetro ω no instante $t = 8$, assim, pode-se calcular com maior precisão e rapidez os outros valores de $\omega(t)$. Para determinar os demais valores de $\omega(t)$, o procedimento mais simples seria a soma em cadeia, em que cada valor só poderia ser calculado após todos seus antecessores já estivessem calculados, conforme o exemplo a seguir:

$$\begin{aligned} \omega(2) &= \omega(1) + \omega(1) \\ \omega(3) &= \omega(2) + \omega(1) \\ \omega(4) &= \omega(3) + \omega(1) \\ &\dots \\ \omega(14) &= \omega(13) + \omega(1). \end{aligned} \quad (5.7)$$

Porém, uma soma via deslocamentos de *bits* (operador $\gg$ (Schildt & Mayer, 2006)), mostra-se mais eficiente, conforme ilustrado a seguir:

$$\begin{aligned}\omega(4) &= \omega(8) \gg 1 \\ \omega(2) &= \omega(4) \gg 1 \\ \omega(1) &= \omega(2) \gg 1 \\ \omega(0) &= 0.\end{aligned}\tag{5.8}$$

Na prática, o FPGA apenas redireciona as conexões, uma vez que não é necessário passagem de *carry*, como na operação de soma, assim o descolacamento é feito sem atraso. Os próximos valores seriam calculados através de somas:

$$\begin{aligned}\omega(3) &= \omega(2) + \omega(1) \\ \omega(5) &= \omega(4) + \omega(1) \\ \omega(6) &= \omega(4) + \omega(2) \\ \omega(9) &= \omega(8) + \omega(1) \\ \omega(10) &= \omega(8) + \omega(2) \\ \omega(12) &= \omega(8) + \omega(4),\end{aligned}\tag{5.9}$$

os valores mais complexos necessitam de duas somas:

$$\begin{aligned}\omega(7) &= \omega(4) + \omega(2) + \omega(1) \\ \omega(11) &= \omega(8) + \omega(2) + \omega(1) \\ \omega(13) &= \omega(8) + \omega(4) + \omega(1) \\ \omega(14) &= \omega(8) + \omega(4) + \omega(2).\end{aligned}\tag{5.10}$$

Com somas independentes, os valores de frequência para cada instante da janela são calculados em um ciclo de processamento, podendo então ser adicionados ao ângulo de fase para se calcular as Equações 5.4 e 5.5.

Com as restrições de valores de frequência, Tabela 5.2, os valores obtidos das Equações 5.8, 5.9 e 5.10 são distribuídos por um ciclo completo de onda, valores indo de 0 a 2π , igualmente espaçados de acordo com a frequência representada. Isso torna o acesso aos dados da memória de seno fragmentado por toda a sua extensão, o que permite dividi-la em 16 memórias menores acessadas de forma independente, sem prejudicar o desempenho ao efetuar múltiplos acessos. De fato, quando se aumenta o paralelismo de indivíduos processados simultaneamente, torna-se necessário a replicação da memória para manter o desempenho. Isto se torna o gargalo do sistema, pois dificulta o aumento da precisão do ângulo e o paralelismo de indivíduos.

Tabela 5.2: Codificação final de parâmetros.

Parâmetro	No. de bits	Valores Inteiros	Valores Reais	Valores Restritos
V_m	16	$[0; 65365]$	$[0V; 2, 0V)$	$[0, 50V; 1, 50V]$
f	24	$[0; 2^{24} - 1]$	$[0Hz; 96, 15Hz)$	$[55, 0Hz; 65, 0Hz]$
ϕ	16	$[0; 65365]$	$[0; 2\pi)$	

Por fim, na última etapa, ocorre a multiplicação da amplitude pelos valores de seno, Equação 5.6, estes calculados para cada instante de tempo. A representação desses valores é feita de forma que, quando multiplicados, o resultado esteja na mesma escala do sinal amostrado na entrada, evitando conversões desnecessárias.

Conforme a evolução dos testes e adaptações dos parâmetros, chegou-se a configuração final, apresentada na Tabela 5.2. Os intervalos descritos na coluna "Valores Restritos" representam o espaço de busca inicial do AG, podendo este explorar valores além dessas faixas conforme as condições do sinal.

5.3 Paralelização em FPGA

Os recursos básicos de um FPGA são os blocos lógicos, que são usados para criar funções combinatórias, além de circuitos específicos atuando como memórias e DSPs. O AG precisa balancear a qualidade da estimação com a quantidade de blocos lógicos utilizados.

Paralelizar a geração de todos os indivíduos iria requisitar muitos recursos excedendo as capacidades dos FPGAs. A Tabela 5.3 mostra os requisitos mínimos para o funcionamento do AG com diferentes níveis de paralelismo no nível de indivíduos, para $p = 4$ tem-se um resultado próximo a $p = 2$ visto que foi sintetizado em um FPGA com diferentes características.

Tabela 5.3: Requisitos mínimos do AG.

Característica	Requisitos		
	$p = 1$	$p = 2$	$p = 4$
Funções Combinatórias	5.339	9.205	9.192
Bits de Memória	404,736	408.064	305.664
Registradores	2,989	3.032	3.031
DSPs	16	32	32
Desempenho (ms)	0,355	0,177	0,089
FPGA	F1	F1	F2
Preço	US\$ 995	US\$ 995	US\$ 1.995

O valor apresentado na linha de Desempenho da Tabela 5.3 é o tempo médio que o AG leva para chegar a valores com erro abaixo de 0,1 % para condições normais de operação, não apresentando grande desvio da média para casos de falta no sinal.

Tabela 5.4: Características dos FPGAs.

FPGA	F1 / Altera ep2S60f672c3	F2 / Altera ep2s180f1508c5	F3 / Altera ep4sgx230kf40c2n
Blocos Lógicos	60k	180K	680k
Pinos	493	1.171	1.517
Bits de Memória	2,5M	9,3M	22,4M
DSPs	288	768	1,360
	(9x9 bits)	(9x9 bits)	(18x18 bits)
Preço	US\$ 995	US\$ 1,995	US\$ 4,495

Para efeitos de comparação, a Tabela 5.4 apresenta diferentes FPGAs da família Stratix, desenvolvidos pela Altera. As versões com um e dois indivíduos em paralelo foram testadas no FPGA F1, enquanto a versão com quatro indivíduos só foi suportada a partir do FPGA F2.

5.3.1 Componentes

A Figura 5.5 contém os principais elementos do Relé desenvolvido: *DataInput* um bloco de memória simulando a entrada de dados, os sinal gerado previamente é convertido para binário e armazenado no FPGA; *Relé* o relé em si com código referente ao AG; e por fim, *Send80bits* um bloco de saída para comunicação serial. Existem outros blocos para tratar a entrada através de botões no FPGA e exibir dados nos *displays* mas foram omitidos por não serem necessários ao funcionamento sendo usados apenas em casos de depuração.

Nos testes executados o sinal de entrada é armazenado na memória do FPGA, ele recebe como entrada um sinal da porta serial, *SerialIn*, que reinicia o sistema, um sinal de relógio, *clock*, para sincronia e um sinal de pronto, *readRequest*, para avançar para o próximo dado na memória. Como saída, retorna uma amostra do sinal, *data_out*, que é lida sequencialmente pelo relé. Numa versão prática, a memória deveria ser substituída por um Conversor Analógico Digital (ADC - *Analog-to-Digital Converter*) e o sinal viria diretamente da rede elétrica.

O relé recebe o sinal do bloco anterior e, ao receber um aviso para leitura, *read*, atualiza a janela avaliada e reinicia o processamento do AG. O sinal de relógio, *clock*, serve de sincronia e o sinal da serial reinicia o algoritmo. Após o processo o relé atualiza as saídas:

- *freq* - frequência estimada;
- *ampl* - amplitude;
- *fase* - ângulo de fase;
- *aptidao* - aptidão do melhor indivíduo.

E então, envia um sinal, *ready*, para o controlador da serial avisando o término.

O controlador recebe os dados, *entrada*, e os envia via porta serial, *SerialOut*, ao fim do envio o controlador realimenta os blocos de dados e do relé para iniciar o processo na nova janela, *done*.

Figura 5.5: Blocos principais do relé.

5.3.2 Processos

O Relé possui quatro processos trabalhando simultaneamente:

- **AG** - é o processo principal, em que os indivíduos são gerados e a aptidão é calculada;
- **GeraIndices** - responsável por ler os valores pseudoaleatórios da memória, a cada iteração lê mais dez valores que serão utilizados pelo AG para o torneio, cruzamento e mutação;
- **Timer** - responsável por contar o tempo e acordar o processo que atualiza o sinal lido;
- **InicializaSinal** - Lê o sinal de entrada e atualiza o vetor de sinal.

O AG é uma máquina de estados finitos com cinco estados, executando continuamente, o estado é reiniciado externamente a cada atualização da janela do sinal:

Figura 5.6: Máquina de estados finita do AG.

- **Inicializa** - Aqui, o AG inicia a população zero, com valores vindo de vetores com números aleatórios, este estado alterna com o estado Espera até que os 30 indivíduos tenham sido inicializados;
- **Espera** - Um ciclo é desperdiçado para que se calcule a aptidão do indivíduo;
- **Seleciona** - é feito o torneio com os indivíduos da população anterior;
- **Calcula** - cruzamento e mutação do novo indivíduo, este vai para a avaliação de aptidão;
- **Avalia** - verifica a validade dos valores, controla a troca de geração e serve como espera para calcular a aptidão.

5.4 Método de Regularização

Devido à má formação da onda no momento da queda de tensão ou sobrecarga do sistema, a função objetivo perde a capacidade de bem representar o sinal. Para contrabalançar essa mudança brusca pontual foi adicionado um sistema de regularização baseado na inércia do gerador.

Regularização é uma técnica utilizada em diferentes campos de pesquisa (problemas inversos, estatística, aprendizado de máquina), para resolver problemas de desvio na estimação ou para evitar especialização excessiva, *overfitting*, em que uma heurística se adapta extritamente a uma instância do problema. Basicamente, informação relevante do domínio do problema é adicionada à formulação do algoritmo, com o objetivo de prevenir soluções inadequadas. Essencialmente, a regularização é uma maneira formal de penalizar soluções complexas, que utilizariam parâmetros

desnecessários, favorecendo a estimação de modelos mais simples, que teoricamente se fixariam nos parâmetros indispensáveis. Restrições podem envolver suavização de arestas (Monaco et al., 2009) ou limitação da faixa de variação dos parâmetros (Golub et al., 1999).

Tais métodos são utilizados para orientar o processo de busca, para que as soluções consistentes com requisitos práticos de problema tenha preferência. Um método de regularização é em geral, baseado na manutenção de uma condição para manter um número em uma faixa aceitável de valores ou impondo suavidade para uma variação de parâmetro (Engl et al., 1996).

A regularização para estimação de frequência em SEP pode ser sintetizada pela Equação 5.11 (Courty et al., 2012).

$$\Delta f = f_0 \frac{1}{2H} \Delta t, \quad (5.11)$$

em que Δf é a variação de frequência, H é a constante de inércia do gerador, Δt é o intervalo de tempo entre amostras, e f_0 é a frequência fundamental do sistema. Com isso é possível reduzir o espaço de busca na dimensão da frequência para os valores contidos no intervalo $|f - \Delta f, f + \Delta f|$, guiando o processo de estimação para uma solução mais plausível.

5.5 Algoritmos Evolutivos Multiobjetivos Implementados

O sinal trabalhado é trifásico, possuindo três componentes defasadas um terço de ciclo uma em relação à outra. Versões iniciais do algoritmo consideravam apenas uma fase, aplicando uma transformada sobre o sinal de cada fase gerando um sinal monofásico para cada, ignorando o sinal vindo das demais. O tempo necessário para efetuar a transformada é uma desvantagem, além da perda de informação associada as outras fases. Além disso quando ocorre uma queda de fase, com a tensão indo a zero, torna-se impossível estimar a frequência baseado apenas no sinal em que ocorreu a falta.

A variação de tensão em cada fase pode ser significativamente diferente, mas não a frequência, que se mantém a mesma entre para todas as fases do sistema. Assim, executando o algoritmo e combinando o resultado das três ondas senoidais é possível fazer uma estimação de frequência mais robusta menos sujeita a variações e erros do AG. Baseado nos modelos multiobjetivos, apresentados na Seção 3.3, foram abordadas duas propostas: uma baseada na composição de uma função objetivo mais ampla e a outra baseada no AEMT. Essas propostas, ainda não totalmente implementadas em FPGA, com o gerenciamento de tabelas do AEMT em desenvolvimento, tem os resultados apresentados na Sessão 6.4.3, que foram obtidos pelo algoritmo desenvolvido em linguagem C.

5.5.1 Relé usando Método de Agregação de Funções

Para esta versão, cujo método foi apresentado na Seção 3.3.1, a representação do indivíduo é modificada, passando a ter três valores de amplitude e fase. O cálculo da frequência é feito da mesma forma, porém passa a ser usado nas três fases distintas.

Num primeiro momento, eram armazenadas as três aptidões de forma separada e usados diferentes indivíduos para gerar os pares de amplitude e fase, porém o acréscimo de complexidade não resultou em melhora significativa do algoritmo, uma vez que isso favorecia as avaliações de forma separada em detrimento do sinal trifásico como um todo.

Dessa forma, a função objetivo passou a ser composta pela soma das diferenças nas três fases do sistema

$$\begin{aligned}
 Aptidao = & \sum_{t=0}^{N-1} |Sinal_A(t) - Vm_A \cdot sen(2\pi f \frac{t}{F} + \phi_A)| \\
 & + \sum_{t=0}^{N-1} |Sinal_B(t) - Vm_B \cdot sen(2\pi f \frac{t}{F} + \phi_B)| \\
 & + \sum_{t=0}^{N-1} |Sinal_C(t) - Vm_C \cdot sen(2\pi f \frac{t}{F} + \phi_C)|,
 \end{aligned} \tag{5.12}$$

assim, mesmo que ocorra queda na tensão, essa passa a ser desconsiderada na soma das diferenças, sem a necessidade de multiplicar pela amplitude. Os resultados desse método se encontram na Seção 6.4.2.

5.5.2 Relé usando o AEMT

Com base na Seção 3.3.3, no Relé com AEMT, cada fase do sinal foi considerada um objetivo distinto, com uma tabela para armazenar os melhores resultados. A soma das três fase também foi considerada, totalizando quatro tabelas. Um indivíduo é gerado com base em valores já salvos nas tabelas, e avaliado para os quatro objetivos distintos.

A função usada no cálculo da aptidão dos diferentes objetivos é a mesma apresentada na Equação 5.2, com uma única variação de que agora as três fases do sinal são consideradas simultanea-

mente:

$$Aptidao_A = \sum_{t=0}^{N-1} |Sinal_A(t) - V_m A \cdot \text{sen}(2\pi f \frac{t}{F} + \phi_A)| \quad (5.13)$$

$$Aptidao_B = \sum_{t=0}^{N-1} |Sinal_B(t) - V_m B \cdot \text{sen}(2\pi f \frac{t}{F} + \phi_B)| \quad (5.14)$$

$$Aptidao_C = \sum_{t=0}^{N-1} |Sinal_C(t) - V_m C \cdot \text{sen}(2\pi f \frac{t}{F} + \phi_C)| \quad (5.15)$$

$$Aptidao_{soma} = \frac{1}{3} \times (Aptidao_A + Aptidao_B + Aptidao_C). \quad (5.16)$$

A função objetivo original é executada três vezes gerando quatro resultados, considerando a soma dos três já calculados. Assim um indivíduo gerado é avaliado para as diferentes tabelas. Caso o novo apresente uma aptidão melhor que um dos indivíduos já tabelados, o anterior é substituído. Ao fim da iteração, o Relé escolhe o indivíduo com melhor aptidão entre todas as tabelas.

5.6 Considerações

Este capítulo apresentou a base matemática do cálculo de estimação de frequência, a criação da estrutura de dados necessária otimizada para FPGA. Foi apresentado o algoritmo base e como este é usado para estimar a frequência do sinal, além dos procedimentos envolvendo a mudança para a próxima janela dados. Descreveu-se também a utilização do FPGA em unidades lógicas e quantidade de processamento de fato. Além disso, propõe-se das variações promissoras de AEs como relé com versões baseadas na regularização do cálculo de frequência e utilização de métodos multiobjetivos.

Resultados Experimentais

Este capítulo apresenta os resultados obtidos pelos novos relés baseados em AEs durante suas fases de desenvolvimento até se atingir a proposição mais eficiente e robusta. A Seção 6.1 trata da primeira versão, avaliando apenas uma janela de amostragem. Na Seção 6.2 é dada continuidade ao processo, e a avaliação passa a atuar com janelas deslizantes, conforme o tempo passa os dados de entrada são atualizados e o AG reinicia a cada nova janela. A Seção 6.3 apresenta os resultados da regularização evitando valores discrepantes durante o processamento. Na Seção 6.4 o algoritmo passa a considerar as três fases do sistema retornando apenas um valor de frequência. A Seção 6.5 mostra resultados para estimação usando os mesmos algoritmos utilizados na estimação de frequência. A Seção 6.7 apresenta considerações sobre os resultados obtidos listando as principais proposições de relés por AEs desta pesquisa.

6.1 Avaliação de janela independente

A primeira versão do AG tratava apenas uma janela do sinal, terminando a execução ao fim dessa janela. Os experimentos são baseados em uma forma de onda senoidal com uma frequência fundamental de 60 Hz. Todos os casos foram gerados observando-se os intervalos da Tabela 6.1, em cada caso um parâmetro era variável e os outros dois permaneciam fixos.

Erros relativos acima de 1% na estimação da frequência, não são desejados para a proteção de unidades geradoras em sistemas elétricos de potência. Portanto, as estimações obtidas frente a este conjunto de teste caracterizado, estão adequadas à necessidade do problema delineado.

De um modo geral o AG converge primeiro para os valores de amplitude e ângulo de fase, levando mais tempo para convergir para o valor correto de frequência. Isso ocorre devido a relação

Tabela 6.1: Parâmetros testados.

Parâmetro	Valores	Intervalos
Amplitude (V_m)	[2.5V, 12, 5V]	2.5V
Frequência (f)	[58.0Hz, 62.0Hz]	0.5Hz
Fase (ϕ)	[0, 360]	30°

não linear entre frequência e tempo no argumento do seno no modelo da componente fundamental do sinal elétrico. Dos três parâmetros (A , f e ϕ) a frequência é o mais difícil de estimar. O produto frequência $\times$ tempo fica ainda suscetível a precisão de t gerando dificuldade adicional na estimativa de frequência.

Assim os testes focam na avaliação da qualidade da estimativa da frequência. Para cada caso presente nas Seções 6.1.1 e 6.1.2, os testes foram executados 20 vezes e é apresentado nos gráficos a média dos resultados obtidos.

Figura 6.1: População do AG: (a) Geração 1 (b) Geração 20.

A Figura 6.1 apresenta a disposição dos indivíduos em dois instantes diferentes: na primeira geração, e em seguida na vigésima geração, a seta indica o melhor indivíduo no momento. É pos-

sível perceber que os valores de amplitude e ângulo de fase já estão definidos, e resta ao algoritmo buscar no espaço relativo a frequência, a linha formada pelos indivíduos na Figura 6.1 (b).

A frequência se mostrou o parâmetro mais vulnerável às perturbações no sinal, apresentando mínimos locais e sendo influenciada por erros na estimação dos outros parâmetros. Ao longo do desenvolvimento foram acrescentadas técnicas para tornar a frequência mais resistente e menos volúvel às perturbações.

6.1.1 Resultados em *software*

Em um primeiro momento, o algoritmo foi executado em *software* para permitir um ajuste mais prático dos parâmetros e validação dos experimentos. Os resultados desta seção são referentes a essa execução.

A Figura 6.2 mostra os erros percentuais para a estimação dos três parâmetros mediante a variação da amplitude. O AG obteve sucesso na estimação da frequência e do ângulo de fase, alcançando erros relativos menores que 0,1%. Já para a estimação da amplitude, o algoritmo apresentou um desempenho satisfatório, em que o erro relativo encontrado está em torno de 0,3%.

Figura 6.2: Variação da amplitude simulada.

Na Figura 6.3 são exibidos a variação da frequência e os respectivos erros percentuais de estimação para os parâmetros em análise. Para todos os parâmetros, a estimação fornece erros relativos menores que 0,03%.

Figura 6.3: Variação da frequência simulada.

Figura 6.4: Variação do ângulo de fase simulada.

A Figura 6.4 mostra o comportamento dos erros percentuais mediante a variação do ângulo de fase. Os erros relativos obtidos são, em geral, maiores para esse conjunto de testes. O erro máximo na estimação do ângulo de fase encontrado foi para o caso de 30° , nele o erro relativo foi de aproximadamente 0,88%.

Os bons resultados desta análise mostraram a eficiência do método, o que permitiu ajustar o algoritmo para ser executado no FPGA. Evitou-se o uso operações complexas, e a utilização de memórias para cálculo do seno mostrou-se satisfatória.

6.1.2 Resultados em *hardware*

Figura 6.5: Variação da amplitude em VHDL.

Esta etapa tratou do desenvolvimento do algoritmo proposto na linguagem VHDL. Os resultados desta etapa, são obtidos a partir do FPGA, utilizando cerca de 1 ms de tempo de processamento para fornecer a frequência, amplitude e ângulo de fase de um dado sinal.

O comportamento da estimação da frequência de um dado sinal, pode ser visualizado na Figura 6.5. Como se percebe, o ângulo de fase continua imune à variação da amplitude, com erro nulo na sua estimação. A frequência, por sua vez, mostra-se adequada frente tal variação, o que é denotado por um erro relativo máximo em torno de 0,18 %. Já a amplitude demonstra uma variação brusca entre para 12,5 V, devido a falha na representação de valores extremos do intervalo, com erro relativo percentual em 0,32 %.

Figura 6.6: Variação da frequência em VHDL.

Figura 6.7: Variação do ângulo de fase em VHDL.

Diante do principal objetivo do relé, a estimação com precisão da frequência, na Figura 6.6 tem-se que essa função ainda não foi totalmente satisfeita uma vez que os erros relativos percentuais na faixa de 0,30 % são conseguidos mediante a variação da mesma. O ângulo de fase não sofre interferência na sua estimação. Apesar da estimação da amplitude apresentar um valor um pouco acima do erro máximo almejado, 1 % para os três parâmetros em análise, constata-se sua boa estimação frente à variação da frequência.

Quanto à estimação dos três parâmetros mediante a variação do ângulo de fase, a Figura 6.7 mostra que, em alguns casos, seus erros relativos ultrapassaram o valor desejado, 1 %, atingindo um percentual em torno de 1,15 %. Para as demais variações, houve um aumento generalizado dos erros, contudo, permanecendo abaixo do 1 % estabelecido como limite base. Portanto, existe influência da variação do ângulo de fase na estimação do próprio ângulo, bem como da frequência e da amplitude puderam ser observadas.

6.2 Avaliação em tempo real

Nesta versão, o relé no FPGA apresenta novos dados a cada 1,3 ms, para obtê-los foi utilizada a interface serial e escrito um programa para leitura dos dados. Os dados obtidos foram comparados com os calculados pelo AG em *software* utilizando os mesmos parâmetros e operadores utilizados no FPGA.

Figura 6.8: AG Software x AG Hardware convergindo para o valor real.

A Figura 6.8 faz uma comparação entre a frequência do sinal original gerado pelo programa ATP (*Alternative Transients Program*)(Leuven, 1987), a frequência estimada pelo *software* simulador e a frequência calculada diretamente no FPGA, em ambos os casos o relé necessita receber os

primeiros sinais para poder formar uma janela e estimar o valor corretamente, no entanto mesmo quando o relé ainda está estabilizando ambos os resultados, do FPGA e do simulador, apresentam comportamento parecido. Após ter lido cerca de meio ciclo de onda o FPGA já estabiliza e começa a apresentar resultados com menos de 1% de erro, poucas amostras depois o simulador também estabiliza e também começa a apresentar erros menores que 1%.

As figuras 6.9 a 6.14 mostram gráficos de um sinal elétrico de 13 segundos com variação brusca de frequência e amplitude do sinal nos instantes 2 e 6 segundos devido à mudança de carga no sistema. O sinal usado nos teste foi a fornecido pelo Laboratório de Sistemas de Energia Elétrica do Departamento de Engenharia Elétrica da EESC/USP. Foram testadas versões do AG com diferentes configurações. As mudanças ocorridas são descritas conforme são mostrados os gráficos.

Figura 6.9: Janela com 8 amostras de tempo.

A Figura 6.9 mostra os resultados para relé com janelas de 8 amostras, aproximadamente meio-ciclo de 60 hertz, no item (a) o AG possui 30 indivíduos por geração calculando 1550 gerações por janela de tempo de 1,3 milissegundos. Os resultados em (b), (c) e (d) são variações no número

de indivíduos por geração o que implica num menor número de gerações no mesmo intervalo de tempo. Pode se notar pelos gráficos que aumentando o número de indivíduos o número de pontos fora da curva diminui, visto que o espaço de busca atingido em cada geração é bem maior, torna-se difícil para o AG convergir para um mínimo local.

Figura 6.10: Janela com 15 amostras de tempo.

A Figura 6.10 mostra os valores calculados em um relé que avalia um ciclo completo de onda, da mesma forma dos gráficos da Figura 6.9 foram feitas simulações com diferentes números de indivíduos por geração. Apesar da melhora na precisão do cálculo da frequência, ainda ocorrem muitos erros de máximo local conforme o AG avalia o sinal.

Figura 6.11: Janela com 8 amostras de tempo e memória do resultado anterior.

Na Figura 6.11 é feita uma pequena alteração no AG, ao invés de a cada nova janela calcular novamente os valores de frequência, amplitude e ângulo de fase, o AG utiliza os valores calculados na janela anterior como chute inicial da nova janela, assim como não ocorrem muitas variações bruscas de frequência e amplitude, e o ângulo de fase varia de forma quase constante de janela em janela, o chute inicial é bastante preciso. Com essa alteração o número de indivíduos por geração passou a ser menos significativo uma vez que o número de soluções fora da faixa esperada diminuiu consideravelmente. O mesmo vale para os gráficos da Figura 6.12 em relação aos gráficos da Figura 6.10 .

Figura 6.12: Janela com 15 amostras de tempo e memória do resultado anterior.

Na Figura 6.13, foi modificada a faixa de amplitude aceita pelo AG, que antes era de $[0.2:1.0]$, passou para $[0.75:1.0]$. A mudança ocorreu para aumentar a precisão da amplitude e também para facilitar os cálculos, anteriormente salvos em uma tabela, uma vez que o intervalo de 0.8 não é nem múltiplo nem submúltiplo de dois, o que dificultava a conversão entre a escala da amplitude e a escala do sinal, seguindo a Equação 5.6. Desse modo o novo intervalo possui variação de 0.25, sendo submúltiplo de dois facilitou a multiplicação e diminuiu o erro de arredondamento. Esta versão já apresenta resultados com mais rapidez do que relés comuns vendidos comercialmente.

Figura 6.13: Modificação na representação da amplitude.

Na última versão, Figura 6.14, foram feitas alterações na forma como é representada a frequência, ao invés de usar uma escala de 0 a 255 como foi feita nas outras simulações a frequência é representada diretamente como o ângulo gerado por numa escala de 24 *bits*. Assim os diferentes ângulos calculados conforme os instantes de tempo são submúltiplos do valor armazenado como frequência, conforme explicado na Seção 5.2.2 na Equação 5.3, isso evita perda de desempenho uma vez que antes para todo indivíduo gerado era necessário fazer multiplicações e somas para transformar a frequência em ângulo. Do modo como foi programado, calcula-se o ângulo diretamente que é convertido para o valor de frequência somente ao final do processamento de uma janela.

Nessa versão, a função objetivo já é a descrita na Seção 5.2.2, o ângulo de fase passou a ter precisão de 12 *bits* enquanto a amplitude continuou com 10 e com intervalo modificado. A frequência passou a ter 24 *bits* de precisão, porém uma parte considerável não era usada na prática pois a faixa de 55 a 65 Hz fica em um espaço menor dentro dos 24 *bits*. Assim, a última versão

Figura 6.14: Modificação na representação da frequência.

em hardware utilizava essa configuração e analisava uma janela com 15 pontos. Nas Figuras 6.15 e 6.16 seguem os resultados para a versão de 15 pontos.

A linha verde representa o valor real da frequência, a linha vermelha são os pontos calculados pelo AG e a linha azul é a frequência estimada por um relé comercial utilizando Transformada Discreta de Fourier. Na Figura 6.15, a diferença entre a frequência estimada pelo AG e a estimada pelo relé comercial não apresentam grande diferença em relação ao valor real dado pelo ATP. Num primeiro momento em que não há variação de frequência, nenhum apresenta dificuldades em sua estimação, porém aos dois segundos, quando ocorre a primeira variação de carga no sistema com queda brusca na amplitude, o AG apresenta uma variação brusca e em seguida volta a estimar corretamente a frequência enquanto o relé 81 não apresenta erro no momento da variação de carga porém começa a apresentar dificuldade em acompanhar a variação de frequência enquanto ela se estabiliza junto com a amplitude. Isso acontece enquanto a onda varia, e aos seis segundos quando há outra variação brusca de carga no sistema, o mesmo ocorre, com um pico gerado pelo AG que logo estabiliza e o relé 81 que passa a apresentar um atraso significativo em relação ao valor do

Figura 6.15: Comparação entre o AG e um relé comercial.

Figura 6.16: Comparação entre o AG e um relé comercial ampliada.

ATP. A Figura 6.16 apresenta o mesmo resultado da Figura 6.15 porém com modificação na escala do gráfico para dar uma melhor visão de como é o atraso do Relé 81 em comparação com o ATP e o AG.

6.3 Efeito da Regularização

O relé ainda apresenta picos na estimação de frequência no instante de falha, o método de regularização foi usado no FPGA para suavizar essas falhas pontuais. Foram feitos testes para constante de inércia de rotação, H , com valores 3, 6, 12 e 24 segundos, substituindo na Equação 5.11 se obtém respectivamente um Δf máximo entre janelas de $1, 3 \cdot 10^{-2}$, $6, 5 \cdot 10^{-3}$, $3, 25 \cdot 10^{-3}$ e $1, 625 \cdot 10^{-3} Hz$.

Figura 6.17: Sinal regularizado.

A regularização se mostrou eficaz em reduzir os picos e tornar o sinal mais suave, isto é com menores variações bruscas dentro da faixa de 0,1 % de erro. Porém resultados com constante de inércia maior, Figura 6.17 (c) e (d), começaram a apresentar atraso, embora menor que o relé comercial ainda significativo.

Em uma nova abordagem, utilizando diferentes amostras de sinal com 16 *bits* de precisão, foi possível obter valores mais suaves com menos atraso para $H = 6$. Dos quatro casos testados, três apresentaram resultados muito bons, Figura 6.18 (a), (c) e (d). A exceção foi o Caso B em que ocorre uma falha grave, uma das componentes do Sistema Trifásico cai totalmente, enquanto as outras duas apresentam deformação fugindo do padrão senoidal.

Figura 6.18: Sinal regularizado para amostras de 16 *bits*.

6.4 Contribuição de Modelos dos Multiobjetivos

O resultado obtido no Caso B, Figura 6.18 (b), apesar de satisfatório, é obtido apenas para uma das componentes do sistema trifásico, quando exibidas as outras componentes o resultado é o da Figura 6.19 (b), nele as três fases divergem quanto à frequência, a partir de uma queda de tensão no

instante 2s, ao contrário dos Casos A, C e D, em que mesmo com variações na tensão a estimação de frequência se manteve dentro da faixa de segurança para todos os casos.

Figura 6.19: Resultado para o sistema trifásico.

Até então, o sistema de avaliação para cada fase era independente, cada sinal sendo processado por um AG separado sem considerar os resultados das fases adjacentes. Apesar de mostrar bons resultados para casos individuais, esta abordagem é mais sujeita a erros, nos casos em que uma fase sofre queda brusca de tensão. Nas próximas seções são mostrados resultados para versões multiobjetivas do algoritmo. O Caso B é usado como exemplo pois foi o que apresentou maior variação nos resultados, o que dificulta ao relé verificar qual fase apresenta o melhor resultado a ser usado.

A Figura 6.20 considera um modelo hipotético, em que o Relé consegue adivinhar qual fase apresenta a melhor estimação de frequência. O Caso B é usado como exemplo pois foi o que

Figura 6.20: Caso B: Resultado hipotético para o sistema trifásico utilizando oráculo.

apresentou maior variação nos resultados, dificultando a escolha de qual fase apresenta o melhor resultado a ser usado.

Considerando um 'oráculo' para adivinhar qual o melhor resultado, teríamos um resultado muito próximo a curva da frequência original, praticamente sem atraso perceptível, como mostrado na Figura 6.20 (a). Diminuindo a taxa de acerto do AG Oráculo, Figura 6.20 (b), (c) e (d), o resultado vai se degradando, apresentando maiores variações na estimativa e atraso em relação ao valor real. Estes resultados, com Relé acertando entre 50% a 100%, servem de comparação para as versões multiobjetivas a seguir.

6.4.1 Heurística de Troca das Fases

A princípio, investiga-se os efeitos de uma heurística simples que considera as informações da três fases do sinal, e escolhe a solução que apresenta menor erro, dado pela função objetivo. Este resultado, Figura 6.21, é usado para comparar com resultados das heurísticas apresentadas a seguir.

Tratando cada fase como uma população diferente e ao final do processamento selecionando a população com melhor indivíduo não apresentou bom resultado, uma vez que a queda na amplitude em uma fase minimizou o erro na mesma em comparação com as demais fases. Com a amplitude em zero, o valor da frequência ficou a deriva sem convergir para nenhum valor próximo à curva original.

Figura 6.21: AG Multipopulacional simples.

Para contrabalançar o peso da amplitude, a mesma foi usada na comparação de aptidão entre indivíduos de diferentes populações. O que adicionou processamento ao operador de torneio e de

elitismo aplicado logo em seguida a função de avaliação. O resultado foi relativamente bom, visto que partindo de três estimações diferentes, foi possível obter um resultado mais próximo do valor original gerado pelo ATP, Figura 6.22.

Figura 6.22: AG Multipopulacional com amplitude ponderada.

6.4.2 AG Multiobjetivo por Função Agregação

Nesta versão, segundo o método apresentado na Seção 3.3.1 e detalhado Seção 5.5.1, as três componentes do sinal são avaliadas de modo a gerar apenas um resultado referente a um único indivíduo. Isto tornou a função objetivo mais complexa, e o custo de mudar um único parâmetro torna-se maior visto que ele afeta menos a função objetivo composta do que afetaria a função avaliando cada onda separadamente. Assim, essa versão apresenta uma convergência mais lenta

do que a versão de multipopulações, o que não chega a ser um problema na prática, dado que de modo geral o AG converge para o ótimo em tempo inferior ao limite de 1,3 ms (Tabela 5.3).

Figura 6.23: AG com função objetivo composta.

A vantagem desse método é que não é necessário usar a amplitude como peso para anular o efeito de amplitude baixa na função objetivo. Visto que é gerada uma única solução não é necessário considerar qual é a mais vantajosa, o que evita a multiplicação por amplitude nas etapas de comparação.

Dos casos de 16 *bits* apresentados na Figura 6.18, o Caso B foi o que apresentou maior dificuldade, devido tanto a queda completa em uma das fases do sinal, quanto à deformação ocorrida nas fases que sobraram. Apesar de tudo o resultado apresentado na Figura 6.23 está dentro da faixa de erro tolerável, sem adicionar complexidade extra à função objetivo, apenas adicionando operadores de soma.

6.4.3 AEMT para Relé de Frequência

O método AEMT, Seção 3.3.3 e Seção 5.5.2, apresentou bons resultados conforme mostra a Figura 6.24, para os Casos A, C e D, o relé estimou com precisão a frequência real e no Caso B ele se manteve próximo ao valor real do sinal, apesar de ocorrerem vários picos conforme a execução do algoritmo recebe dados do sinal.

Figura 6.24: AG com multiplas tabelas seguindo o modelo AEMT.

No AEMT, após a avaliação do algoritmo é escolhido a melhor fase, que apresenta o menor erro em relação ao sinal real. Isso porém pode causar erros nos casos em que uma fase apresenta amplitude significativamente menor que as outras. Adaptando a avaliação do AEMT, para considerar também a amplitude,

Figura 6.25: AEMT com amplitude ponderada.

$$Aptidao_A = \frac{1}{V_m A} \times \sum_{t=0}^{N-1} |Sinal_A(t) - V_m A \cdot \sin(2\pi f \frac{t}{F} + \phi_A)| \quad (6.1)$$

$$Aptidao_B = \frac{1}{V_m B} \times \sum_{t=0}^{N-1} |Sinal_B(t) - V_m B \cdot \sin(2\pi f \frac{t}{F} + \phi_B)| \quad (6.2)$$

$$Aptidao_C = \frac{1}{V_m C} \times \sum_{t=0}^{N-1} |Sinal_C(t) - V_m C \cdot \sin(2\pi f \frac{t}{F} + \phi_C)| \quad (6.3)$$

$$Aptidao_{Soma} = \frac{1}{3} \times (Aptidao_A + Aptidao_B + Aptidao_C), \quad (6.4)$$

da mesma forma mostrada na Seção 6.4.1, o relé manteve sua estimação bem próxima ao valor real, com menos picos que os apresentados na Figura 6.24, a Figura 6.25 foi o melhor resultado obtido até então.

6.5 TVE usando o AEMT

Para avaliação da estimação de amplitude e fase, foi utilizado o TVE, conforme descrito na Seção 2.3. Para os testes, foram utilizados dois sinais em que ocorrem variação brusca de parâmetro e logo em seguida o sinal retorna a normalidade. Em ambos os testes a frequência do sinal era de 60Hz com 16 amostras por ciclo, totalizando uma frequência de amostragem de 960Hz.

A Tabela 6.2 apresenta o tempo que cada algoritmo levou para convergir após uma variação de parâmetro. O tempo apresentado é contado do instante em que ocorreu a variação até o momento em que o TVE se estabiliza novamente com valor menor que 0,2%.

Tabela 6.2: Tempo de recuperação dos algoritmos

Variação	Tempo de Recuperação (s)		
	DFT	AG	AEMT
Fase	0.111979	0.026562	0.017708
Amplitude	0.043229	0.015104	0.012500
Frequência	0.095312	0.017708	0.156250

Os gráficos das Figuras 6.26, 6.27 e 6.28 foram calculados usando casos de variação de fase, amplitude e frequência do sinal, com os mesmos parâmetros obtidos usando a DFT, no AG mono-objetivo com regularização e no AEMT com amplitude ponderada.

A Figura 6.26 apresenta o erro de estimação no caso de uma variação de fase. No instante $t = 0.0s$, o sinal apresenta mudança de fase em meio ciclo. Simultaneamente os três estimadores perdem a qualidade da estimação e o TVE sobe para acima de 5%. Antes disso, praticamente não havia erro na estimação, apenas com pequenas variações do AG tradicional. Logo após um ciclo de 60Hz ser completo, tanto o AG quanto o AEMT já estimam corretamente a nova fase do sinal. A DFT por sua vez só se estabiliza após um décimo de segundo.

De igual modo, na Figura 6.27 ocorre um queda de 10% na amplitude do sinal. Neste caso, os três estimadores apresentam TVE maior que 5%. Após a variação, a DFT leva um tempo maior para se adaptar, enquanto as versões do AG e AEMT voltam a convergir praticamente ao mesmo tempo, assim que se completa um ciclo de onda após a queda de amplitude.

Para a variação de frequência, Figura 6.28, o AEMT apresentou dificuldade devido à regularização que impede variação brusca de frequência. Uma vez que o sinal utilizado foi criado com um simulador simples, que permite variações irreais, o método proposto levou maior tempo para convergir à frequência ideal.

Figura 6.26: TVE para variação de fase.

Figura 6.27: TVE para variação de amplitude.

Porém o AG com regularização apresentou atraso menor que o AEMT. Para investigar esse caso foram testadas diferentes versões do AEMT, variando o número de indivíduos na população, visto que por definição o AG possui mais indivíduos, 50 no AG contra 5 no AEMT. A Figura 6.28

Figura 6.28: TVE para variação de frequência.

mostra o resultado para 5, 10 e 20 indivíduos na população do AEMT, além de uma versão sem regularização usada como comparação.

6.6 Eficiência computacional do AE e AEMT

Estimações em tempo real do AE e do AEMT foram obtidas com a implementação em FPGA. Neste teste é avaliado um sinal contínuo através de um conversor analógico-digital, com taxa de amostragem de $960Hz$. Uma nova avaliação deve ser concluída em menos de $\frac{1}{960Hz} = 1.0417ms$. Cada indivíduo é avaliado em dois ciclos de $50MHz$ (usando o FPGA **EPS260F672C3** da Altera).

O gargalo em termo de poder computacional é a avaliação da função objetivo para cada indivíduo. A versão em FPGA do AE avalia até seis indivíduos em paralelo, gerando uma vazão de três indivíduos por ciclo. O AEMT necessita da avaliação de três funções objetivos para cada indivíduo, além do efeito de soma e regularização aplicados em paralelo, permitindo que a cada ciclo um novo indivíduo seja criado.

Assim o máximo número de indivíduos (N_{ind}) que podem ser avaliados em $1.04ms$ pelo AEMT é:

$$N_{ind} = \frac{6}{2 \times 3} \frac{50MHz}{960Hz} = 52,083. \quad (6.5)$$

Este número de indivíduos é suficiente para executar mais de 1.000 gerações com 50 indivíduos cada. Em fato, os resultados apresentados nas Seções anteriores usavam no máximo 25.000 indivíduos.

Tabela 6.3: Tempo de computação do AE e AEMT para $1 \leq M \leq 3$ and $1 \leq T - M \leq 2$.

Método	Números de Critério (T) e Objetivos (M)		Tempo (ms)	
			FPGA	Software
AE	$M = 1$	$T = 1$	0.166	8,690
		$T = 1$	0.166	10,207
AEMT	$1 \leq M \leq 3$	$T = 2$	0.333	12,843
		$T = 3$	0.500	14,712
	$1 \leq T - M \leq 2$	$T = 4$	0.750	14,862
		$T = 5$	0.750	15,012

A Tabela 6.3 mostra o tempo do AEMT no FPGA com a avaliação de em média um indivíduo (com 3 objetivos) por ciclo. O AEMT avalia 25.000 indivíduos retornando em 0,500ms, em que cada objetivo extra adiciona em média 0,166ms no tempo do AEMT em FPGA.

A adição de um critério ($T = M + 1$) require 1 ciclo extra por indivíduo para utilizar os três objetivos calculados anteriormente. Assim, 2 indivíduos são avaliados a cada 3 ciclos. Em consequência, o AEMT em FPGA necessita $25,000 \times \frac{3}{2}$ ciclos de 50MHz, obtendo resultados em 0.750ms. Assim, a adição de outro critério não aumenta o tempo de computação, uma vez que os novos critérios podem ser todos avaliados paralelamente, desde que exista espaço disponível no FPGA. Ficando o tempo de computação limitado a 0,750ms para $T > M$.

A Tabela 6.3 também mostra que o tempo medido para o AEMT desenvolvido em software aumenta conforme o número de critérios ($T > M$). O software escrito em linguagem C, compilado com o GCC-4.4.1 e executando em core i5 2.4GHz com 3.0 GRAM. Nota-se que o aumento para cada critério adicional é pequeno em relação ao tempo requerido para cada objetivo adicionado.

6.7 Considerações

Ao longo desse capítulo foi mostrado o desenvolvimento e resultados de AEs como novos relés de frequência e fase. Começando com a comparação dos métodos em software com os resultados para FPGA, verifica-se um aumento significativo de estimativas que são ótimos locais e margem de erro considerável. São apresentadas diversas estratégias para resolver cada problema de desempenho de AEs como relé, desde aumentar o número de indivíduos por população, mudar a representação e precisão dos dados, utilizar o resultado anterior como ponto de partida da nova janela, a inserção do cálculo da inércia de rotação da turbina, até o uso de funções compostas das três fases do sistema em modelagens multiobjetivas.

A modelagem multiobjetiva, com ênfase no AEMT, considerando as três fases simultaneamente, mostrou-se mais robusta do que o modelo original, usando conversão para sinal monofásico. Pode ser concluir que a adição de fases é benéfica ao processo de estimação. Partindo desse ponto é possível utilizar informações vindas de outros pontos da rede para aprimorar o Relé, com o sistema se comunicando através de uma Rede Elétrica Inteligente. Com a informação de onda de

diferentes lugares, é possível para o AEMT estimar a frequência com um maior número de tabelas, quatro para cada ponto de medição, além de combinações entre as diferentes tabelas. Com isso, poderia-se projetar métodos de estimação mais abrangentes, em que cada relé pode informar suas estimativas para outros relés da rede melhorando as estimativas de uma forma geral ao invés de focado em um único local. Dessa forma, tratando o sinal de diversos pontos diferentes ter-se-ia um relé inteligente no conceito de *Smart Power Grids* (Meliopoulos et al., 2011).

A melhoria na estimativa pelo AE é nítida quando comparado a um relé comercial, que não necessariamente se propõe a apresentar a solução instantaneamente. Dependendo do problema em questão, necessitando de um tratamento simultâneo da mudança de frequência, o relé desenvolvido utilizando AG em FPGA apresentou um atraso muito menor nos instantes em que a frequência do sistema apresentava mudanças bruscas. O Capítulo 7 apresenta as considerações finais sobre o projeto, os exitos alcançados além de artigos publicados.

Conclusão

A correta e rápida estimação de frequência e tensão em um sinal é vital para garantir a estabilidade e o bom funcionamento de Sistemas Elétricos de Potência. A análise do sinal como sendo um problema multiobjetivo garante uma abordagem nova ao problema. O desenvolvimento do algoritmo diretamente em FPGA permite aplicações embarcadas que lidem com o sinal em tempo real, garantindo mais segurança ao sistema. O uso de um algoritmo evolutivo permite uma maior resistência a ruídos e falhas no sinal, quando comparado aos métodos tradicionais de estimação por mínimos quadrados e transformada de fourier.

Foram desenvolvidos com sucesso diferentes versões do algoritmo, adaptadas conforme a necessidade do sistema. A utilização de hardware reconfigurável permitiu uma rápida prototipação, o que acelerou o desenvolvimento do método. A função objetivo projetada para valores inteiros permite o cálculo da aptidão, utilizando apenas uma multiplicação, para cada amostra de tempo da janela. Considerando seis indivíduos gerados em paralelos, com 15 amostras de tempo por janela, podem ser realizadas até 90 operações de multiplicação simultaneamente. Assim usando frequência de operação do FPGA em 100 MHz, tem-se o equivalente, em uma aproximação simplificada, a um processador de um núcleo operando a 9 GHz.

O uso de valores tabelados, para cálculo de números aleatórios e valores de seno, diminuiu a latência dessas operações. Isto facilitou o gerenciamento de ciclos e eliminou a necessidade de criação e gerenciamento de filas. Assim, com o algoritmo em funcionamento, tem-se a vazão de um novo indivíduo a cada dois ciclos, valor multiplicado pela paralelização do núcleo do AG.

O relé, funcionando *on-line*, executando um novo AG na frequência de amostragem, atualiza constantemente o valor estimado e, em seguida, reinicia sua operação do zero. Apesar dos resultados iniciais apresentarem, estatisticamente, uma pequena margem de erro de segurança, essa

margem tende a aumentar conforme o algoritmo executa e o erro é acumulado. Desse modo, ao partir de uma avaliação simples de uma janela, para uma avaliação janela a janela, novos métodos de controle foram necessários. Para controlar e impedir a propagação de erros foi adicionado um Método de Regularização baseado na constante de inércia de um gerador equivalente que evita uma variação brusca de frequência. Isso garantiu uma maior estabilidade e minimizou os efeitos de erro acumulativo e distúrbio no sinal.

Utilizando o AEMT, o relé passou a considerar o sinal trifásico, tanto utilizando os erros de estimativa de cada onda separadamente quando o erro obtido por meio da função agregação. Isso tornou o processo de estimativa do relé robusto a quedas de tensão e variações bruscas no sinal de uma única fase. É importante destacar que o AEMT pode ser expandido para receber informações vindas de diferentes pontos da rede, isto é, medições de sinais das três fases em cada ponto poderiam ser compartilhadas entre vários relés de forma a melhorar as estimativas de cada um dos relés da rede. Com isso, pode-se dizer que o AEMT possibilitaria utilizar uma rede de informações de maneira mais eficiente e inteligente, podendo ser uma contribuição relevante no desenvolvimento das Redes Elétricas Inteligentes.

Resultados em (Souza et al., 2007b), (de Souza, 2008), (Souza et al., 2008), (Coury et al., 2009a), (Coury et al., 2009b), (Coury et al., 2012) e (Silva, 2013) confirmaram o desempenho dos AEs desenvolvidos neste projeto de pesquisa. Por fim, ressalta-se que o paralelismo necessário foi obtido, tanto no processamento da função, quanto na geração de indivíduos. As melhorias garantiram uma adequada utilização do chip e velocidade do processamento que viabilizam AEs usados em tempo real para estimação de sinais elétricos.

Para o futuro espera-se o aprimoramento do relé executando em FPGA e a implementação de uma *smart-grid* com monitoramento em tempo real da qualidade do sinal em diversos pontos da rede. A *smart-grid* permitirá avaliar onde e quando uma falha do sistema ocorreu, permitindo uma correção mais rápida em caso de falha, garantindo assim a qualidade do serviço prestado.

Referências Bibliográficas

- ABRAHAM, A.; JAIN, L. Evolutionary multiobjective optimization. *Evolutionary Multiobjective Optimization*, p. 1–6, 2005.
- ALBA, E. *Parallel metaheuristics: a new class of algorithms*, v. 47. Wiley-Interscience, 2005.
- ASHENDEN, P. *The designer's guide to vhdl*, v. 3. Morgan Kaufmann, 2008.
- BLICKLE, T. Tournament selection. In: BÄCK, T.; FOGEL, D.; MICHALEWICZ, Z., eds. *Handbook of Evolutionary Computation*, Oxford University Press, submitted - to appear in 1997, 1997.
- BOBDA, C. *Introduction to reconfigurable computing: Architectures, algorithms, and applications*. Springer Publishing Company, Incorporated, 2007.
- BOLLEN, M. *Understanding power quality problems*. IEEE press New York, 2000.
- BROWN, S.; VRANESIC, Z. *Fundamentals of digital logic with vhdl design; 3rd ed.* Boston, MA: McGraw-Hill, 2008.
- CANTUPAZ, E. Designing efficient master-slave parallel genetic algorithms. 1997.
- DE CARVALHO, J.; COURRY, D.; JORGE, D.; DELBEM, A.; SIMÕES, E. Algoritmos genéticos aplicados à medição fasorial e projeto de pmus. *CBQEE*, p. 1Y6, 2009.
- CHEN, S.; SONG, S.; LI, L.; SHEN, J. Survey on smart grid technology. *Power System Technology*, v. 33, n. 8, p. 1–7, 2009.
- CHRISTIE, R.; BOSE, A. Load frequency control issues in power system operations after deregulation. *Power Systems, IEEE Transactions on*, v. 11, n. 3, p. 1191–1200, 1996.
- COELLO, C.; LAMONT, G.; VAN VELDHUIZEN, D. *Evolutionary algorithms for solving multi-objective problems*, v. 5. Springer-Verlag New York Inc, 2007.

- CORMEN, T. H.; LEISERSON, C. E.; RIVEST, R. L.; STEIN, C. *Introduction to algorithms, second edition*. McGraw-Hill Science/Engineering/Math, 2001.
- COURY, D.; DELBEM, A.; CARVALHO, J.; OLESKOVICZ, M.; SIMÕES, E.; BARBOSA, D.; SILVA, T. Frequency estimation using a genetic algorithm with regularization implemented in fpgas. *IEEE Transactions on Smart Grid*, v. 3, p. 1353–1361, 2012.
- COURY, D.; OLESKOVICZ, M.; DELBEM, A.; SIMÕES, E.; SILVA, T.; DE CARVALHO, J.; BARBOSA, D. Frequency relaying based on genetic algorithm using fpgas. In: *Intelligent System Applications to Power Systems. ISAP'09. 15th International Conference on*, IEEE, 2009a, p. 1–7.
- COURY, D.; OLESKOVICZ, M.; DELBEM, A.; SIMÕES, E.; SILVA, T.; DE CARVALHO, J.; BARBOSA, D. A genetic based algorithm for frequency relaying using fpgas. In: *Power & Energy Society General Meeting. PES'09. IEEE*, IEEE, 2009b, p. 1–8.
- DASH, P.; SWAIN, D.; ROUTRAY, A.; LIEW, A. An adaptive neural network approach for the estimation of power system frequency. *Electric power systems research*, v. 41, n. 3, p. 203–210, 1997.
- DAVE, N.; ARVIND; PELLAUER, M. Scheduling as rule composition. In: *MEMOCODE '07: Proceedings of the 5th IEEE/ACM International Conference on Formal Methods and Models for Codesign*, Washington, DC, USA: IEEE Computer Society, 2007, p. 51–60.
- DEB, K. *Multi-objective optimization using evolutionary algorithms*, v. 16. Wiley, 2001.
- DIAS, A.; DE VASCONCELOS, J. Multiobjective genetic algorithms applied to solve optimization problems. *Magnetics, IEEE Transactions on*, v. 38, n. 2, p. 1133–1136, 2002.
- EDWARDS, C.; PENNEY, D. *Calculus, early transcendentals*. Prentice Hall, 2008.
- EMANUEL, A.; ORR, J.; CYGANSKI, D.; GULACHENSKI, E. A survey of harmonic voltages and currents at distribution substations. *Power Delivery, IEEE Transactions on*, v. 6, n. 4, p. 1883–1890, 1991.
- ENGL, H.; HANKE, M.; NEUBAUER, A. *Regularization of inverse problems*, v. 375. Springer Netherlands, 1996.
- FOSTER, I.; KARONIS, N. A grid-enabled mpi: Message passing in heterogeneous distributed computing systems. In: *Proceedings of the 1998 ACM/IEEE conference on Supercomputing (CDROM)*, IEEE Computer Society, 1998, p. 1–11.
- FUCHS, E.; MASOUM, M.; SERVICE), S. O. *Power quality in power systems and electrical machines*. Academic Press/Elsevier, 2008.

- GIRGIS, A.; HAM, F. A new fft-based digital frequency relay for load shedding. *Power Apparatus and Systems, IEEE Transactions on*, , n. 2, p. 433–439, 1982.
- GOIS, M. M. *Representação nó-profundidade em fpga para algoritmos evolutivos aplicados ao projeto de redes de larga-escala*. Dissertação de Mestrado, ICMC-USP, 2011.
- GOLDBERG, D. Genetic algorithms in search, optimization, and machine learning. 1989.
- GOLDBERG, D. *The design of innovation: Lessons from and for competent genetic algorithms*, v. 7. Springer, 2002.
- GOLUB, G.; HANSEN, P.; O'LEARY, D. Tikhonov regularization and total least squares. *SIAM Journal on Matrix Analysis and Applications*, v. 21, n. 1, 1999.
- GRAMA, A.; KARYPIS, G.; KUMAR, V.; GUPTA, A. *Introduction to parallel computing (2nd edition)*. Addison Wesley, 2003.
- HAUPT, R.; HAUPT, S.; HAUPT, S. *Practical genetic algorithms*. Wiley Online Library, 1998.
- HOLLAND, J. Adaptation in natural and artificial systems. 1975.
- HORN, J.; NAFPLIOTIS, N.; GOLDBERG, D. A niched pareto genetic algorithm for multiobjective optimization. In: *Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on*, Ieee, 1994, p. 82–87.
- KILTS, S. *Advanced fpga design: architecture, implementation, and optimization*. Wiley-IEEE Press, 2007.
- KUNDUR, P.; BALU, N.; LAUBY, M. Power system stability and control. 1994.
- LE, T.; REJEB, J. A detailed mpi communication model for distributed systems. *Future Generation Computer Systems*, v. 22, n. 3, p. 269–278, 2006.
- L'ECUYER, P. Random number generation. *Handbook of Simulation*, p. 93–137, 1996.
- LEE, K. *Modern heuristic optimization techniques: theory and applications to power systems*, v. 39. Wiley-IEEE press, 2008.
- LEUVEN, E. Alternative transients program rule book. *Leuven EMTP Center, Belgium*, 1987.
- LIN, J. Feature extraction of machine sound using wavelet and its application in fault diagnosis. *NDT & E International*, v. 34, n. 1, p. 25–30, 2001.
- LOBOS, T.; REZMER, J. Real-time determination of power system frequency. *Instrumentation and Measurement, IEEE Transactions on*, v. 46, n. 4, p. 877–881, 1997.

- MACÊDO, R.; DA SILVA-FILHO, D.; COURY, D.; CARNEIRO, A.; DE CARVALHO, A. A genetic algorithm based approach to track voltage and current harmonics in electrical power systems. *Journal of Intelligent and Fuzzy Systems-Appl in Engineering and Technology*, v. 13, n. 2, p. 167–176, 2002.
- MARTIN, K.; HAMAI, D.; ADAMIAK, M.; ANDERSON, S.; BEGOVIC, M.; BENMOUYAL, G.; BRUNELLO, G.; BURGER, J.; CAI, J.; DICKERSON, B.; ET AL. Exploring the ieee standard c37. 118–2005 synchrophasors for power systems. *Power Delivery, IEEE Transactions on*, v. 23, n. 4, p. 1805–1811, 2008.
- MARTIN, P. A hardware implementation of a genetic programming system using fpgas and handel-c. *Genetic Programming and Evolvable Machines*, v. 2, n. 4, p. 317–343, 2001.
- MELIOPOULOS, A.; COKKINIDES, G.; HUANG, R.; FARANTATOS, E.; CHOI, S.; LEE, Y.; YU, X. Smart grid technologies for autonomous operation and control. *IEEE Trans. on Smart Grid*, v. 2, n. 1, p. 1–10, 2011.
- MONACO, F.; DELBEM, A.; FEDERSON, F. Genetic algorithm for the determination of linear viscoelastic relaxation spectrum from experimental data. *Journal of Applied Polymer Science*, v. 113, n. 1, p. 122–135, 2009.
- MÜLLER, W.; ROSENSTIEL, W.; RUF, J., eds. *Systemc: methodologies and applications*. Norwell, MA, USA: Kluwer Academic Publishers, 2003.
- MURASE, M. Linear feedback shift register. US Patent 5,090,035, 1992.
- PALNITKAR, S. *Verilog®hdl: a guide to digital design and synthesis, second edition*. Upper Saddle River, NJ, USA: Prentice Hall Press, 2003.
- PHADKE, A.; THORP, J. *Computer relaying for power systems*. Wiley Online Library, 1988.
- RERKPREEDAPONG, D.; HASANOVIC, A.; FELIACHI, A. Robust load frequency control using genetic algorithms and linear matrix inequalities. *Power Systems, IEEE Transactions on*, v. 18, n. 2, p. 855–861, 2003.
- SACHDEV, M.; GIRAY, M. A least error squares technique for determining power system frequency. *Power Apparatus and Systems, IEEE Transactions on*, , n. 2, p. 437–444, 1985.
- SANTOS, A.; DELBEM, A.; LONDON, J.; BRETAS, N. Node-depth encoding and multiobjective evolutionary algorithm applied to large-scale distribution system reconfiguration. *Power Systems, IEEE Transactions on*, v. 25, n. 3, p. 1254–1265, 2010.
- SANTOSO, S.; BEATY, H.; DUGAN, R.; MCGRANAGHAN, M. *Electrical power systems quality*. McGraw Hill, 2002.

- SASTRY, K.; GOLDBERG, D. Probabilistic model building and competent genetic programming. *GENETIC PROGRAMMING SERIES*, v. 6, p. 205–220, 2003.
- SCHAFFER, J. Multiple objective optimization with vector evaluated genetic algorithms. In: *Proceedings of the 1st international Conference on Genetic Algorithms*, L. Erlbaum Associates Inc., 1985, p. 93–100.
- SCHILDT, H.; MAYER, R. *C completo e total*. Makron Books do Brasil, 2006.
- SCOTT, L.; CLARK, T.; BAGHERI, B. *Scientific parallel computing*. Princeton University Press, 2011.
- SILVA, T. V. ; DELBEM, A. C. B. . C. D. V. Multi-objective evolutionary algorithm to estimate parameters of real-time electrical signals. *IEEE Transactions on Evolutionary Computation*, 2013.
- DE SOUZA, S. *Algoritmos genéticos aplicados à proteção e estimação de harmônicos em sistemas elétricos de potência*. Tese de Doutorado, Universidade de São Paulo, 2008.
- DE SOUZA, S.; ARARIPE DE MACEDO, R.; TAPIA VARGAS, E.; VINICIUS COURY, D.; OLESKOVICZ, M. Parameter estimation for an electric power system using genetic algorithms. *Latin America Transactions, IEEE (Revista IEEE America Latina)*, v. 4, n. 1, p. 47–54, 2006.
- SOUZA, S.; OLESKOVICZ, M.; COURY, D. Caracterização de componentes harmônicas em um sistema elétrico via algoritmos genéticos. 2007a.
- SOUZA, S.; OLESKOVICZ, M.; COURY, D.; SILVA, T.; DELBEM, A.; SIMOES, E. An efficient frequency estimation methodology using genetic algorithms in fpga. In: *Industrial Electronics Society, 2007. IECON 2007. 33rd Annual Conference of the IEEE*, IEEE, 2007b, p. 2020–2025.
- SOUZA, S.; OLESKOVICZ, M.; COURY, D.; SILVA, T.; DELBEM, A.; SIMOES, E. Fpga implementation of genetic algorithms for frequency estimation in power systems. In: *Power and Energy Society General Meeting-Conversion and Delivery of Electrical Energy in the 21st Century, IEEE*, IEEE, 2008, p. 1–6.
- STEUER, R. Multiple criteria optimization: theory, computation and application. *New York*, 1986.
- STEVENSON, W. *Elements of power system analysis*. McGraw-Hill, 1982.
- SZAFRAN, J.; REBIZANT, W. Power system frequency estimation. In: *Generation, Transmission and Distribution, IEE Proceedings-, IET*, 1998, p. 578–582.
- USIDA, W.; COURY, D.; FLAUZINO, R.; SILVA, I. Alocação de dispositivos indicadores de faltas em alimentadores primários usando algoritmos genéticos. 2010.

- VARGAS, E.; SOUZA, S.; OLESKOVICZ, M.; COURY, D. Aplicação de algoritmos genéticos em relés digitais de frequência. In: *Sixth Latin-American Congress on Electricity Generation and Transmission*, 2005.
- VOLDER, J. The cordic trigonometric computing technique. *Electronic Computers, IRE Transactions on*, , n. 3, p. 330–334, 1959.
- ZITZLER, E.; LAUMANN, M.; THIELE, L.; ZITZLER, E.; ZITZLER, E.; THIELE, L.; THIELE, L. Spea2: Improving the strength pareto evolutionary algorithm. 2001.