

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO CEARÁ
PRÓ-REITORIA DE PESQUISA, PÓS-GRADUAÇÃO E INOVAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Leonardo da Silva Costa

**Novas técnicas de poda no classificador Floresta de Caminhos
Ótimos via Algoritmos Genéticos**

Fortaleza – CE

3 de novembro de 2020

Leonardo da Silva Costa

Novas técnicas de poda no classificador Floresta de Caminhos Ótimos via Algoritmos Genéticos

Dissertação submetida ao Programa de Pós-Graduação em Ciência da Computação do Instituto Federal de Educação, Ciência e Tecnologia do Ceará, como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Linha de Pesquisa: Inteligência Artificial

Orientador: Prof. Dr. Ajalmar Rêgo da Rocha Neto

Instituto Federal de Educação, Ciência e Tecnologia do Ceará (IFCE)

Pró-Reitoria de Pesquisa, Pós-Graduação e Inovação (PRPI)

Programa de Pós-Graduação em Ciência da Computação (PPGCC)

Fortaleza – CE

3 de novembro de 2020

Dados Internacionais de Catalogação na Publicação
Instituto Federal do Ceará - IFCE
Sistema de Bibliotecas - SIBI
Ficha catalográfica elaborada pelo SIBI/IFCE, com os dados fornecidos pelo(a) autor(a)

- C837n Costa, Leonardo da Silva.
Novas técnicas de poda no classificador Floresta de Caminhos Ótimos via Algoritmos Genéticos /
Leonardo da Silva Costa. - 2020.
111 f. : il. color.
- Dissertação (Mestrado) - Instituto Federal do Ceará, Mestrado em Ciência da Computação, Campus
Fortaleza, 2020.
Orientação: Prof. Dr. Ajalmar Rêgo da Rocha Neto.
1. Floresta de Caminhos Ótimos. 2. Métodos de Poda. 3. Algoritmos Genéticos. I. Título.
CDD 004
-



Instituto Federal de Educação, Ciência e Tecnologia do Ceará (IFCE)
Pró-Reitoria de Pesquisa, Pós-Graduação e Inovação (PRPI)
Programa de Pós-Graduação em Ciência da Computação (PPGCC)

Leonardo da Silva Costa

Esta dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação, sendo aprovada pela Coordenação do Programa de Pós-Graduação em Ciência da Computação do Instituto Federal de Educação, Ciência e Tecnologia do Ceará e pela banca examinadora:

**Orientador: Prof. Dr. Ajalmar Rêgo da Rocha
Neto**

Instituto Federal de Educação, Ciência e Tecnologia do
Ceará (IFCE)

Profª. Dra. Ananda Lima Freire

Instituto Atlântico (IA)

Prof. Dr. Geraldo Luis Bezerra Ramalho

Instituto Federal de Educação, Ciência e Tecnologia do
Ceará (IFCE)

Fortaleza – CE
3 de novembro de 2020

*Dedico este trabalho a minha irmã, Lindovânia,
pela motivação integral durante esta jornada.*

Agradecimentos

Durante toda esse jornada como estudante de mestrado em ciência da computação, tive o privilégio de ter a experiência acadêmica focada na pesquisa, bem como no desenvolvimento de novas tecnologias, onde pude adquirir conhecimento científico, como também aprendi lições para a vida profissional e pessoal. Acabei me tornando uma pessoa mais madura ao compreender melhor o significado das obrigações e *deadlines* a cumprir. Nesse contexto, tais conquistas e experiências não seriam concretizadas se não fosse o apoio das pessoas envolvidas diretamente ou indiretamente nessa empreitada.

Nesse sentido, gostaria de agradecer primeiramente a Deus pela sabedoria, aos meus pais, Maria Liduina e João Almeida, que mesmo sem possuir boas condições estruturais a nível financeiro e educacional, desempenharam todos os esforços para me oferecer uma educação básica de qualidade. Aproveito esse momento também para fazer um agradecimento em especial a minha irmã, Lindovânia, que desde muito cedo sempre me incentivou nos estudos, sendo que no ensino médio tive a honra de aprender com ela. Estendo os agradecimentos ao meu irmão, conhecido como Dodó, com quem tenho a oportunidade de dividir a companhia, principalmente nos momentos de descontração após um dia cansativo de trabalho. Devo satisfação também a todos os meus familiares que me ajudaram de alguma forma no desenvolvimento pessoal e profissional.

Acolher um estudante curioso para iniciar no mundo da pesquisa não é uma tarefa fácil, pois é necessário empenho e dedicação por parte do orientador para com o aluno. Portanto, dedico agradecimento especial ao professor orientador Ajalmar Rocha, por depositar confiança em mim e por ter dedicado esforços nas contribuições desta dissertação. Além disso, sou grato por todo o corpo docente e envolvidos nesta causa, que abraçam a ideia do Programa de Pós-Graduação em Ciência da Computação (PPGCC) para que ele esteja ativo e formando bons profissionais até hoje.

Agradeço à Winnie por todo o acolhimento e paciência durante o período do mestrado, com quem sempre pude contar incondicionalmente tanto nos momentos difíceis, como em épocas de alegria.

Não poderia deixar de reconhecer também meu legítimo agradecimento ao Gabriel, companheiro de pesquisa com quem tive a felicidade de aprender e desvendar nossas tecnologias. Estendo meu agradecimento pela sua análise e sugestão minuciosa na revisão da dissertação. Por fim, e não menos importante, gostaria de agradecer também a todos os meus companheiros de turma, disciplina e laboratório durante a graduação e mestrado, no qual tive o privilégio de compartilhar experiências acadêmicas e pessoais.

*“A tarefa não é tanto ver aquilo que ninguém viu,
mas pensar o que ninguém ainda pensou
sobre aquilo que todo mundo vê.”
(Arthur Schopenhauer)*

Resumo

O classificador Floresta de Caminhos Ótimos (*Optimum Path Forest*, OPF), baseado em grafo completo, tem sido aplicado de forma efetiva em diversos problemas de reconhecimento de padrões. Esse classificador é baseado em um subconjunto de protótipos e numa função de custo de caminho em um grafo representado pelo conjunto de treinamento, onde esses dados são particionados em uma coleção de árvores de caminhos ótimos (*Optimum Path Trees*, OPTs), cada qual formada por nós mais fortemente conectados. Essa abordagem resulta em um classificador livre de parâmetros que não faz suposições sobre a dispersão ou separabilidade das classes, lida com múltiplas classes sem modificações ou extensões e não apresenta erro de classificação no conjunto de treinamento.

Apesar de possuir tais vantagens, uma limitação do OPF é que o tempo necessário para executar o seu estágio de classificação cresce proporcionalmente ao tamanho do conjunto de treinamento, visto que, para classificar uma nova amostra, é necessário conectá-la com todos os nós do grafo de treinamento. Dessa forma, o uso de técnicas de poda no grafo do classificador OPF torna-se uma alternativa atraente para reduzir o custo computacional na tarefa de classificação.

Nesse contexto, são propostas três técnicas de poda para o classificador OPF utilizando algoritmos genéticos (*Genetic Algorithms*, GAs) neste trabalho: (i) GAP-OPF, que utiliza GAs para selecionar e remover amostras menos representativas do conjunto de treinamento do OPF que seriam utilizadas em sua fase de classificação; (ii) FSGAP-OPF, que assim como o GAP-OPF, visa gerar um conjunto de treinamento mais compacto e representativo, ao utilizar uma modelagem de poda de tamanho fixo; e por último, (iii) MOGAP-OPF que utiliza otimização multiobjetivo guiada não só pela taxa de erro na classificação, mas também pelo número de OPTs.

O desempenho obtido pelos métodos propostos é avaliado por aplicá-los em bases de dados obtidas de experimentos reais, e compará-los com outras técnicas de poda presentes na literatura. As principais métricas utilizadas para comparação foram: acurácia medida sobre o conjunto de teste, percentual de redução no conjunto de treinamento (taxa de poda) e tempo de classificação necessário para classificar todas as amostras de teste. Com base nos experimentos, pode-se afirmar que GAP-OPF, FSGAP-OPF e MOGAP-OPF são alternativas válidas para tarefas de classificação.

Palavras-chave: Floresta de Caminhos Ótimos, Métodos de Poda, Algoritmos Genéticos, GAP-OPF, FSGAP-OPF e MOGAP-OPF.

Abstract

The Optimum-Path Forest classifier, based on complete graph, has been effectively applied in many pattern recognition problems. This classifier is based on a subset of prototype samples and on a path-cost function in a graph represented by the training set, in which this data is partitioned into a collection of optimum-path trees (OPTs), each one of them formed by the most strongly connected nodes. This approach results in a parameter free classifier that does not make assumptions about the dispersion or separability of the classes, handles multiple classes without modifications or extensions and achieves zero classification errors on the training set.

Despite having the aforementioned advantages, one limitation of OPF is that the time necessary to execute its classification process grows proportionally to training set, since, to classify a new sample, it is necessary to connect it with all nodes in the training graph. That being said, the application of pruning techniques on the training graph of the OPF classifier becomes an interesting approach, in order to reduce the its computational cost of its classification task.

In this context, three pruning techniques based on genetic algorithms (GAs) are proposed in this work: (i) GAP-OPF, which uses GAs to select and remove the less representative samples of the OPF classifier training set that would be used in its classification stage; (ii) FSGAP- OPF, that just as GAP-OPF, aims to generate a training set more compact and representative, using a fixed-size pruning approach; finally, (iii) MOGAP-OPF that uses multiobjective optimization technique that is guided not only by the classification error rate, but also on the number of OPTs.

The performance obtained by the proposed methods is assessed by applying them to databases obtained from real experiments, and comparing them with other pruning techniques found on the literature. The main metrics utilized for comparison were: classification accuracy over the test set, reduction percentage of the training set (pruning rate) and classification time spent to classify all test samples. According to the experiments, one can state that GAP-OPF, FSGAP-OPF and MOGAP-OPF are valid alternatives for classification tasks.

Keywords: Optimum-Path Forest, Pruning Methods, Genetic Algorithms, GAP-OPF, FSGAP-OPF and MOGAP-OPF.

Lista de ilustrações

Figura 1 – (a) Grafo ponderado completo gerado a partir de um determinado conjunto de treinamento. (b) MST construída com base nos padrões de treinamento. (c) Protótipos selecionados como sendo os elementos adjacentes de diferentes classes na MST (nós circulados). (d) Floresta de caminhos ótimos resultante da aplicação da função de custo de caminho $fmax$. A seta indica o nó predecessor no caminho ótimo. As entradas (x, y) ao lado dos nós representam, respectivamente, o custo e o rótulo das amostras.	36
Figura 2 – Exemplo do processo de classificação. O primeiro passo é (a) conectar (linhas tracejadas) uma amostra de teste (losango cinza) com todos os nós de treinamento. Depois disso, (b) t recebe o caminho ótimo do seu nó mais fortemente conectado (custo de classificação de 0,6), bem como o seu rótulo (círculo).	39
Figura 3 – Processo evolutivo do GA.	45
Figura 4 – Exemplo de representações em GAs.	47
Figura 5 – Método da roleta para uma população com 4 indivíduos.	49
Figura 6 – Seleção por torneio considerando 4 indivíduos para agrupamento em pares.	50
Figura 7 – Cruzamento uniponto.	51
Figura 8 – Cruzamento multi-ponto.	52
Figura 9 – Cruzamento uniforme.	52
Figura 10 – Cruzamento <i>Order Crossover</i> para cromossomos de codificação inteira.	53
Figura 11 – Mutação <i>bit-flip</i>	54
Figura 12 – Mutação <i>swap</i>	55
Figura 13 – Processo de transformação entre codificações binária-inteira para aplicação do operador <i>Order Crossover</i> (Seção 3.6.2.4) seguido pela transformação de codificação inteira-binária.	68
Figura 14 – Exemplo de representação genética para o MOGAP-OPF considerando um indivíduo com 4 genes, ou seja, 4 OPTs. Inicialmente, é apresentado (a) o modelo sem poda e, em seguida, (b) o indivíduo com metade das OPTs podadas pelo processo de otimização do AG.	70
Figura 15 – Mapa de cores para a base de dados <i>Iris Flower</i> . Em cada linha, são apresentados os três métodos OPF, OPF-TPPA e GAP-OPF, bem como suas distribuições para cada par de atributos.	80

Figura 16 – Mapa de cores para a base de dados <i>Iris Flower</i> . Cada linha dispõe dos três métodos de poda OPF-TPPA, ROB-OPF e FSGAP-OPF com $\alpha = 0,9$, bem como suas distribuições para cada par de atributos. . . .	83
Figura 17 – Acurácia, taxa de poda e tempo de classificação obtidos pelos métodos empregados neste trabalho aplicados nos conjuntos de dados apresentados na Tabela 6.	86
Figura 18 – Mapa de cores para a base de dados <i>Haberman's Survival</i> . Os métodos OPF, OPF-TPPA, ROB-OPF e MOGAP-OPF são apresentados, bem como suas distribuições para cada par de atributos.	88
Figura 19 – Comparação entre os métodos empregados nesse trabalho para as métricas acurácia e taxa de poda com α percorrendo o intervalo entre $[0,8;1]$, quando aplicados nas bases de dados <i>Balance Scale</i> , <i>Breast Cancer Wisconsin</i> e <i>Credit Approval</i>	90
Figura 20 – Comparação entre os métodos empregados nesse trabalho para as métricas acurácia e taxa de poda com α percorrendo o intervalo entre $[0,8;1]$, quando aplicados nas bases de dados <i>Haberman's Survival</i> , <i>Pima Indians Diabetes</i> e <i>Vertebral Column</i>	91
Figura 21 – Convergência da solução objetivo do GA para 150 gerações.	108
Figura 22 – Convergência da solução objetivo do GA para 150 gerações.	109

Lista de tabelas

Tabela 1	–	Significado dos termos em GAs.	46
Tabela 2	–	Resumo geral dos parâmetros empregadas nas abordagens propostas. .	77
Tabela 3	–	Lista das bases de dados avaliadas nesta proposta.	79
Tabela 4	–	Valores de acurácia e desvio padrão para os classificadores OPF, K -NN e MLP com metade da base (50%) utilizada para treino e o restante (50%) usado para teste.	79
Tabela 5	–	Resultados para OPF antes da poda (OPF), OPF-TPPA e GAP-OPF com 60%, 20% e 20% do conjunto de dados particionados para treinamento, validação e teste, respectivamente. O valor da métrica #CT está em segundos.	82
Tabela 6	–	Lista das bases de dados avaliadas nesta proposta.	83
Tabela 7	–	Resultados obtidos nesta proposta. Os valores das métricas α , ACC e #PR estão em porcentagem e #CT em segundos.	85
Tabela 8	–	Lista das bases de dados avaliadas nesta proposta.	87
Tabela 9	–	Resultados para OPF, OP-TPPA, ROB-OPF e MOGAP-OPF. Os valores das métricas ACC e PR estão em porcentagem e #CT em segundos.	92

Lista de Algoritmos

1	Algoritmo de Treinamento para o Classificador OPF com grafo completo .	38
2	Algoritmo de classificação para o classificador OPF com grafo completo . .	40
3	Algoritmo OPF-TPPA	60
4	Algoritmo de aprendizagem	61
5	Algoritmo de aprendizagem com poda	62
6	Algoritmo GAP-OPF	65
7	Algoritmo FSGAP-OPF	69
8	Algoritmo MOGAP-OPF	72

Lista de abreviaturas e siglas

ANN	Redes Neurais Artificiais
CX	<i>Crossover</i> Circular
DTs	Árvores de Decisão
ELM	<i>Extreme Learning Machine</i>
ERM	<i>Empirical Risk Minimization</i>
FN	Falsos Negativos
FP	Falsos Positivos
FSGAP-OPF	<i>Fixed-Size Genetic Algorithm for Pruned OPF</i>
GAP-OPF	<i>Genetic Algorithms for Pruned OPF</i>
GAs	Algoritmos Genéticos
IA	Inteligência Artificial
IFT	<i>Image Foresting Transform</i>
KNN	<i>K</i> Vizinhos mais Próximos
LSSVM	Máquinas de Vetores-Suporte por Mínimos Quadrados
ML	Aprendizagem de Máquina
MLP	Perceptron Multicamadas
MOGAP-OPF	<i>Multi-Objective Genetic Algorithm for Pruned OPF</i>
MST	Árvore Geradora Mínima
OPF	Floresta de Caminhos Ótimos
OPF-TPPA	<i>OPF Training Patterns Pruning Algorithm</i>
OPTs	Árvores de Caminhos Ótimos
OX	<i>Order Crossover</i>
PMX	<i>Crossover</i> Parcialmente Mapeado

PR	Reconhecimento de Padrões
ROB-OPF	<i>Robust Pruning of Training Patterns for OPF</i>
SRM	<i>Structural Risk Minimization</i>
SVM	Máquinas de Vetores Suporte

Lista de símbolos

Z	Conjunto de dados
Z_1	Conjunto de treinamento
Z'_1	Novo conjunto de treinamento podado
Z_2	Conjunto de validação
Z_3	Conjunto de teste
s	Amostra pertencente ao conjunto Z
v	Amostra pertencente ao conjunto Z
$d(s, t)$	Distância entre dois padrões s e t
G^{tr}	Grafo completo a partir dos dados de treinamento
A	Conjunto de arestas do grafo completo
π_s	Caminho de nós adjacentes com destino em s
S	Conjunto de protótipos
$fmax(.)$	Função de custo de caminho
L	Mapa de rótulos obtido pelo modelo
λ	Rótulos verdadeiros
P	Mapa de nós predecessores no grafo
C	Mapa de custos
v	Amostra pertencente ao conjunto de treinamento Z_1
v^*	Amostra que conquista um nó t_i
p_i	Probabilidade do i -ésimo indivíduo na roleta
i_i	<i>Fitness</i> absoluto do i -ésimo indivíduo na roleta
μ_t	Média das aptidões dos indivíduos na geração t
$a_{i,t}$	Aptidão do indivíduo i da geração t

ϵ	Erro
$f_i(\cdot)$	A i -ésima função de otimização
w_i	Peso da i -ésima função de otimização
LM	Lista de nós mal classificados
R	Lista de nós relevantes
I	Lista de nós irrelevantes
g_i	Gene na i -ésima posição do cromossomo
T_c	Quantidade de padrões restantes após a poda
$f(g_i)$	Quantidade de nós da i -ésima OPT
t_{max}	Número máximo de gerações
α	Associado a poda nos classificadores FSGAP-OPF e MOGAP-OPF
acc	Acurácia
x_i	Padrão i
$x_{i_{norm}}$	Padrão i normalizado
μ_x	Média dos vetores de atributos
σ_x	Desvio padrão dos vetores de atributos
$e_{.,.}$	Parcela de erro
$E(i)$	Erro parcial da classe i

Sumário

1	INTRODUÇÃO	29
1.1	Objetivos Geral e Específicos	31
1.2	Produção científica	31
1.3	Estrutura da Dissertação	32
2	FLORESTA DE CAMINHOS ÓTIMOS	35
2.1	OPF baseado em grafo completo	35
2.1.1	Treinamento	36
2.1.2	Classificação	39
2.1.3	Análise de complexidade	40
2.2	Conclusão	41
3	ALGORITMOS GENÉTICOS	43
3.1	Introdução	43
3.2	Algoritmos genéticos	44
3.3	Representação genética	46
3.4	População	47
3.5	Função de avaliação	48
3.6	Operadores genéticos	48
3.6.1	Métodos de seleção	48
3.6.1.1	Seleção por roleta	48
3.6.1.2	Seleção por torneio	49
3.6.1.3	Seleção por elitismo	50
3.6.2	Técnicas de reprodução	50
3.6.2.1	Cruzamento de único ponto	51
3.6.2.2	Cruzamento por multi-pontos	51
3.6.2.3	Cruzamento uniforme	52
3.6.2.4	<i>Order Crossover</i>	52
3.6.3	Estratégias de mutação	54
3.6.3.1	Mutação <i>bit-flip</i>	54
3.6.3.2	Mutação <i>swap</i>	55
3.7	Crítérios de parada	55
3.8	Otimização mono e multiobjetivo em GAs	56
3.9	Conclusão	57
4	ABORDAGENS DE PODA NO OPF	59

4.1	Introdução	59
4.2	Métodos presentes no estado da arte	59
4.2.1	OPF-TPPA	59
4.2.2	ROB-OPF	60
4.3	Técnicas baseadas em GAs	63
4.3.1	GAP-OPF (Proposta 1)	63
4.3.1.1	Indivíduos ou cromossomos	64
4.3.1.2	Função de <i>fitness</i>	64
4.3.1.3	Operadores genéticos	65
4.3.1.4	Algoritmo GAP-OPF	65
4.3.2	FSGAP-OPF (Proposta 2)	66
4.3.2.1	Indivíduos ou cromossomos	66
4.3.2.2	Função de <i>fitness</i> e método de seleção	67
4.3.2.3	Reprodução	67
4.3.2.4	Mutação	68
4.3.2.5	Algoritmo FSGAP-OPF	68
4.3.3	MOGAP-OPF (Proposta 3)	69
4.3.3.1	Representação genética	69
4.3.3.2	Função de <i>fitness</i>	70
4.3.3.3	Operadores genéticos	71
4.3.3.4	Algoritmo MOGAP-OPF	71
4.4	Conclusão	72
5	SIMULAÇÕES COMPUTACIONAIS	75
5.1	Metodologia	75
5.2	Resultados	77
5.2.1	Para o classificador GAP-OPF (Proposta 1)	78
5.2.1.1	Efetividade do Algoritmo Floresta de Caminhos Ótimos	79
5.2.1.2	Efetividade do GAP-OPF	79
5.2.2	Para o classificador FSGAP-OPF (Proposta 2)	82
5.2.3	Para o classificador MOGAP-OPF (Proposta 3)	86
5.2.3.1	Efetividade do MOGAP-OPF	88
5.3	Conclusão	93
6	CONCLUSÕES E TRABALHOS FUTUROS	95
6.1	Conclusões	95
6.2	Contribuições deste trabalho	96
6.3	Trabalhos futuros	97

REFERÊNCIAS 99

APÊNDICES 105

APÊNDICE A – EVOLUÇÃO DO GA PARA O MOGAP-OPF . . . 107

A.1 Resultados alcançados 107

1 Introdução

Inteligência Artificial (IA) é um ramo de pesquisa que tem atraído bastante interesse por muitos estudiosos das mais diversas áreas do conhecimento científico e tecnológico (NILSSON, 2014; DOMINGOS, 2015; RUSSELL; NORVIG, 2016), tais como medicina (RAMESH et al., 2004), biologia (HOLLAND, 1992), matemática (PEARL, 1984), dentre outras (KORBICZ et al., 2012; O'HARE; JENNINGS; JENNINGS, 1996; LU et al., 2018). Seu principal foco está em desenvolver mecanismos e tecnologias que buscam reproduzir o comportamento humano para tomada de decisões mais inteligentes a fim de resolver problemas de propósito geral. Dentre os subtemas da IA, destacam-se: reconhecimento de padrões (*Pattern Recognition*, PR), aprendizagem de máquina (*Machine Learning*, ML), redes neurais artificiais (*Artificial Neural Networks*, ANN), dentre outras.

PR é um ramo de ML que tem como foco a classificação de objetos em categorias ou classes. Esses objetos podem ser imagens, sinais em forma de onda ou qualquer outro tipo de medida que necessite ser classificada (THEODORIDIS, 2008). Um classificador de padrões, portanto, pode ser descrito como um elemento de *software* que mapeia dados de entrada para uma categoria ou classe pertencente a um conjunto finito de possibilidades. Muitas aplicações têm sido desenvolvidas com a ajuda de classificadores de padrões, tais como reconhecimento de fala (GRAVES; MOHAMED; HINTON, 2013), diagnóstico médico (ALICKOVIC; SUBASI, 2016), previsão do tempo (HOSSAIN et al., 2015) e filtragem de *spams* em e-mails (BHOWMICK; HAZARIKA, 2018).

Algoritmos de segmentação de imagens em alguns casos podem ser modificados de modo a lidar com problemas de PR. Como um exemplo, pode-se citar o classificador de padrões conhecido como Floresta de Caminhos Ótimos (*Optimum-Path Forest*, OPF), que foi inspirado por uma técnica de segmentação de imagens, mais especificamente pelo algoritmo *Image Foresting Transform* (IFT), proposto por Falcão, Stolfi e Lotufo (2004), e posteriormente modificado a fim de lidar com problemas de classificação (PAPA; FALCAO; SUZUKI, 2009).

Existem algumas derivações do classificador OPF, em que podem ser aplicadas tanto para resolver problemas supervisionados como não supervisionados (PAPA et al., 2008b). No âmbito dos problemas supervisionados, existem duas versões do OPF, sendo uma baseada em um grafo k -NN (PAPA; FALCÃO, 2008) e a outra em um grafo completo (PAPA; FALCAO; SUZUKI, 2009). Levando em consideração a simplicidade e difusão do modelo, optou-se por utilizar a versão do OPF baseado em grafo completo para a criação das propostas.

O classificador OPF, baseado em grafo completo, reduz o problema de PR àquele

de otimização combinatória, por utilizar Teoria dos Grafos. Nessa abordagem, as amostras são modeladas como sendo os nós do grafo e as arestas são definidas por uma dada relação de adjacência e ponderadas de acordo com a distância entre os nós. A ideia é particionar o grafo de modo que amostras de uma classe fiquem em uma partição e amostras de outras classes fiquem em partições diferentes. Essas partições podem ser vistas como árvores de caminhos ótimos (*Optimum-Path Trees*, OPTs), cada qual enraizada por amostras chaves (protótipos). A classificação de novas amostras consiste em conectá-las com todos os nós t das OPTs, como se elas fizessem parte do grafo de treinamento, e atribuindo a elas o rótulo do nó t que minimiza os seus custos.

Além disso, o OPF tem sido aplicado efetivamente em muitos problemas, tais como na identificação de doenças pulmonares (FILHO et al., 2019), classificação de sinais de eletrocardiograma (LUZ et al., 2013), detecção de invasão em redes de computadores (PEREIRA et al., 2012), diagnóstico de epilepsia (NUNES et al., 2014), dentre outros (PAPA et al., 2007; MONTOYA-ZEGARRA et al., 2007; PAPA; FALCAO; SUZUKI, 2009). Algumas vantagens do classificador OPF é o fato dele ser livre de parâmetros em algumas versões, não fazer suposições sobre a forma e a separabilidade das classes, lidar com múltiplas classes sem modificações ou extensões e não apresentar erro de classificação no conjunto de treinamento, sem *overfitting* (PAPA et al., 2008a). No entanto, uma desvantagem desse classificador é que o custo computacional da sua fase de classificação cresce proporcionalmente ao tamanho do conjunto de treinamento. Dessa forma, técnicas de poda no conjunto de treinamento do classificador OPF podem ser adotadas para obter bases de dados mais compactas e representativas, visando reduzir o custo computacional da sua fase de classificação (PAPA; FALCÃO, 2009; PAPA et al., 2010; PAPA et al., 2012; RODRIGUES et al., 2014).

Existem alguns trabalhos anteriores que abordaram o problema de reduzir a grande quantidade de amostras usadas no estágio de classificação do OPF. Por exemplo, Papa e Falcão (2009) desenvolveram um método chamado **Algoritmo de Poda de Padrões de Treinamento do OPF** (*OPF Training Patterns Pruning Algorithm*, OPF-TPPA) capaz de gerar um classificador mais rápido por identificar e remover amostras irrelevantes do conjunto de treinamento, enquanto mantém a acurácia do classificador em um nível similar. Uma melhoria dessa técnica, chamada **Poda Robusta de Padrões de Treinamento do OPF** (*Robust Pruning of Training Patterns for OPF*, ROB-OPF) (PAPA et al., 2010), foi feita ao introduzir uma fase de aprendizagem que é usada para auxiliar um processo de transferência de amostras irrelevantes com origem no conjunto de treinamento para um conjunto de validação, até que não haja mais amostras irrelevantes a serem podadas. Esse estágio de aprendizagem é executado durante um determinado número de iterações. Posteriormente, Nakamura et al. (2011) consideraram o problema de poda do OPF como um problema de otimização mono-objetivo, cujo o objetivo é encontrar o valor de um parâmetro de poda que controla a perda desejada em acurácia com relação ao conjunto

final de treinamento podado.

Além do mais, o uso de meta-heurísticas, tais como algoritmos genéticos (*Genetic Algorithms*, GAs), têm sido empregadas de forma eficaz para reduzir a complexidade computacional de muitos classificadores, a saber: redes neurais artificiais (*Artificial Neural Networks*, ANN) (WHITLEY; BOGART, 1990; ALENCAR; ROCHA NETO; GOMES, 2016), máquinas de vetores suporte por mínimos quadrados (*Least Square Support-Vector Machines*, LSSVM) (SILVA; ROCHA NETO, 2015) e árvores de decisão (*Decision Trees*, DTs) (CHEN; WANG; ZHAI, 2009). Além disso, as capacidades de GAs podem ser estendidas para uma ampla gama de aplicações de ML, como otimização de hiperparâmetros de classificadores (LAM et al., 2001) e seleção de atributos (HUANG; WANG, 2006). Devido às suas características subjacentes, GAs podem ser usados para resolver problemas de otimização sem a suposição de linearidade, continuidade, diferenciabilidade ou convexidade da função objetivo (ALENCAR; ROCHA NETO; GOMES, 2016).

1.1 Objetivos Geral e Específicos

Como arcabouço desta dissertação, almeja-se aproveitar as vantagens oferecidas pelos GAs com o objetivo de identificar e eliminar padrões, protótipos, OPTs, dentre outros atributos pertinentes ao classificador OPF, que afetam o desempenho do classificador em termos de tempo de classificação e taxa de acerto, ou até mesmo que aumentam a complexidade e dificultam a capacidade de aprendizagem e generalização do modelo.

Como resultado, é possível obter um classificador mais rápido e representativo que também pode ser aplicado em problemas onde um baixo consumo de memória é crucial.

Nesse contexto, os seguintes objetivos específicos são levantados:

1. propor novos métodos de poda para o OPF utilizando GAs;
2. manter ou melhorar o desempenho do classificador OPF em termos de acurácia, mesmo após aplicada a redução no conjunto de treinamento;
3. comparar técnicas de poda presentes na literatura aplicadas ao OPF com os métodos propostos nesta dissertação.

1.2 Produção científica

Durante este projeto de mestrado, os seguintes trabalhos científicos foram aceitos e publicados:

- BARBOSA, G. S; **COSTA, L. S.**; ROCHA NETO, AJALMAR R. A New Genetic Algorithms-Based Pruning Method for Optimum-Path Forest. In: 7th

Brazilian Conference on Intelligent Systems (BRACIS), 2018, IEEE, p. 13–18. Disponível em: <<https://doi.org/10.1109/BRACIS.2018.00011>>

- **COSTA, L. S.**; BARBOSA, G. S.; ROCHA NETO, AJALMAR R. A Fixed-Size Pruning Approach for Optimum-Path Forest. IN: 15th International Work-Conference on Artificial Neural Networks (IWANN), 2019, Springer, p. 723–734. Disponível em: <https://dx.doi.org/10.1007/978-3-030-20518-8_60>

1.3 Estrutura da Dissertação

O restante desta dissertação é composta por 5 capítulos: Floresta de Caminhos Ótimos, Algoritmos Genéticos, Abordagens de poda no OPF, Simulações Computacionais e, por fim, Conclusões e Trabalhos Futuros.

No Capítulo 2, será apresentada a fundamentação teórica do Algoritmo Floresta de Caminhos Ótimos (OPF), assunto que será abordado como fonte de pesquisa nesta dissertação. Além do mais, serão exemplificados os procedimentos de aprendizagem e classificação do OPF utilizando grafo completo, para uma melhor compreensão das propostas.

Tendo em vista que, todos os métodos propostos tomam como base Algoritmos Genéticos na tarefa de reduzir o conjuntos de dados, no Capítulo 3, será abordada a fundamentação teórica necessária para a compreensão desta meta-heurística. As subseções buscarão esclarecer as principais características, iniciando por uma breve introdução que trata sobre assuntos históricos que inspiraram essa técnica, bem como os operadores genéticos clássicos de seleção, cruzamento e mutação, que por sua vez, potencializam o processo de evolução dos indivíduos.

No Capítulo 4, serão apresentadas algumas abordagens de poda utilizando o classificador OPF. Inicialmente, serão mostradas as técnicas presentes no estado da arte, chamadas OPF-TPPA e ROB-OPF, ambas criadas pelo próprio autor do algoritmo OPF. Em seguida, serão exibidos os modelos propostos nesta dissertação, intitulados GAP-OPF, FSGAP-OPF e MOGAP-OPF. Além disso, detalhes de implementação para cada algoritmo serão esclarecidos por meio de pseudocódigos.

Posteriormente, no Capítulo 5, serão apresentadas a metodologia e os resultados obtidos para os modelos introduzidos na seção anterior. Detalhes de normalização, parâmetros do GA como taxas de reprodução, mutação e elitismo, bem como tamanho da população, número de gerações, bases de dados e métricas avaliadas para comparação, serão expostas na metodologia. Em seguida, os resultados serão expressos e comparados por meio de tabelas e gráficos, comparando o percentual de redução no conjunto de treinamento (poda), acurácia e tempo de classificação.

Finalmente, o Capítulo 6 descreverá as conclusões e contribuições deixadas por

esta dissertação, bem como recomendações de trabalhos futuros.

2 Floresta de Caminhos ótimos

Floresta de caminhos ótimos (*Optimum-Path Forest*, OPF) é um classificador utilizado para resolver problemas supervisionados e não supervisionados (PAPA et al., 2008b). No contexto dos problemas supervisionados, existem duas abordagens do OPF que visam resolver tais problemas e diferem-se basicamente quanto às suas relações de adjacência e a maneira a qual é definido o custo do caminho e os protótipos. A primeira abordagem usa um grafo k -NN para a relação de adjacência, em que os protótipos são computados pelo máximo de uma função de densidade probabilística construída a partir dos arcos. Nessa estratégia, almeja-se maximizar os custos, dados pelo menor valor de densidade probabilística do caminho (PAPA; FALCÃO, 2008). Na segunda abordagem, é utilizado um grafo completo como relação de adjacência e seus protótipos são encontrados em regiões com sobreposição de classes. Os protótipos, por sua vez, iniciam um processo de competição visando conquistar as demais amostras do conjunto de dados. Como resultado desse processo de competição, cada protótipo será raiz de uma árvore de caminhos ótimos, cada qual contendo as amostras mais fortemente conectadas à este protótipo. A coleção dessas árvores implica em uma floresta de caminhos ótimos, que por sua vez remete ao nome do classificador (PAPA; FALCAO; SUZUKI, 2009).

Na subseção a seguir, é apresentada a abordagem supervisionada do classificador de padrões OPF utilizando grafo completo. Essa versão foi escolhida para o desenvolvimento dos métodos propostos nesta dissertação.

2.1 OPF baseado em grafo completo

O algoritmo OPF com grafo completo modela o problema de classificação por computar uma floresta de caminhos ótimos no espaço de características induzido por um grafo (PAPA; FALCAO; SUZUKI, 2009; PAPA et al., 2012). Nesse grafo, cada vetor de características extraído de cada amostra é representado como sendo um nó do grafo, e os arcos são definidos de tal forma que todo vértice é adjacente a todos os outros vértices exceto ele próprio e ponderados por uma função de distância¹ entre seus nós correspondentes. A ideia principal é comandar um processo de competição iniciado pelas amostras que obedecem a ordem dos custos. Por sua vez, os nós protótipos que possuem custo zero iniciam esse processo para conquistar novas amostras ao propagar seus rótulos e atualizar os custos para os outros nós. Esse procedimento cria partições ótimas discretas, ou seja, árvores de caminhos ótimos enraizadas nos protótipos. Ao combinar essas árvores, é possível obter uma floresta de caminhos ótimos. Vale a pena ressaltar que diferentes

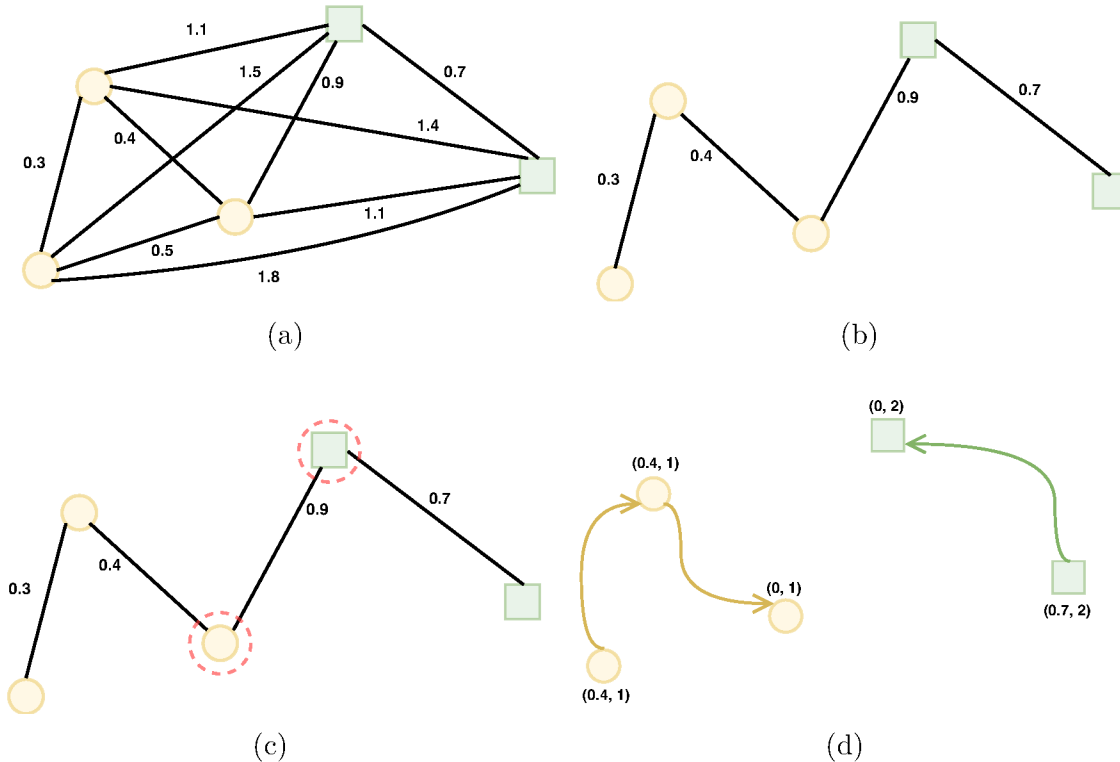
¹ Comumente se utiliza a distância euclidiana.

funções de custo de caminho levam a diferentes partições no grafo. A seguir, nas Seções 2.1.1 e 2.1.2, são apresentados os procedimentos de treinamento e teste para o OPF baseado em grafo completo, respectivamente.

2.1.1 Treinamento

Considere Z um conjunto de dados λ -rotulado e assuma Z_1 e Z_3^2 os conjuntos de treinamento e teste, respectivamente, sendo $|Z_1|$ e $|Z_3|$ a quantidade de amostras de cada partição, tal que $Z = Z_1 \cup Z_3$. Além disso, considere $s \in Z$ uma amostra n -dimensional e $d(s, v)$ uma função que calcula a distância entre dois padrões s e v , em que $v \in Z$.

Figura 1 – (a) Grafo ponderado completo gerado a partir de um determinado conjunto de treinamento. (b) MST construída com base nos padrões de treinamento. (c) Protótipos selecionados como sendo os elementos adjacentes de diferentes classes na MST (nós circutados). (d) Floresta de caminhos ótimos resultante da aplicação da função de custo de caminho $fmax$. A seta indica o nó predecessor no caminho ótimo. As entradas (x, y) ao lado dos nós representam, respectivamente, o custo e o rótulo das amostras.



Fonte: Próprio autor.

Seja $G^{tr} = (Z_1, A)$ um grafo completo (Figura 1a) derivado do conjunto de treinamento com seus arcos ponderados pela função de distância d . Um caminho π_s especifica uma sequência de nós distintos e adjacentes em G^{tr} começando no nó s . Além disso, um

² Posteriormente, na Seção 4, uma outra partição Z_2 será usada para compor o conjunto de validação.

custo $f(\pi_s)$ é atribuído a cada caminho π_s dado por uma função de conectividade f . De um modo geral, o OPF visa minimizar a seguinte função de custo de caminho:

$$\min f(\pi_s), \forall s \in Z_1. \quad (2.1)$$

A fase de treinamento consiste basicamente em encontrar um conjunto de protótipos $S \subseteq Z_1$ que serão utilizados para iniciar o processo de competição entre as amostras de treinamento e gerar uma coleção de árvores de caminhos ótimos (floresta de caminhos ótimos). Uma maneira de estimar os protótipos é escolhê-los a partir de regiões com classes sobrepostas, visto que são regiões muito suscetíveis a erros de classificação. De fato, os protótipos são os nós de classes diferentes que possuem a menor distância entre si (Figura 1c), que podem ser calculados por computar uma árvore geradora mínima (*Minimum Spanning Tree*, MST) (CORMEN et al., 1990) em G^{tr} (Figura 1b). Observe que cada classe pode ser representada por vários protótipos e deve haver pelo menos um protótipo por classe. Caminhos ótimos são calculados a partir dos protótipos para cada amostra de treinamento, de forma que cada protótipo se torne raiz de uma árvore de caminho ótimo composta por suas amostras mais fortemente conectadas (Figura 1d).

O algoritmo OPF é utilizado para calcular a floresta de caminhos ótimos, por meio da utilização de uma função de custo de caminho suave que possa agrupar amostras com propriedades semelhantes (FALCÃO; STOLFI; LOTUFO, 2004). Em todos os métodos empregados nesta dissertação, foi utilizada a função de custo de caminho $fmax$ (MIRANDA et al., 2008), que é calculada da seguinte forma:

$$fmax(\langle s \rangle) = \begin{cases} 0 & \text{se } s \in S, \\ +\infty & \text{caso contrário,} \end{cases} \quad (2.2)$$

$$fmax(\pi_s.\langle s, t \rangle) = \max\{fmax(\pi_s), d(s, t)\}.$$

A função $fmax(\langle \pi_s \rangle)$ calcula a distância máxima entre amostras adjacentes de π_s , quando π_s não é composto por uma única amostra (caminho trivial) representado por $\langle s \rangle$. A concatenação de um caminho π_s e um arco $\langle s, t \rangle$ é representado por $\pi_s.\langle s, t \rangle$.

Por aplicar a Equação 2.2 em todas as amostras $s \in Z_1$, obtém-se uma coleção de Árvores de Caminhos Ótimos (*Optimum Path Trees*, OPTs) conduzidas por S , o que resulta em uma Floresta de Caminhos Ótimos (ALLÈNE et al., 2007). Esse procedimento é descrito pelo Algoritmo 1.

Todos os nós do conjunto de treinamento, com exceção dos protótipos, são inicializados nas linhas 1 – 3, atribuindo para todos esses nós o custo infinito e nó predecessor nulo. Já para os nós protótipos, nas linhas 4 – 6, são designados definitivamente os valores de custo zero e nó predecessor nulo, bem como $L(s)$, que corresponde ao rótulo verdadeiro de cada amostra s , representado por $\lambda(s)$. Após atribuídos os parâmetros citados

Algoritmo 1 Algoritmo de Treinamento para o Classificador OPF com grafo completo

Entradas: Conjunto de treinamento Z_1 λ -rotulado, protótipos $S \subseteq Z_1$, $d(s, t)$ para designar a distância entre o par de amostras s e t , fila de prioridade Q e variável de custo cst .

Saídas: Floresta de caminhos ótimos P , mapa de custos C e mapa de rótulos L para o conjunto de treinamento Z_1 .

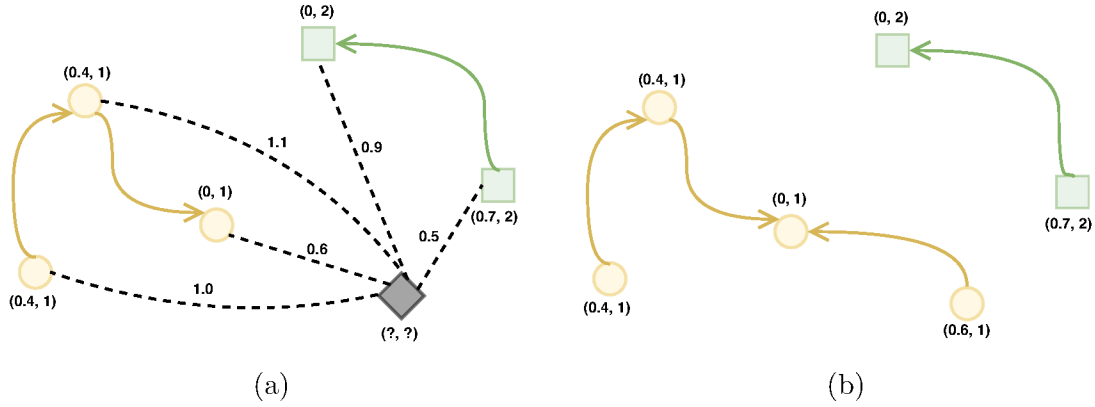
```

1: para todo  $s \in Z_1 \setminus S$  faça
2:   conjunto  $C(s) \leftarrow +\infty$  e  $P(s) \leftarrow nil$ 
3: fim para
4: para todo  $s \in S$  faça
5:   conjunto  $C(s) \leftarrow 0$ ,  $P(s) \leftarrow nil$ ,  $L(s) \leftarrow \lambda(s)$  e adicione  $s$  em  $Q$ 
6: fim para
7: enquanto  $Q$  não está vazio faça
8:   remova a partir de  $Q$  uma amostra  $s$  tal que  $C(s)$  é mínimo
9:   para todo  $t \in Z_1$  tal que  $t \neq s$  e  $C(t) > C(s)$  faça
10:    compute  $cst \leftarrow \max\{C(s), d(s, t)\}$ 
11:    se  $cst < C(t)$  então
12:      se  $C(t) \neq +\infty$  então
13:        remova  $t$  de  $Q$ 
14:      fim se
15:       $P(t) \leftarrow s$ ,  $L(t) \leftarrow L(s)$  e  $C(t) \leftarrow cst$ 
16:      adicione  $t$  em  $Q$ 
17:    fim se
18:  fim para
19: fim enquanto
20: devolve [  $P, L, C$  ]
```

anteriormente para cada protótipo, o mesmo é adicionado iterativamente em uma fila de prioridade de mínimo Q articulada pelo custo de cada amostra s (linha 5). O laço principal (linhas 7 – 19) compreende ao processo de competição do classificador OPF. Em cada iteração, o nó de menor custo s é removido da fila (linha 8), significando que esse nó já alcançou o menor custo que ele pode oferecer. No laço interno (linhas 9 – 18), todos os nós adjacentes a s são analisados no intuito de serem conquistados por s . A seguir, na linha 10, é computado o $fmax$ de acordo com a Equação 2.2. Ao final do laço principal, na linha 15, $P(t)$ (nó predecessor), $L(t)$ (mapa de rótulos) e $C(t)$ (mapa de custo) são atualizados, sendo que $P(t)$ e $L(t)$ são gerados com informações do nó s e o custo de t é obtido pela função $fmax$ conforme descrito na Equação 2.2. Na linha 9, percebe-se que $C(t) > C(s)$ é falso quando o nó t já estiver alcançado o seu custo mínimo possível, ou seja, quando t não estiver mais em Q . Por fim, na linha 20, o algoritmo retorna a floresta de caminhos ótimos P , os custos de caminhos ótimos C e os rótulos L para cada amostra $s \in Z_1$.

2.1.2 Classificação

Figura 2 – Exemplo do processo de classificação. O primeiro passo é (a) conectar (linhas tracejadas) uma amostra de teste (losango cinza) com todos os nós de treinamento. Depois disso, (b) t recebe o caminho ótimo do seu nó mais fortemente conectado (custo de classificação de 0,6), bem como o seu rótulo (círculo).



Fonte: Próprio autor.

Na fase de classificação, cada amostra $t \in Z_3$ é classificada individualmente por conectar-se a todos os nós da floresta de caminhos ótimos (Z_1) gerada no processo de treinamento (Figura 2a). Após isso, a amostra t receberá o rótulo do nó de treinamento v^* que satisfaz a Equação 2.3:

$$C(t) = \min\{\max\{C(v), d(v, t)\}\}, \forall v \in Z_1. \quad (2.3)$$

Em outras palavras, o processo de classificação visa encontrar o nó v do conjunto de treinamento que minimiza $C(t)$, como pode ser visto na Figura 2b.

Vale ressaltar que, de acordo com o exemplo mostrado na Figura 2a, embora t esteja posicionado mais próximo a um nó pertencente à classe quadrado, ele acabou sendo classificado por um nó da classe círculo. Isso enfatiza que o OPF não é um classificador baseado apenas em distâncias, assim como no k -NN por exemplo, mas, na verdade, um classificador que usa a influência da conectividade entre as amostras para executar seu processo de classificação. Um caso particular do OPF baseado em grafo completo é quando o custo de todos os nós $C(v)$ do conjunto de treinamento for menor que todas as distâncias $d(v, t_i)$ para um determinado nó $t_i \in Z_3$ (Equação 2.3). Nessas condições, a amostra t_i seria conquistada por um padrão v^* de Z_1 que oferecesse a menor distância, funcionando assim como o classificador 1-NN.

O Algoritmo 2 descreve o procedimento de classificação do OPF. O laço principal (linhas 1 – 13) desempenha a função de classificar cada mostra de teste $t \in Z_3$. Considere $v_i \in Z_1$, $i = 0, 1, \dots, |Z_1| - 1$ cada padrão do conjunto de treinamento ordenado

pelos custos. O laço interno (linhas 5 – 12) busca encontrar o elemento v_i que tiver o caminho ótimo $\pi_{v_i} \langle v_i, t \rangle$. Inicialmente, na linha 4, o nó t é classificado pela amostra de treinamento v_i que possui o menor custo. Caso o valor da variável *mincost*, atualizado na linha 3, seja menor que o custo do próximo nó v_{i+1} , então significa que o nó de teste já foi classificado pelo nó de treinamento de melhor custo de caminho. Caso contrário, existe um outro nó de treinamento apto a oferecer um custo de caminho melhor. A busca por esse novo nó é realizada no laço interno nas linhas 5 – 12, em que na linha 6 o valor do custo de caminho do nó v_{i+1} é computado. Se esse valor for menor que *mincost* (linha 7), então um melhor valor de custo de caminho foi encontrado e a amostra t é classificada pelo nó v_{i+1} (linha 9), bem como o valor de custo do caminho é atualizado (linha 8). Por fim, o algoritmo retorna a Floresta de Caminhos Ótimos, mapa de custos e rótulos (linha 14).

Algoritmo 2 Algoritmo de classificação para o classificador OPF com grafo completo

Entradas: Classificador OPF treinado [P, L, C] e conjunto de teste Z_3 .

Saídas: Floresta de caminho ótimo P, mapa de custos C e mapa de rótulos L para o conjunto de teste Z_3 .

```

1: para todo  $t \in Z_3$  faça
2:    $i \leftarrow 0$ 
3:    $mincost \leftarrow \max\{C(v_i), d(v_i, t)\}$ 
4:    $L(t) \leftarrow L(v_i), P(t) \leftarrow v_i, C(t) \leftarrow mincost$ 
5:   enquanto  $i < |Z_1|$  e  $mincost > C(v_{i+1})$  faça
6:      $cst \leftarrow \max\{C(v_{i+1}), d(v_{i+1}, t)\}$ 
7:     se  $cst < mincost$  então
8:        $mincost \leftarrow cst$ 
9:        $L(t) \leftarrow L(v_{i+1}), P(t) \leftarrow v_{i+1}, C(t) \leftarrow mincost$ 
10:    fim se
11:     $i \leftarrow i + 1$ 
12:  fim enquanto
13: fim para
14: devolve [ P, L, C ]
```

2.1.3 Análise de complexidade

Como visto nas seções anteriores, o classificador OPF é dividido basicamente em duas etapas: treinamento e classificação. No estágio de treinamento, inicialmente é feito um pré-rotulamento nos dados com o objetivo de identificar os protótipos, para em seguida executar o algoritmo de treinamento do OPF. Considere $|V|$ como sendo o número de vértices e $|A|$ a quantidade de arestas no conjunto de treinamento para representar o grafo completo $G^{tr} = (Z_1, A)$. A complexidade necessária para computar a MST utilizando o Algoritmo de Prim (CORMAN et al., 1990) é $O(|A| \log |V|)$, enquanto que para identificar e rotular os protótipos a complexidade é $O(|V|)$.

No Algoritmo 1, os laços das linhas 7 e 9 executam em tempo $O(|V|)$, cada. Portanto, a complexidade final do Algoritmo 1 é $O(|V|^2)$. Mais precisamente, o esforço

total demandado pelo processo de treinamento do OPF é dado por $O(|A| \log |V|) + O(|V|) + O(|V|^2)$, que pode ser expresso pela função dominante $O(|V|^2)$.

Para a etapa de classificação, no Algoritmo 2, o laço principal na linha 1 é executado em tempo $O(|Z_2|)$, onde $|Z_2|$ representa a quantidade de amostras para classificação. No pior caso, o laço interno na linha 5 é executado durante $|V|$ iterações (verificar todas as amostras de treino), que resulta em uma complexidade dada por $O(|V|)$. Dessa forma, a complexidade final exigida pelo processo de classificação do OPF é $O(|Z_2||V|)$. Por fim, a complexidade total estimada para executar os procedimentos de treinamento e classificação no OPF é definida por $O(|V|^2) + O(|Z_2||V|)$.

2.2 Conclusão

Neste capítulo, foram descritos os conceitos básicos para a compreensão do classificador OPF utilizando grafo completo. Dentre suas principais vantagens comparado a técnicas clássicas do estado da arte como Redes Neurais Artificiais de Múltiplas Camadas (*Artificial Neural Networks using Multilayer Perceptron*, ANN-MLP) e Máquinas de Vetores Suporte (*Support Vector Machines*, SVM), destacam-se: livre de parâmetros, não faz suposições sobre a forma e/ou separabilidade das classes, lida com múltiplas classes sem modificações ou extensões em seu algoritmo e possui erro de classificação zero no conjunto de treinamento sem ocasionar *overfitting* no modelo.

3 Algoritmos Genéticos

Algoritmo genético (*Genetic Algorithm*, GA) é uma meta-heurística da Computação Evolutiva (CE) capaz de resolver problemas de otimização e busca. Seu princípio é inspirado na teoria da evolução, proposto por Darwin, onde aplicam-se os conceitos de herança, seleção de indivíduos, mutação e reprodução.

Neste Capítulo, é apresentado a fundamentação teórica sobre GAs necessária para o entendimento de sua utilização no contexto de poda aplicado ao classificador OPF. As propostas de poda baseadas em GAs são posteriormente apresentadas na seção 4.3.

3.1 Introdução

Por volta do século XIX, estudos realizados por Charles Darwin acerca da teoria da evolução evidenciaram peculiaridades fundamentais nos indivíduos, tais como número finito de seres em uma determinada população, propriedades e habilidades diferentes entre cada um deles, dentre outras. Além disso, notou-se que a descendência de novos indivíduos era gerada com propriedades similares, e aqueles ditos mais resistentes tendiam a propagar suas características para gerações futuras.

Em meados dos anos 60 e 70, foram realizados os primeiros experimentos computacionais inspirados na evolução natural, com o objetivo de resolver problemas matemáticos difíceis da época. As soluções candidatas eram geradas de forma aleatória e a medida que as gerações passavam, na qual era realizado o processo de recombinação genética, melhores soluções eram encontradas até que se chegasse a um ótimo ou subótimo. Rechenberg (1965) criou uma estratégia evolucionária para otimizar parâmetros em problemas contínuos, enquanto que em Fogel, Owens e Walsh (1966), o foco deles estava em desenvolver uma ferramenta de programação evolutiva a fim de melhorar as soluções por meio de mutações aleatórias e seleção de indivíduos. Mais adiante, Holland (1992) formalizaram e criaram o conceito de GA.

Embora as abordagens trabalhassem com nomenclaturas e notações diferentes, partiam de um mesmo princípio que envolve reprodução, aleatoriedade, competição e seleção de indivíduos. A medida que essas técnicas foram se consolidando, houve a real necessidade de agrupá-las de forma padronizada dentro de um contexto na computação (BACK, 1996). Desta forma, surge a CE¹, que representa todos os esforços desempenhados e inspirados na evolução biológica, podendo ser dividida em: Programação Evolutiva (FOGEL; OWENS; WALSH, 1966), Estratégias Evolutivas (RECHENBERG, 1973), GAs (HOLLAND, 1992)

¹ Computação Evolutiva.

e Programação Genética (KOZA; KOZA, 1992; KOZA, 1994).

GA têm se destacado com relação as demais subáreas da CE, pois apresenta soluções eficazes, oferece maior flexibilidade e facilidade de implementação (CARDOSO et al., 1998; SIVAPATHASEKARAN et al., 2010). Além do mais, essa meta-heurística possui características fundamentais de prospecção e exploração. Na prospecção, a ideia é que a melhor solução seja investigada baseado no conhecimento já adquirido, enquanto que na exploração, novas áreas no espaço de busca são exploradas com o objetivo de escapar de subótimos.

Segundo Mitchell (1998), GA nada mais é do que uma metáfora associada aos acontecimentos e fenômenos biológicos ocorridos em indivíduos, bem como suas terminologias e representações.

3.2 Algoritmos genéticos

GA é um ramo da CE que pertence a classe dos algoritmos estocásticos, uma vez que seu processo de modelagem e reconstrução de indivíduos inspiram-se em fenômenos da natureza como herança genética e seleção natural. Comparado a outros algoritmos aleatórios, GA possui uma peculiaridade diferenciada, visto que seu processo de recombinação genética para a geração de novas soluções é realizado de maneira direcionada, adaptativa e paralela. Desta forma, GA mostra-se uma potencial ferramenta pelo seu poder de explorar bem várias regiões pertencentes ao domínio de busca (ROZENBERG; BÄCK; KOK, 2012).

Diante disso, essa meta-heurística pode ser usada para resolver problemas de otimização com grandes espaços de busca contendo muitas soluções. Suas características permitem que os problemas sejam resolvidos sem a hipótese de linearidade, continuidade, diferenciabilidade ou convexidade da função objetivo (OLIVEIRA; BEZERRA; ROCHA NETO, 2015), possuindo, portanto, certas vantagens sobre muitos métodos clássicos de otimização e busca.

A etapa inicial consiste na geração aleatória de indivíduos para compor a população primária, em que cada indivíduo deve representar uma solução factível do problema. Após isso, é computada a aptidão de cada solução, no qual será dada uma nota que qualifica a sua codificação, chamada de *fitness*. Após finalizado este estágio, novas gerações são produzidas² até que algum critério de parada seja estabelecido.

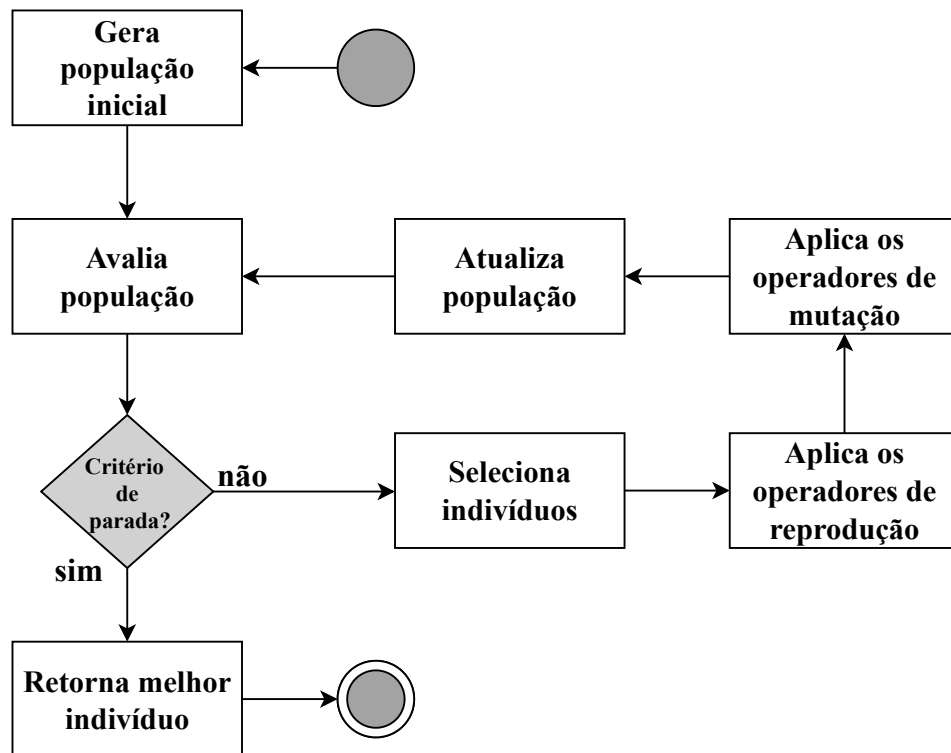
Os indivíduos que obtiverem melhores aptidões, ou seja, os mais resistentes, possuem maiores chances de serem selecionados para realizarem cruzamentos e mutações, de tal modo que suas características tendem a se perpetuarem por gerações futuras. Na reprodução, o objetivo é manter as características das soluções selecionadas por realizar recombinações

² Os indivíduos de gerações seguintes, são definidos baseados nos indivíduos de gerações anteriores.

genéticas, acontecendo geralmente de maneira sexuada entre pares de indivíduos. Já a mutação, consiste em modificar o material genético em alguns poucos indivíduos, com o propósito de gerar soluções em outros pontos no espaço de busca. Além disso, a fim de manter boas soluções já encontradas, comumente é aplicado o processo de elitismo em uma pequena parcela da população, que compreende em replicar os melhores indivíduos da geração corrente para a seguinte, sem nenhuma alteração em sua composição genética.

Ao término de cada geração, é calculado o *fitness* para cada indivíduo da população a fim de avaliar a qualidade das soluções. Em seguida, é verificado o critério de parada, geralmente concebido quando um número determinado de gerações é atingido ou quando há uma estagnação na melhoria de novas aptidões (SILVA, 2011). Quando o critério de parada for satisfeito, o GA retorna o indivíduo com a melhor aptidão³ da população atual. A Figura 3 ilustra todo o processo de otimização do GA, desde a criação dos primeiros indivíduos até a obtenção da melhor solução encontrada.

Figura 3 – Processo evolutivo do GA.



Fonte: Próprio autor.

Termos utilizados na evolução natural foram incorporados na abordagem artificial no intuito de apresentar uma analogia entre as terminologias, porém com semânticas diferentes. Na Tabela 1 (GOLBERG, 1989), é apresentado o significado do termo técnico

³ Pode acontecer do GA retornar mais de um indivíduo, isso vai depender do problema.

quando aplicado na teoria da evolução, bem como seu respectivo propósito quando abordado no contexto de GAs.

Tabela 1 – Significado dos termos em GAs.

Linguagem natural (termo)	Algoritmos genéticos (significado)
Cromossomo	Indivíduo
Gene	Característica
Alelo	Valor de uma característica
Locus	Posição do gene
Genótipo	Estrutura codificada
Fenótipo	Conjunto de parâmetros

Fonte: Próprio autor.

Um indivíduo pode ser constituído por um ou mais cromossomos, sendo cada indivíduo uma representação de solução candidata do problema (LINDEN, 2006). Essas soluções por sua vez, são decompostas por uma sequência de genes, representada por cada fator de característica na codificação do indivíduo (SIVANANDAM; DEEPA, 2007). Os possíveis valores que cada gene pode assumir, são chamados de alelo (MITCHELL, 1998), enquanto que a posição do gene no cromossomo é chamado de locus (LINDEN, 2006).

Outros dois termos relevantes em GAs são o genótipo e fenótipo. O genótipo se refere a estrutura como um todo do cromossomo, ou seja, as condições de satisfação para a codificação do cromossomo. Já o fenótipo, descreve a interação de seu material genético com o ambiente, gerado pelo conjunto de parâmetros do algoritmo (LINDEN, 2006).

3.3 Representação genética

A representação genética nada mais é do que uma forma de codificar a natureza do problema para que seja tratável ao computador, de modo que a informação possa ser processada. Todo esse procedimento está diretamente relacionado ao domínio de soluções no espaço de busca, no qual as restrições e soluções factíveis são direcionados por meio da função objetivo (SILVA, 2011).

A depender do problema, diferentes representações podem ser empregadas. Na sua forma mais simples, o indivíduo é representado por valores binários (HOLLAND, 1992). Essa interpretação tem sido bem aceita nas primeiras pesquisas que envolveu GA e até os dias atuais pela suma simplicidade.

Por outro lado, existem problemas que uma codificação simplista não é suficiente, devido as restrições naturais impostas pelo problema (TANG et al., 1996), que necessitam de um domínio maior para melhor representar todo o espaço de busca. Em vista disso, outras codificações foram inseridas, tais como a representação real aplicada a problemas contínuos, em que deseja-se uma precisão mais apurada (JANIKOW; MICHALEWICZ,

1991), codificação por permutação aplicada em otimização combinatorial (DAVIS, 1991), dentre outras. A Figura 4 apresenta algumas representações em GAs.

Figura 4 – Exemplo de representações em GAs.

Indivíduos	Codificação									
<table><tr><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td></tr></table>	1	1	0	0	0	1	1	0	1	Binária
1	1	0	0	0	1	1	0	1		
<table><tr><td>2</td><td>6</td><td>0</td><td>4</td><td>3</td><td>1</td><td>9</td><td>0</td><td>1</td></tr></table>	2	6	0	4	3	1	9	0	1	Combinatória
2	6	0	4	3	1	9	0	1		
<table><tr><td>1.021</td><td>0.993</td><td>7.038</td><td>3.750</td></tr></table>	1.021	0.993	7.038	3.750	Real					
1.021	0.993	7.038	3.750							
<table><tr><td>f</td><td>a</td><td>e</td><td>h</td><td>g</td><td>c</td><td>d</td><td>f</td><td>b</td></tr></table>	f	a	e	h	g	c	d	f	b	Caracteres
f	a	e	h	g	c	d	f	b		
<table><tr><td>></td><td>=</td><td>!</td><td><</td></tr></table>	>	=	!	<	Comandos					
>	=	!	<							

Fonte: Próprio autor.

3.4 População

Nessa etapa, é essencial estabelecer a quantidade suficientemente adequada de indivíduos por geração, pois valores demasiadamente pequenos podem prejudicar o desempenho do GA, por limitar demais o espaço de busca, ao passo que valores altos pode tornar uma busca custosa computacionalmente e acarretar em um processamento exaustivo e ineficiente. Um valor usual utilizado é de 100 indivíduos, mas que pode ser alterado dependendo da capacidade computacional ou da complexidade do problema (SIVANANDAM; DEEPA, 2007).

A população inicial geralmente é construída de forma aleatória ou por algum procedimento heurístico capaz de gerar indivíduos representativos capazes de atender bem todas as regiões do espaço de busca, para que se possa haver diferentes características e diversidade entre os cromossomos (SIVANANDAM; DEEPA, 2007). Portanto, análogo à teoria da evolução, é essencial existir pluralidade nas características genéticas para que naturalmente ocorra o processo de evolução (SILVA, 2011). Caso contrário, uma investigação apenas local no espaço de busca, pode eliminar as possibilidades de se obter ótimos globais (RAHNAMAYAN; TIZHOOSH; SALAMA, 2007).

A população atual é guiada por decorrência da população anterior, onde são realizados os procedimentos de seleção e aplicação dos operadores genéticos nos indivíduos da população anterior, para que novos indivíduos geneticamente modificados sejam inseridos na geração seguinte.

3.5 Função de avaliação

Cada indivíduo, independente de geração, deve representar uma solução disponível ao problema. Cada solução geneticamente codificada, por sua vez, carrega uma nota que qualifica a sua capacidade de resolução da função objetivo. Essa métrica deve ser capaz de guiar o processo de evolução dos indivíduos, portanto é essencial modelar bem a função de *fitness* para que indivíduos melhores sejam naturalmente encontrados no decorrer das gerações. A depender da otimização, quanto maior (maximização) ou menor (minimização) é o valor de *fitness* do indivíduo, melhor é a eficácia da solução apresentada.

A função de avaliação deve ser projetada de modo a refletir bem todas as restrições e características impostas pelo problema. Portanto, a função objetivo deve ser tratada com bastante rigor, para que seja extraído o máximo de informação possível acerca do problema. Quanto mais conhecido é o problema a priori, maiores são as métricas disponíveis para se modelar com mais precisão a função de *fitness* (LINDEN, 2006).

3.6 Operadores genéticos

Esta seção tem por finalidade apresentar os principais métodos de seleção, reprodução e mutação utilizados em GAs, sendo muitas dessas técnicas empregados nas abordagens propostas no Capítulo 4.

3.6.1 Métodos de seleção

Os indivíduos antes de sofrerem mutação ou serem cruzados, precisam primeiramente passar por um processo de triagem, para que os cromossomos mais aptos, possuam maiores chances de serem selecionados. Deste modo, algumas técnicas foram concebidas com esse propósito.

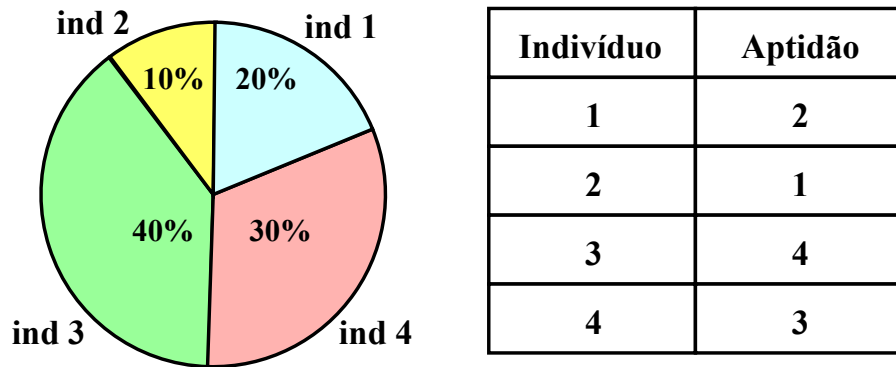
3.6.1.1 Seleção por roleta

Seu mecanismo baseia-se em computar a aptidão relativa de um determinado indivíduo com relação aos demais da população, para estimar sua fatia de influência na roleta. Quanto mais apto é o indivíduo, melhor é seu *fitness* e maior a sua chance de ser selecionado. A parcela probabilística do i -ésimo indivíduo ser escolhido é dada por:

$$p_i = \frac{i_i}{\sum_{j=1}^n i_j} \quad (3.1)$$

onde i_i é o *fitness* absoluto do i -ésimo indivíduo e n representa o tamanho da população. A seleção de k soluções é dada pelo giro independente da roleta k vezes. A Figura 5 ilustra um exemplo simples com 4 indivíduos para um problema de maximização.

Figura 5 – Método da roleta para uma população com 4 indivíduos.



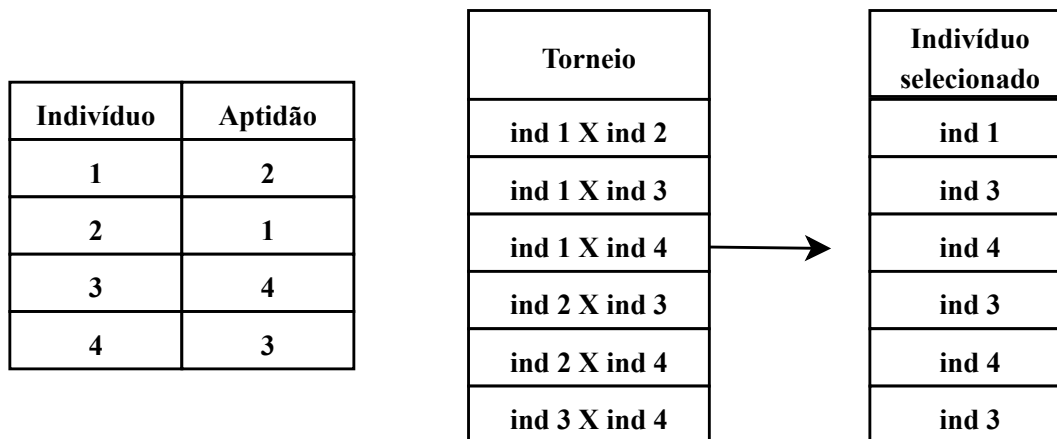
Fonte: Próprio autor.

Um ponto relevante nesse método, está no fato de ele permitir que todos os indivíduos da população possuam chances de serem selecionados. No entanto, uma população que possui indivíduos muito melhores que outros, pode impedir o progresso desses indivíduos menos resistentes de se propagarem. Por outro lado, uma geração que possua déficit de diversidade, ou seja, as características dos cromossomos são bem parecidas, dificilmente conseguirá avançar para regiões melhores (RAZALI; GERAGHTY et al., 2011).

3.6.1.2 Seleção por torneio

Consiste na eleição do indivíduo que possui o melhor *fitness* dentre n escolhidos aleatoriamente, podendo haver repetições entre esses indivíduos selecionados para ranqueamento. Esse processo deve ser executado k vezes para a seleção de k indivíduos. No geral, o valor de n é considerado 2, no entanto, existem outras variações que utilizam grupos maiores para a eleição do melhor indivíduo no torneio (THIERENS; GOLDBERG, 1994). Uma característica peculiar desse método comparado ao da roleta, é o fato de ele impossibilitar a seleção dos cromossomos que possuem as menores aptidões na população, uma vez que no melhor caso, pelo menos o pior indivíduo de todos nunca é selecionado, visto que dentre dois indivíduos no mínimo para o agrupamento, o pior de todos nunca é elegido. Observe na Figura 6, onde se tem todas as 6 possíveis combinações tomados 2 a 2 de indivíduos, que o indivíduo 2 nunca é selecionado quando sorteado para torneio, pois sua aptidão sempre é a pior de todas, considerando os valores do exemplo anterior aplicado na técnica da roleta.

Figura 6 – Seleção por torneio considerando 4 indivíduos para agrupamento em pares.



Fonte: Próprio autor.

3.6.1.3 Seleção por elitismo

Dentre as técnicas citadas, pode-se considerar a mais simples, pois é realizado apenas a replicação percentual dos melhores indivíduos da geração corrente para a geração seguinte (MITCHELL, 1998). Esse processo é comumente utilizado para uma pequena parcela da população, algo em torno de 10% dos melhores indivíduos.

É importante destacar que esse procedimento garante que a população posterior, no pior caso, deve possuir como melhor indivíduo, ao menos o mesmo valor do melhor indivíduo da geração anterior.

3.6.2 Técnicas de reprodução

Chamado também de recombinação ou cruzamento, esse operador busca gerar descendentes a partir da reprodução sexuada proveniente de dois cromossomos progenitores (DRÉO et al., 2006). Seu princípio compreende na troca de informações genéticas entre pares de indivíduos selecionados aleatoriamente, no intuito de produzir soluções melhores (YANG, 2014).

Os cromossomos previamente selecionados por meio de algum dos operadores citados na seção 3.6.1, são submetidos ao processo de cruzamento. Além disso, um valor de taxa de cruzamento deve ser fixado para definir a quantidade de indivíduos que serão reproduzidos (DRÉO et al., 2006). Geralmente é utilizado um intervalo entre 70% a 100% do tamanho da geração corrente para reprodução (YANG, 2014).

Pode ser considerado o principal operador no processo do GA, pois além de possuir o maior peso na taxa, seu procedimento de busca não degrada as soluções já encontradas. Os resultados de novos indivíduos são gerados por meio de soluções bem próximas as já

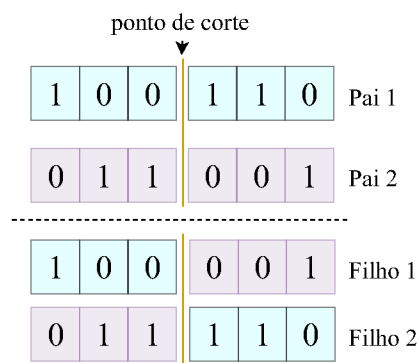
alcançadas, o que garante a permanência das características de adaptação das gerações anteriores (GOLBERG, 1989).

A seguir, os operadores de reprodução clássicos da literatura são apresentados. Primeiramente são mostrados os métodos de reprodução aplicados em indivíduos de codificação binária. Em seguida, as abordagens baseadas em representação inteira.

3.6.2.1 Cruzamento de único ponto

Inicialmente é escolhido aleatoriamente um ponto de corte definido no intervalo $[1, <\text{tamanho do indivíduo} - 1 >]$ para fragmentar em duas partes ambos os cromossomos que serão cruzados. Após essa etapa, o primeiro indivíduo gerado é resultado do agrupamento entre a primeira e segunda partição do primeiro e segundo cromossomo, respectivamente. O segundo cromossomo descendente é gerado de modo a complementar o passo anterior, ou seja, a segunda parte do primeiro indivíduo gerador se agrega com a primeira parte do segundo indivíduo progenitor (DRÉO et al., 2006). A Figura 7 ilustra seu funcionamento aplicado a dois indivíduos pais que produzem dois novos indivíduos filhos com ponto de corte na posição 3.

Figura 7 – Cruzamento uniponto.

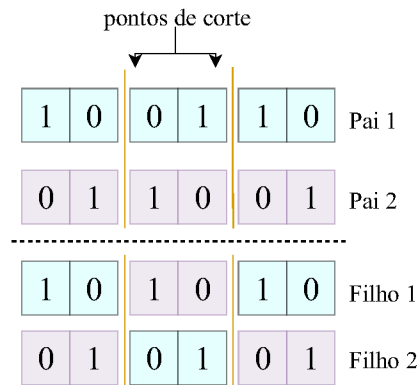


Fonte: Próprio autor.

3.6.2.2 Cruzamento por multi-pontos

Similar ao método anterior, porém devendo ser dividido em mais partições, delega-se aleatoriamente n pontos no intervalo entre $[1, <\text{tamanho do indivíduo} - 1 >]$ para definição dos pontos de corte. A construção de dois novos cromossomos descendentes se dar por meio da recombinação alternada das n partes entre os dois indivíduos pais (DRÉO et al., 2006; SILVA, 2011), como ilustrado na Figura 8 para um n igual a 2 com pontos de corte nas posições 2 e 4.

Figura 8 – Cruzamento multi-ponto.

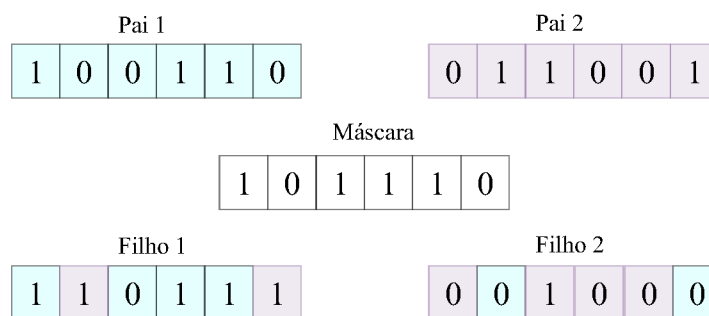


Fonte: Próprio autor.

3.6.2.3 Cruzamento uniforme

Fornece uma máscara de tamanho igual ao do indivíduo, para indicar em cada *bit*, qual o gene progenitor que será repassado aos filhos. Para cada posição da máscara, é atribuído aleatoriamente a responsabilidade ao pai 1 ou (exclusivo) pai 2. O valor da máscara deve representar o filho 1 de acordo com o valor de gene do pai respectivo, isto é, se o valor do *bit* na máscara é 1, o gene associado ao filho 1 deve vir do pai 1, caso contrário, deve ser reconstituído pelo pai 2; o filho 2 é construído pelo complemento dessa máscara (SILVA, 2011). A Figura 9 exemplifica melhor esse processo de reprodução.

Figura 9 – Cruzamento uniforme.



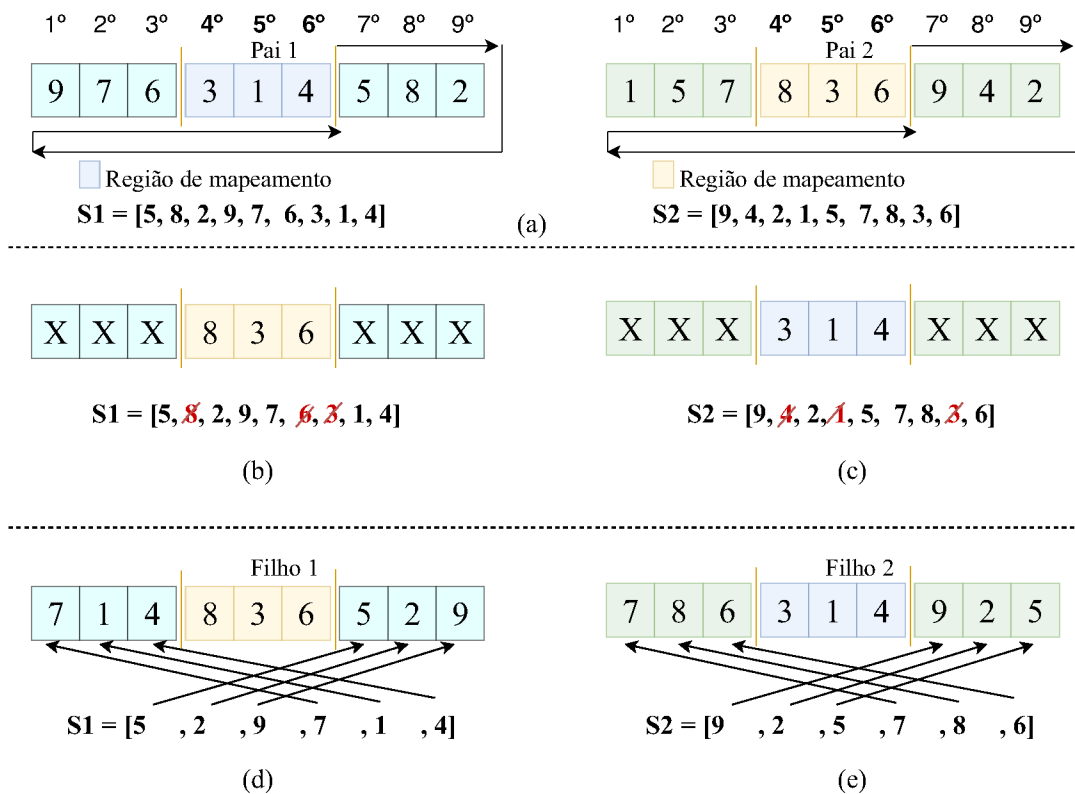
Fonte: Próprio autor.

3.6.2.4 Order Crossover

Conhecido também como operador OX, seu mecanismo se assemelha aos métodos citados anteriormente, tendo em vista que seu processo consiste na geração de duas novas soluções partindo de dois indivíduos pais.

É um operador de permutação que elege aleatoriamente regiões de mapeamento nos pais, para em seguida, definir sequências de valores ($s1$ e $s2$) que serão utilizados nos passos seguintes. No exemplo da Figura 10(a), os pontos de corte para mapeamento foram definidos nas posições 4, 5 e 6 do vetor, seguido pelas sequências geradas a partir dos pais por $s1$ e $s2$, possuindo origem na posição 7 do cromossomo e destino na posição 6, dando uma volta completa para que se possa percorrer todos os elementos do vetor. A codificação dos pais é repassada aos filhos sem nenhuma alteração em valor e posição referentes aos segmentos mapeados (Figuras 10(b) e 10(c)), onde esses valores já preenchidos (em vermelho) são ignorados para a etapa seguinte. A reconstituição dos demais genes faltosos é realizada partindo do primeiro ponto após o corte (posição 7 do vetor) até completar todos os espaços em branco, seguindo a ordem de alocação estabelecida pela sequência $s1 = [5, 8, 2, 9, 7, 6, 3, 1, 4]$ (Figura 10(d)) para o filho 1 e $s2 = [9, 4, 2, 1, 5, 7, 8, 3, 6]$ (Figura 10(e)) para o filho 2.

Figura 10 – Cruzamento *Order Crossover* para cromossomos de codificação inteira.



Fonte: Adaptado de Alves, Oliveira e Rocha Neto (2015).

Note que assim como nos pais, os valores não se repetem, logo a restrição se mantém para os filhos.

3.6.3 Estratégias de mutação

É um operador genético utilizado para introduzir novas soluções na geração seguinte assim como acontece na reprodução, porém esse processo geralmente ocorre de maneira assexuada, ou seja, a partir de um único cromossomo progenitor é gerado um novo descendente.

Este operador altera geneticamente alguns poucos *bits* de forma aleatória, no conjunto de cromossomos previamente selecionados. Tal modificação busca alcançar novas regiões no espaço, que até então não foram exploradas. Esse procedimento além de diversificar as soluções, pode evitar uma convergência prematura em regiões específicas no espaço de busca (ALBAYRAK; ALLAHVERDI, 2011).

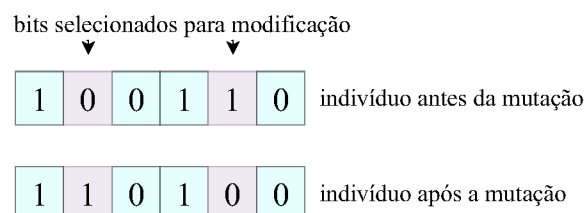
Usualmente utiliza-se uma taxa de mutação pequena em torno de 10% da população total, visto que seu processo aleatório pode produzir indivíduos piores ao invés de potencialmente encontrar uma nova região promissora. Uma taxa de mutação devidamente equilibrada garante a preservação na diversidade genética dos indivíduos, fator considerado de fundamental importância na evolução das espécies (DRÉO et al., 2006).

Os indivíduos mutados ocasionalmente podem produzir soluções infactíveis para o problema. No entanto, a depender das restrições, uma técnica ou outra pode se adequar melhor para atender essas regras. A seguir, os principais métodos de mutação são apresentados (GOLBERG, 1989).

3.6.3.1 Mutação *bit-flip*

Conhecido também como mutação por giro, é um operador clássico utilizado principalmente em indivíduos de codificação binária. A princípio, é selecionado de forma aleatória uma pequena taxa de genes em um determinado cromossomo para submetê-los ao processo de *flipping*, que consiste na modificação de seu valor. No caso de indivíduos binários, se o valor do gene for 1, substitui por 0, caso contrário, inverte para 1 (Figura 11).

Figura 11 – Mutação *bit-flip*.



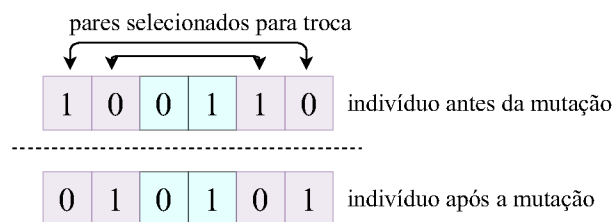
Fonte: Próprio autor.

Note que nesse operador, a quantidade de zeros ou uns pode mudar após finalizado seu processo, o que pode acabar violando alguma restrição de um determinado problema em específico.

3.6.3.2 Mutaç o swap

Pares de *bits* s o selecionados em pequenas propor  es de forma aleat ria para trocarem entre si suas posi  es e valores como ilustrado na Figura 12. Perceba que nesse operador, a quantidade de 0's e 1's de um determinado cromossomo n o se altera, tendo em vista que na pr tica,   realizado apenas um rearranjo nos genes. Portanto, pode-se considerar uma t cnica bastante usual quando deseja-se manter a mesma quantidade de 0's ou 1's, bem como para indiv duos de representa  o inteira, na obten  o de uma nova apar ncia nos indiv duos com os mesmos valores de antes.

Figura 12 – Muta  o *swap*.



Fonte: Pr prio autor.

3.7 Crit rios de parada

O problema de otimiza  o que qual o GA deve resolver   submetido a um crit rio de parada, que determina em qual momento o algoritmo de busca deve parar. O fluxo ideal de parada acontece quando a solu  o ou o conjunto de solu   es  timas do problema   encontrado (OLIVEIRA, 1999). Entretanto, na maiorias dos casos, quando se trata de problemas do mundo real, essa solu  o  tima, na maioria das vezes n o   conhecida ou o tempo necess rio para se chegar no resultado   invi vel. Portanto, a fim de contornar essa situa  o, s o apresentados algumas estrat gias de parada para melhor lidar com as incertezas dos dados:

- **Quantidade de gera  es:** o processo de busca   interrompido quando a quantidade de gera  es previamente estabelecida   alcan ada.
- **Tempo de execu  o:** baseado na natureza do problema, estima-se um tempo limite que acredita ser suficiente para encontrar a melhor solu  o. Caso esse tempo seja esgotado, o processo de otimiza  o   paralisado.

- **Aptidão do indivíduo:** acontece quando o melhor indivíduo da geração corrente apresenta uma solução factível considerada ótima, que é previamente definido pelo usuário.
- **Convergência na solução:** critério satisfeito quando ocorre a estagnação na evolução dos indivíduo após um determinado número de gerações. Pode ser calculado como segue na Equação 3.2.

$$|\mu_t - \mu_{t-1}| < \epsilon, \quad (3.2)$$

$$\mu_t = \frac{\sum_{i=1}^n a_{i,t}}{n}, \quad (3.3)$$

onde μ_t representa a média das aptidões de todos os indivíduos na geração t , n a quantidade de indivíduos e $a_{i,t}$ a aptidão do i -ésimo indivíduo da geração t . Se o módulo da diferença entre μ_t e μ_{t-1} (geração anterior a t) for menor que um determinado valor ϵ , acredita-se que a população vigente convergiu, logo a execução do algoritmo é imediatamente suspensa. Geralmente utiliza-se o valor 10^{-3} para ϵ .

Antes de iniciar o processo de otimização no GA, é necessário estabelecer quais medidas de parada vão ser habilitadas, para que todas sejam verificadas simultaneamente em tempo de execução. Dentre elas, a primeira que for satisfeita automaticamente encerra o processo de busca.

3.8 Otimização mono e multiobjetivo em GAs

Otimização pode ser entendido como um processo de busca que visa minimizar ou maximizar uma determinada função matemática, devendo satisfazer todas as restrições que foram modelas para o problema abordado (BRANKE et al., 2008). Esse aperfeiçoamento por investigar melhores soluções, pode ser desvendado de duas maneiras: mono-objetivo e multiobjetivo.

A otimização mono-objetivo concentra-se em melhorar uma única função objetivo, que normalmente resulta em apenas uma solução (BRANKE et al., 2008; FREITAS et al., 2010). No entanto, tratando de problemas do mundo real, a estratégia simplista oferecida por esse modelo mono-objetivo, pode se tornar incapaz de resolver alguns desses problemas, tendo em vista que, no geral, a função de otimização possui mais de um objetivo. Daí, surgem as estratégias multiobjetivo encarregadas para resolver problemas dessa natureza.

Na abordagem multiobjetivo, é levado em consideração várias funções objetivo no processo de otimização, como segue na Equação 3.4. Note que o domínio de busca

representado por x é compartilhado entre as funções objetivo $f_1(x), f_2(x), \dots, f_m(x)$, podendo resultar facilmente em conflitos uma sobre as outras, isto é, o aprimoramento de um determinado objetivo $f_i(x)$ implica no agravamento de um outro objetivo $f_j(x)$ e vice-versa, com $i \in [1, m]$ (CASTRO, 2001). Portanto, o objetivo principal dessa equação consiste em determinar o menor valor de retorno possível, quando somadas todas as funções ponderadas a partir de um determinado valor x .

$$\acute{o}timo = \min(w_1 * f_1(x) + w_2 * f_2(x) + \dots + w_m * f_m(x)) \quad (3.4)$$

$$\sum_{i=1}^m w_i = 1 \quad (3.5)$$

Cada função $f_i(x)$ representa um custo específico associado as características do problema, que nesse caso, quanto menor é a imagem da função, melhor é a sua contribuição para o resultado final. Os valores de $w_1, w_2, \dots, w_m$ simbolizam os pesos de cada função, que se comportam de maneira proporcional ao seu nível de relevância, ou seja, quanto maior é o valor de w_i , maior será a importância para a função $f_i(x)$.

3.9 Conclusão

Nesta seção foram discutidos os principais aspectos de GAs, desde sua invenção inspirada em fenômenos da teoria da evolução, como também, características e diferentes formas de representação, seleção, mutação e cruzamento de indivíduos.

Uma das principais vantagens presente nessa meta-heurística, é o fato dela explorar várias soluções paralelamente em diferentes regiões do espaço de busca, definido pelo número de indivíduos por população. Tal característica deve garantir a diversidade nas soluções, assim diminuindo as possibilidades do algoritmo de otimização convergir para ótimos locais.

Cada problema a ser resolvido possui sua peculiaridade com relação ao seu domínio de soluções, representação genética, função objetivo, dentre outras. Portanto, é fundamental uma boa adequação para o GA a fim de atender todas as restrições do problema, tendo em vista que, uma inadequada modelagem pode acarretar numa convergência prematura, soluções insatisfatórias ou até provocar um processamento exaustivo e ineficiente.

Para este trabalho, a configuração utilizada foi a binária, pois além de ser a alternativa mais simples, se adequou bem a modelagem das técnicas propostos.

4 Abordagens de poda no OPF

4.1 Introdução

Neste Capítulo são apresentadas algumas técnicas para poda do conjunto de treinamento do classificador OPF. Inicialmente, na Seção 4.2, são abordadas duas técnicas presentes no estado da arte que posteriormente serão usadas para comparação com os métodos desenvolvidos neste trabalho; presentes na Seção 4.3. Tais métodos desenvolvidos são baseados em GAs e constituem parte da proposta desta dissertação.

4.2 Métodos presentes no estado da arte

Nesta seção, são apresentados os métodos OPF-TPPA (PAPA; FALCÃO, 2009) e ROB-OPF (PAPA et al., 2010) utilizados para obter classificadores OPF podados. Ambos os métodos funcionam por eliminar amostras irrelevantes do conjunto de treinamento ao passo que mantêm o desempenho de classificação do OPF. Foram utilizadas essas duas técnicas pelo fato da primeira, OPF-TPPA, ter sido uma das pioneiras no quesito abordagens de poda para o classificador OPF. Por outro lado, ROB-OPF veio como uma melhoria do OPF-TPPA em termos de poda, bem como serve de base para o aperfeiçoamento de outras estratégias (NAKAMURA et al., 2011; PAPA et al., 2012; RODRIGUES; SOUZA; PAPA, 2017).

4.2.1 OPF-TPPA

O algoritmo *OPF Training Patterns Pruning Algorithm* (OPF-TPPA) usa um outro conjunto de validação Z_2 para o auxílio na identificação e eliminação de nós irrelevantes do conjunto de treinamento Z_1 . Papa e Falcão (2009) propuseram considerar como relevantes os nós pertencentes a Z_1 que participam no processo de classificação das amostras do conjunto de validação Z_2 ; quer seja diretamente, os nós de treinamento que satisfazem a Equação 2.3, ou indiretamente, os nós predecessores dos nós que diretamente classificaram as amostras de Z_2 . No final desse processo, os nós que não forem considerados como relevantes são eliminados de Z_1 , gerando assim um conjunto de treinamento mais compacto e representativo. Finalmente, o conjunto de treinamento podado é usado para projetar um classificador mais rápido e eficiente que pode ter seu desempenho de classificação avaliado no conjunto de teste Z_3 . O Algoritmo 3 descreve o passo a passo do processo de poda no método OPF-TPPA.

A linha 1 usa o Algoritmo 1 para treinar o modelo utilizando os padrões de

Algoritmo 3 Algoritmo OPF-TPPA

Entradas: Conjunto de treinamento Z_1 , validação Z_2 λ -rotulados e $d(s, t)$ para designar a distância entre o par de amostras s e t . Vetores FP e FN de tamanho igual ao número de classes c , para indicar os falsos positivos e falsos negativos, respectivamente, além da lista LM de nós mal classificados.

Saídas: Um novo conjunto de treinamento podado Z'_1 .

```

1: treine o OPF com o conjunto  $Z_1$ 
2: para todo  $t \in Z_2$  faça
3:   use o classificador obtido na linha 1 para classificar  $t$  com o rótulo  $L(t)$ 
4:   enquanto  $P(t) \neq R(t)$  faça
5:     marque  $P(t)$  como relevante
6:      $t \leftarrow P(t)$ 
7:   fim enquanto
8:   marque  $R(t)$  como relevante
9: fim para
10: remova as amostras não marcadas de  $Z_1$ 
11: devolve conjunto de treinamento podado  $Z'_1$ 

```

treinamento Z_1 . No *loop* principal (linhas 2 – 9), as amostras em Z_1 que participaram diretamente ou indiretamente no processo de classificação das amostras de Z_2 são marcados como relevantes. Na linha 3, a amostra $t \in Z_2$ recebe o rótulo de um determinado padrão $v^* \in Z_1$ que a conquista. No laço interno (linhas 4 – 7), é garantido que os nós predecessores $P(t)$ de t até chegarem na raiz $R(t)$ sejam também marcados como relevantes. Na linha 8, o último predecessor (a raiz) é marcado como relevante. No final deste processo, os nós que não foram classificados como relevantes são eliminados de Z_1 (linha 10), gerando assim um conjunto mais compacto e representativo (retorno na linha 11).

4.2.2 ROB-OPF

O *Robust Pruning of Training Patterns for OPF* (ROB-OPF) é uma melhoria do método OPF-TPPA (PAPA et al., 2010), capaz de obter taxas de poda mais altas nas amostras de treinamento do classificador OPF. Para isso, ROB-OPF é guiado pelo procedimento de três algoritmos: (i) Algoritmo 1 (treinamento), que atribui um caminho ótimo $P^*(t)$ de S para cada amostra de treinamento t em uma ordem não decrescente de custo mínimo, tal que o grafo seja particionado em uma floresta de caminhos ótimos; (ii) Algoritmo 4 (aprendizagem), que projeta um classificador usando o conjunto de treinamento Z_1 e avalia seu desempenho em Z_2 durante T iterações, a fim de selecionar a instância do classificador (Z_1) com maior taxa de acerto em Z_2 . Essencialmente, pressupõe-se que as amostras mais informativas contidas em Z_2 são as classificadas erroneamente. Diante disso, esses padrões mal classificados são substituídos por amostras em Z_1 que não são protótipos, escolhidas aleatoriamente. Por fim, (iii) Algoritmo 5 (aprendizado com poda), que é combinado com o Algoritmo 4 para identificar e eliminar as amostras irrelevantes do

conjunto Z_1 , com o propósito de reduzir seu tamanho e consequentemente construir um modelo mais rápido e robusto para o processo de classificação utilizando o conjunto Z_3 .

A ideia, basicamente, é: divida o conjunto de dados em Z_1 , Z_2 e Z_3 . Use Z_1 e Z_2 como entrada para o Algoritmo 4, o qual treina em Z_1 e classifica Z_2 no sentido de criar um conjunto Z_1 com amostras representativas (erros em Z_2). Em seguida, utilize esses novos conjuntos Z_1 e Z_2 para fazer a poda. Treine em Z_1 e classifique Z_2 . Quando classificar cada amostra em Z_2 , marque todo o caminho da amostra de Z_1 que conquistou a outra amostra de Z_2 . Após classificar todo o conjunto Z_2 , remova as amostras não marcadas em Z_1 , volte para o início, que seria executar o Algoritmo 4 novamente. Esse processo todo é repetido até um critério ser estabelecido.

Algoritmo 4 Algoritmo de aprendizagem

Entradas: Conjunto de treinamento Z_1 e validação Z_2 λ -rotulados, um número T de iterações e $d(s, t)$ para designar a distância entre o par de amostras s e t . Vetores FP e FN de tamanho igual ao número de classes c , para indicar os falsos positivos e falsos negativos, respectivamente, além da lista LM de nós mal classificados.

Saídas: Classificador OPF com a maior acurácia.

```

1:  $MaxAcc \leftarrow -1$ 
2: para todo iteração  $i = 1, 2, \dots, T$ , faça
3:    $LM \leftarrow \emptyset$ 
4:   projete o classificador OPF utilizando o Algoritmo 1
5:   para todo classe  $i = 1, 2, \dots, c$ , faça
6:      $FP(i) \leftarrow 0$  e  $FN(i) \leftarrow 0$ 
7:   fim para
8:   para todo padrão  $t \in Z_2$ , faça
9:     classifique  $t$  por  $L(t)$  usando a Equação 2.3
10:    se  $L(t) \neq \lambda(t)$  então
11:       $FP(L(t)) \leftarrow FP(L(t)) + 1$ 
12:       $FN(\lambda(t)) \leftarrow FN(\lambda(t)) + 1$ 
13:       $LM \leftarrow LM \cup t$ 
14:    fim se
15:  fim para
16:  compute  $acc$  usando a Equação 5.5
17:  se  $acc > MaxAcc$  então
18:    salve a instância atual do classificador e defina  $MaxAcc \leftarrow acc$ 
19:  fim se
20:  enquanto  $LM \neq \emptyset$  faça
21:     $LM \leftarrow LM \setminus t$ 
22:    Substitua  $t$  por uma amostra selecionada aleatoriamente em  $Z_1$ , que não seja
    protótipo
23:  fim enquanto
24: fim para

```

Na linha 4 do Algoritmo 4, o classificador OPF é treinado utilizando o conjunto Z_1 para a obtenção da floresta de caminhos ótimos P , mapa de custo C e mapa de rótulos L .

Após o modelo ser treinado, na i -ésima iteração (cada iteração do *loop* principal nas linhas 2 – 24), para cada amostra $t \in Z_2$ (*loop* interno nas linhas 8 – 15) os valores de falsos positivos (linha 11), falsos negativos (linha 12) e amostras mal classificadas (linha 13) são incrementados. Em seguida, é calculada a acurácia (linha 16) e a instância de melhor acurácia é salva na variável *MaxAcc* (linha 18). Por fim, as amostras mal classificadas $t \in Z_2$ que foram adicionadas na linha 13 são substituídas aleatoriamente por amostras não protótipos em Z_1 (linha 22), para que finalmente o modelo seja retreinado com um novo conjunto de treinamento Z'_1 ($|Z'_1| = |Z_1|$) e validação Z'_2 ($|Z'_2| = |Z_2|$), a fim de alcançar um melhor resultado na iteração seguinte.

Algoritmo 5 Algoritmo de aprendizagem com poda

Entradas: Conjunto de treinamento Z_1 e validação Z_2 λ -rotulados e $d(s, t)$ para designar a distância entre o par de amostras s e t . Os conjuntos de padrões relevantes R e irrelevantes I .

Saídas: Classificador OPF com o conjunto de treinamento reduzido.

```

1: enquanto  $I \neq \emptyset$  faça
2:   execute o Algoritmo 4 para obter o classificador OPF
3:    $R \leftarrow \emptyset$  e  $I \leftarrow \emptyset$ 
4:   para todo  $t \in Z_2$  faça
5:     classifique  $t$  por  $L(t)$  usando a Equação 2.3
6:     enquanto  $P(t) \neq nil$  faça
7:       insira  $P(t)$  em  $R$ 
8:        $t \leftarrow P(t)$ 
9:     fim enquanto
10:  fim para
11:   $I \leftarrow Z_1 \setminus R$ 
12:  mova amostras de  $Z_1 \setminus R$  para  $Z_2$ 
13: fim enquanto

```

O laço principal (linhas 1 – 13) no Algoritmo 5 almeja basicamente identificar as amostras irrelevantes de Z_1 e inserir no conjunto Z_2 . Na linha 2, é utilizado o Algoritmo 4 para obter a melhor instância em T iterações do classificador OPF. Na linha 3, os conjuntos de nós relevantes R e irrelevantes I são inicializados. Todos os nós $s \in Z_1$ que participam diretamente (nó que conquista t) ou indiretamente (nós predecessores até a raiz) da classificação para cada nó $t \in Z_3$ no laço interno (linhas 4 – 10) são considerados nós relevantes, e portanto adicionados no conjunto R (linha 7). Após classificadas todas as amostras t , o conjunto dos nós irrelevantes I é definido como $Z_1 \setminus R$ (linha 11). Por fim, os nós irrelevantes são movidos de Z_1 para Z_2 (linha 12), gerando um novo conjunto Z'_1 tal que $|Z'_1| < |Z_1|$. Todo esse procedimento é repetido até que não exista nós irrelevantes em Z_1 .

4.3 Técnicas baseadas em GAs

Meta-heurísticas como GAs são alternativas viáveis quando aplicadas para reduzir a complexidade computacional de muitos classificadores, como redes neurais (WHITLEY; BOGART, 1990), máquinas de vetores suporte por mínimos quadrados (SILVA; ROCHA NETO, 2015) e árvores de decisão (CHEN; WANG; ZHAI, 2009). Além disso, os recursos de GAs podem ser estendidos para uma ampla variedade de aplicações relacionadas ao aprendizado de máquina, como ajuste de parâmetros em classificadores (LAM et al., 2001) e seleção de atributos (HUANG; WANG, 2006). Devido às suas características subjacentes, GAs podem ser usados para resolver problemas de otimização sem a suposição de linearidade, continuidade, diferenciabilidade ou convexidade da função objetivo. Infelizmente, essas propriedades desejadas não são encontradas em vários métodos matemáticos clássicos encontrados na literatura (SRINIVAS; PATNAIK, 1994).

Nesse contexto, foram introduzidas nesta dissertação três propostas baseadas em GAs chamadas GAP-OPF, FSGAP-OPF e MOGAP-OPF, que aproveitam as características acima mencionadas para podar amostras irrelevantes ou similares no grafo de treinamento do OPF, que seriam usadas no estágio de classificação. Mais detalhes de cada proposta podem ser consultadas nas seções 4.3.1, 4.3.2 e 4.3.3.

4.3.1 GAP-OPF (Proposta 1)

O método proposto neste tópico, denominado Algoritmos Genéticos para OPF Podado (*Genetic Algorithms for Pruned OPF*, GAP-OPF), emerge da aplicação de um GA padrão ao classificador OPF com o objetivo de selecionar e remover amostras menos representativas do conjunto de treinamento, que seriam utilizadas em sua fase de classificação, enquanto mantendo ou mesmo melhorando a precisão do classificador (BARBOSA; COSTA; ROCHA NETO, 2018). Por utilizar uma versão padrão do GA, pode-se mostrar a capacidade do método proposto e permite que outras técnicas mais sofisticadas baseadas em outras versões de GAs ou em outras meta-heurísticas sejam também exploradas.

O procedimento envolve modelar um indivíduo que contém apenas um cromossomo, que é representado por um vetor de genes que podem assumir apenas os valores “1” (verdadeiro) ou “0” (falso), para indicar, respectivamente, a presença ou ausência de uma amostra no conjunto de treinamento Z_1 . Assim, a quantidade de padrões não podados do conjunto de treinamento Z_1 é designado pela quantidade de genes “1” do indivíduo. Depois disso, operadores genéticos são escolhidos para auxiliar o processo de reprodução, mutação e elitismo.

Também é necessário elaborar uma função de aptidão, que levará em consideração a taxa de erro de classificação de um determinado indivíduo. Esta função terá seu valor de retorno minimizado pelo GA. Quando o GA atingir seu critério de parada, o melhor

indivíduo encontrado será selecionado.

4.3.1.1 Indivíduos ou cromossomos

Considerando o método proposto, o indivíduo é constituído por apenas um cromossomo, que pode ser representado por $\mathbf{c} = [g_1 \ g_2 \ \dots \ g_i \ \dots \ g_{N-1} \ g_N]$, onde $g_i \in \{0, 1\}$ refere-se a i -ésima amostra de treinamento no vetor $\mathbf{c}$ com N igual a $|Z_1|$. Nessa codificação, os genes de $\mathbf{c}$ com o valores iguais a 1 representam as amostras que compõem o mais novo conjunto de treinamento, enquanto as os genes iguais a 0 representam as amostras podadas. Assim, o número total de padrões de treinamento de uma solução candidata pode ser computado como segue na Equação 4.1.

$$T_{\mathbf{c}} = \sum_{i=1}^N g_i. \quad (4.1)$$

Para entender como ocorre o processo de poda, lembre-se que cada indivíduo é gerado em um modo pseudo-aleatório com genes $g_i \in \{0, 1\}$. Isso implica que a chance de um indivíduo ter todos os genes iguais a 1 (sem poda) é bastante baixa, tendo em conta que na primeira geração esta probabilidade pode ser expressa como $\frac{1}{2^N}$. Para gerações consecutivas, essa chance pode aumentar se o GA avaliar que a aptidão do indivíduo tende a melhorar com o aumento das amostras que compõem seu material genético. No entanto, é pouco provável que isso aconteça, uma vez que nem todas as amostras são necessárias para diminuir a taxa de erro em um indivíduo. De fato, será mostrado na Seção 5.2.1.2, que com a diminuição das amostras do modelo OPF, é possível aumentar sua precisão.

4.3.1.2 Função de *fitness*

Dada uma certa solução candidata $\mathbf{c}$, a ideia principal da função de *fitness* é minimizar a taxa de erro de classificação para um determinado indivíduo. Portanto, foi definida a função de aptidão conforme a Equação 4.2.

$$fitness(\mathbf{c}) = error_rate(\mathbf{c}), \quad (4.2)$$

onde $error_rate(\mathbf{c})$ é calculado como

$$error_rate(\mathbf{c}) = 1 - acc, \quad (4.3)$$

e

$$acc = \frac{2c - \sum_{i=1}^c E(i)}{2c} = 1 - \frac{\sum_{i=1}^c E(i)}{2c}, \quad (4.4)$$

a acurácia computada sobre o conjunto de validação Z_2 , que utiliza o somatório do erro parcial E de cada classe c , e considera a influência de classes de tamanhos diferentes em Z_2 ¹. Mais detalhes da Equação 4.4 podem ser consultados na seção 5.1.

¹ Definição semelhante será aplicada para medir a acurácia em Z_3 , bem como para todos os métodos de comparação apresentados nesta proposta.

4.3.1.3 Operadores genéticos

Os indivíduos selecionados para reprodução e mutação foram obtidos por aplicador o método do torneio (Seção 3.6.1.2), que consiste basicamente na eleição da melhor solução, dentre um grupo de n indivíduos escolhidos aleatoriamente.

O operador de reprodução utilizado nesta abordagem foi o cruzamento de único ponto (Seção 3.6.2.1). Nesse operador, um ponto nos cromossomos de ambos os pais é escolhido aleatoriamente e designado como "ponto de cruzamento". Os genes à direita desse ponto são trocados entre os dois cromossomos pai. Este processo resulta em dois descendentes, cada um carregando alguma informação genética de ambos os pais.

No intuito de escapar de mínimos locais e possivelmente alcançar soluções ótimas globais, foi utilizado o operador *bit-flip* (Seção 3.6.3.1) para realizar o mecanismo de mutação nos indivíduos. Este operador trabalha selecionando um ou mais *bits* aleatórios e invertendo-os; ou seja, se o valor do *bit* é 1, inverte para 0, caso contrário (valor do *bit* igual a 0), troca para 1. O número de *bits* selecionados nesse operador foi o valor arredondado obtido pelo cálculo de 10% da quantidade de genes de um cromossomo.

4.3.1.4 Algoritmo GAP-OPF

Algoritmo 6 Algoritmo GAP-OPF

Entradas: Z_1 : conjunto de treino, Z_2 : conjunto de validação, t_{max} : número máximo de gerações, s : tamanho da população.

Saídas: Classificador OPF com o conjunto de treinamento reduzido i.e., $\mathbf{c}^1$.

- 1: Execute o Alg. 1 em Z_1 para obter o classificador OPF
 - 2: $t \leftarrow 0$
 - 3: $P(t) \leftarrow \text{geraPopulacaoInicial}(s)$
 - 4: Considere um conjunto $\{\mathbf{c}\}_{i=1}^s$, onde s é o número de indivíduos em $P(t)$, criado aleatoriamente
 - 5: **para todo** indivíduo $i \in P(t)$ **faça**
 - 6: $i.\text{fitness} \leftarrow \text{funcaoFitness}(\mathbf{c}^i, Z_2)$
 - 7: **fim para**
 - 8: **enquanto** $t \leq t_{max}$ **faça**
 - 9: $[\text{mutacaoSel}, \text{reproducaoSel}] \leftarrow P(t).\text{selecao}(\text{taxaMutacao}, \text{taxaReproducao})$
 - 10: $\text{individuosMutados} \leftarrow \text{mutacao}(\text{mutacaoSel})$
 - 11: $\text{individuosReproduzidos} \leftarrow \text{reproducao}(\text{reproducaoSel})$
 - 12: $\text{individuosElitizados} \leftarrow \text{elitismo}(P(t), \text{taxaElitismo})$
 - 13: $P(s)_{\text{novo}} \leftarrow \text{individuosMutados} \cup \text{individuosReproduzidos} \cup \text{individuosElitizados}$
 - 14: **para todo** indivíduo $j \in P(t)_{\text{novo}}$ **faça**
 - 15: $j.\text{fitness} \leftarrow \text{funcaoFitness}(\mathbf{c}^j, Z_2)$
 - 16: **fim para**
 - 17: $P(t) \leftarrow P(t)_{\text{novo}}$
 - 18: $t \leftarrow t + 1$
 - 19: **fim enquanto**
 - 20: **devolve** $P(t).\text{melhorSolucao}()$
-

Observe que o Algoritmo 6 possui o fluxo convencional de uma GA. Inicialmente, é definido o modelo OPF treinado (linha 1), bem como a criação da população inicial (linha 3). No primeiro laço (linhas 5 – 7), é designado o valor de *fitness* para cada indivíduo da população inicial. Das linhas 8 a 19, é realizado todo o processo de otimização, onde os indivíduos selecionados para mutação e reprodução (linha 9) são alterados (linhas 10 e 11). A fim de manter os indivíduos bons para gerações seguintes, na linha 12, é realizado o processo de elitismo. Na linha 13, uma nova população de indivíduos é formada com os novos indivíduos gerados na etapa anterior, para em seguida serem mapeados com seus respectivos valores de aptidões (linhas 14 – 16). Após finalizado o número de gerações t_{max} , o algoritmo devolve o indivíduo de melhor aptidão (linha 20).

4.3.2 FSGAP-OPF (Proposta 2)

O método proposto nesta seção, denominado Algoritmo Genético de Tamanho Fixo para o OPF Podado (*Fixed-Size Genetic Algorithm for Pruned OPF*, FSGAP-OPF), está embasada em GAs para o classificador OPF com o propósito de manter apenas uma quantidade fixa das amostras mais significativas do conjunto de treinamento para posteriormente serem utilizadas no estágio de teste, construindo assim um conjunto reduzido no treino para maior rapidez no processo de classificação, além disso, busca-se também manter ou até mesmo melhorar a eficiência do classificador (COSTA; BARBOSA; ROCHA NETO, 2019). Para tanto, similar ao método anterior (Seção 4.3.1), foi modelado um indivíduo composto por um único cromossomo, que é representado por um vetor de genes que podem assumir apenas os valores "1" (verdadeiro) ou "0" (falso), para designar, respectivamente, a presença ou ausência de uma amostra no conjunto de treinamento Z_1 . Desta forma, a poda é guiada pela quantidade de genes "0" nos indivíduos.

Tal como no GAP-OPF (Seção 4.3.1), a função de *fitness* é guiada pela taxa de erro de classificação de um determinado indivíduo. Portanto, essa função de otimização mono-objetivo terá seu valor de retorno minimizado pelo GA. Quando o GA atingir seu critério de parada, o melhor indivíduo encontrado é selecionado.

Para obter um conjunto de treinamento de tamanho fixo, o operador de reprodução foi projetado de tal modo que o número de *bits* configurado para "1" seja mantido inalterado. Desta forma, é possível determinar a taxa de poda que se deseja alcançar antes que ocorra o processo de remoção.

4.3.2.1 Indivíduos ou cromossomos

Similar a representação abordada no método anterior (Seção 4.3.1), o indivíduo segue possuindo um único cromossomo exemplificado como $\mathbf{c} = [g_1 \ g_2 \ \dots \ g_i \ \dots \ g_{N-1} \ g_N]$ com $g_i \in \{0, 1\}$, onde i indica a i -ésima posição no cromossomo $\mathbf{c}$ com tamanho N igual

ao conjunto de treinamento ($|Z_1|$). Por exemplo, se o cromossomo está codificado como $\mathbf{c} = [1 \ 1 \ 0 \ 1 \ 0 \ 1]$, significa que as amostras nas posições 3 e 5 foram podadas.

4.3.2.2 Função de *fitness* e método de seleção

O objetivo principal é otimizar a função mono-objetivo (Equação 4.5) que visa minimizar a taxa de erro na classificação, para um certo indivíduo $\mathbf{c}$ que represente uma possível solução ao problema.

$$fitness(\mathbf{c}) = error_rate(\mathbf{c}), \quad (4.5)$$

com

$$error_rate(\mathbf{c}) = 1 - acc. \quad (4.6)$$

O valor de acurácia obtido sobre o conjunto Z_2 é calculado com base na Equação 4.4. Mais detalhes dessa equação² podem ser consultados na seção 5.1.

Note que a fórmula como é calculada a aptidão dos indivíduos é idêntica ao método anterior (GAP-OPF). A principal diferença está na quantidade 1's fixada pelo método para cada indivíduo $\mathbf{c}$. Mais detalhes de como se constrói indivíduos com quantidades fixas de *bits* podem ser vistos na Seção seguinte.

A técnica utilizada para selecionar os indivíduos nesta abordagem foi o torneio (Seção 3.6.1.2), assim como na proposta anterior.

4.3.2.3 Reprodução

Para manter a taxa de poda constante, ou seja, com um número fixo de amostras no conjunto de treinamento, as restrições do problema foram codificadas de modo a manter o número de *bits* configurado para "1" inalterado em cada indivíduo no processo de reprodução. Essa propriedade não seria satisfeita caso fosse permitido uma troca direta entre indivíduos, uma vez que indivíduos inválidos seriam produzidos. Portanto, foi proposta uma codificação baseada na transformação de indivíduos binários para inteiros, ao passo que torne-se possível aplicar operadores de permutação³.

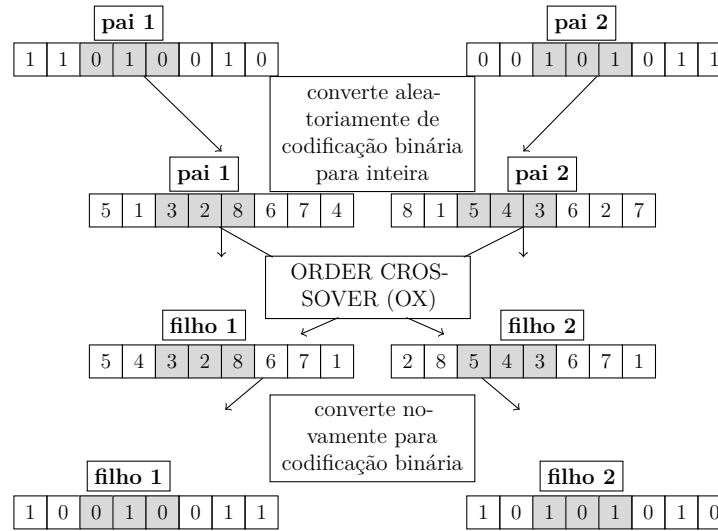
A Figura 13 descreve a maneira como a transformação entre as codificações binária e inteira ocorre e, em seguida, como o operador *Order Crossover* (OX)⁴ funciona sobre os pais de codificação inteira. Após esse processo, cada filho é retornado à codificação binária, de modo que cada uma dessas representações mantém a restrição em manter a quantidade de 1's inalterada.

² Esta mesma equação foi utilizada para calcular a acurácia no conjunto de teste Z_3 .

³ Tais como *Crossover* Parcialmente Mapeado (*Partially Mapped Crossover*, PMX), *Crossover* Circular (*Cycle Crossover*, CX) e *Order Crossover* (OX) (EIBEN; SMITH et al., 2003).

⁴ O operador OX foi usado nesta abordagem.

Figura 13 – Processo de transformação entre codificações binária-inteira para aplicação do operador *Order Crossover* (Seção 3.6.2.4) seguido pela transformação de codificação inteira-binária.



Fonte: Adaptado de Silva e Rocha Neto (2015).

4.3.2.4 Mutação

A fim de explorar outras regiões no espaço de busca com o objetivo de escapar de subótimos locais e alcançar mínimos globais, foi utilizado o operador *bit-flip* (Seção 3.6.3.1) para executar o processo de mutação. Seu mecanismo consiste em selecionar aleatoriamente um ou mais *bits* para a inversão de seus valores. Mais especificamente, são coletados os genes com valores 0's e 1's de um indivíduo para a criação de dois novos subconjuntos contendo os mesmos valores de *bits* (um subconjunto com apenas 1's e outro com apenas 0's), e assim feito o mapeamento de suas posições iniciais (índice) com relação ao indivíduo original. Em seguida, pares de *bits* (cerca 10% do total de genes) aleatoriamente são escolhidos de cada novo subconjunto para inverter seus valores nas posições correspondentes do indivíduo original. Esse procedimento garante que o número de 0's e 1's seja mantido inalterado após toda essa operação ser concluída em cada indivíduo selecionado para mutação.

4.3.2.5 Algoritmo FSGAP-OPF

Similar ao Algoritmo empregado em 4.3.1.4, o Algoritmo 7 segue o mesmo fluxo convencional de uma GA. No entanto, as principais diferenças concentram-se na não violação do número fixo de *bits* 1, sendo elas presentes na invocação do algoritmo que recebe como entrada o parâmetro α para fixar previamente a taxa de poda, na forma como são criados os indivíduos da população inicial (linha 3), na implementação do operador de mutação (linha 10) e no modo como os indivíduos são reproduzidos (linha 11).

Algoritmo 7 Algoritmo FSGAP-OPF

Entradas: Z_1 : conjunto de treino, Z_2 : conjunto de validação, t_{max} : o número máximo de gerações, s : tamanho da população, α : percentual de poda.

Saídas: Classificador OPF com o conjunto de treinamento reduzido i.e., $\mathbf{c}^1$.

```

1: Execute o Alg. 1 em  $Z_1$  para obter o classificador OPF
2:  $t \leftarrow 0$ 
3:  $P(t) \leftarrow geraPopulacaoInicial(s, \alpha)$ 
4: Considere um conjunto  $\{\mathbf{c}\}_{i=1}^s$ , onde  $s$  é o número de indivíduos em  $P(t)$ , criado aleatoriamente, porém, atendendo a restrição de manter a quantidade fixa de 0's e 1's
5: para todo indivíduo  $i \in P(t)$  faça
6:    $i.fitness \leftarrow funcaoFitness(\mathbf{c}^i, Z_2)$ 
7: fim para
8: enquanto  $t \leq t_{max}$  ou outro critério de parada não é satisfeito faça
9:    $[ mutacaoSel, reproducaoSel ] \leftarrow P(t).selecao(taxaMutacao, taxaReproducao)$ 
10:   $individuosMutados \leftarrow mutacao(mutacaoSel)$ 
11:   $individuosReproduzidos \leftarrow reproducao(reproducaoSel)$ 
12:   $individuosElitizados \leftarrow elitismo(P(t), taxaElitismo)$ 
13:   $P(s)_{novo} \leftarrow individuosMutados \cup individuosReproduzidos \cup individuosElitizados$ 
14:  para todo individual  $j \in P(t)_{novo}$  faça
15:     $j.fitness \leftarrow funcaoFitness(\mathbf{c}^j, Z_2)$ 
16:  fim para
17:   $P(t) \leftarrow P(t)_{novo}$ 
18:   $t \leftarrow t + 1$ 
19: fim enquanto
20: devolve  $P(t).melhorSolucao()$ 

```

4.3.3 MOGAP-OPF (Proposta 3)

Algoritmo Genético Multiobjetivo para o OPF Podado (*Multi-Objective Genetic Algorithm for Pruned OPF*, MOGAP-OPF) é baseado em GAs multiobjetivo que visa simultaneamente minimizar a taxa de erro do classificador OPF, como também podar as OPTs que seriam utilizados em sua fase de classificação. Como resultado, pode-se obter um conjunto de treinamento menor e mais representativo, visando manter ou até mesmo melhorar o desempenho do classificador.

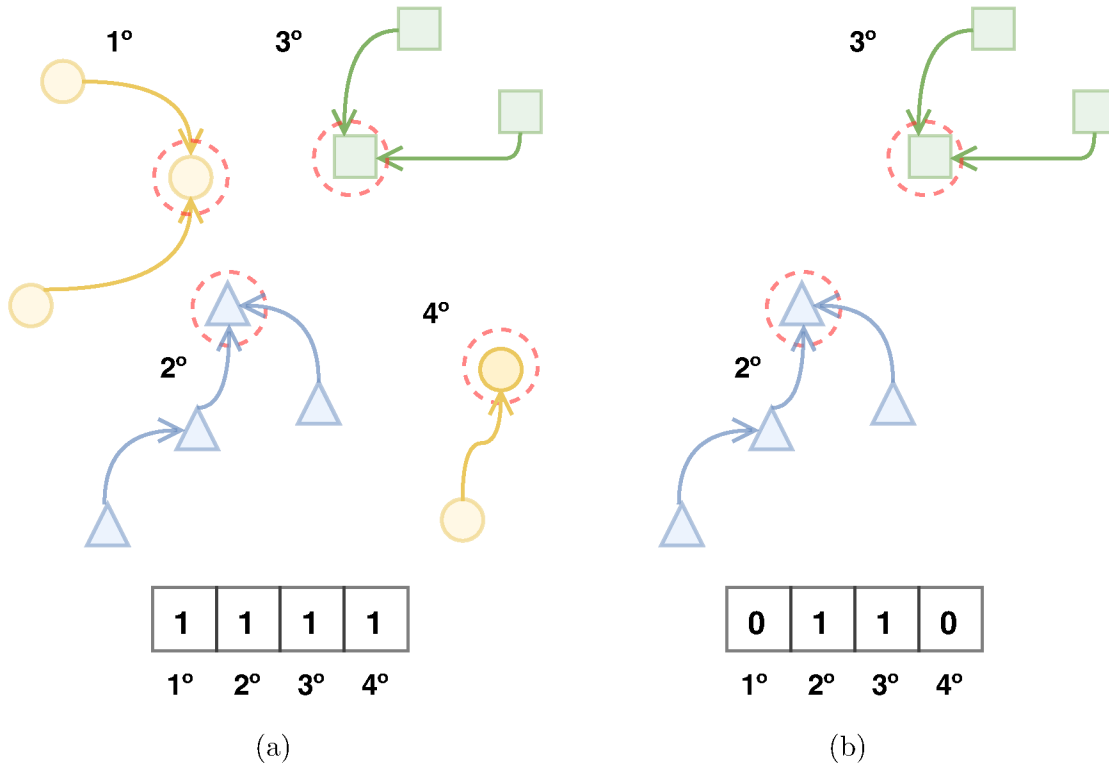
Da mesma forma que os anteriores, o indivíduo é composto de apenas um cromossomo e é representado por um vetor binário de genes, onde cada *bit* é definido como "1" (verdadeiro) se uma determinada OPT for usada no processo de classificação ou "0" (false) caso contrário.

4.3.3.1 Representação genética

O indivíduo codificado nessa proposta pode ser representado como $\mathbf{c} = [g_1 \ g_2 \ \dots \ g_i \ \dots \ g_{N-1} \ g_N]$, no qual $g_i \in \{0, 1\}$ denota a i -ésima OPT no vetor $\mathbf{c}$ e N igual ao número de OPTs geradas pelo algoritmo OPF, que por sua vez é igual ao número de protótipos,

visto que cada protótipo configura a raiz de uma OPT. Nessa representação, os genes de $\mathbf{c}$ que possuem valores iguais a 1 equivalem as OPTs da floresta de caminho ótimo que seriam usadas no processo de classificação, enquanto os genes iguais a 0 indicam as OPTs podadas. Por exemplo, se o cromossomo é codificado como $\mathbf{c} = [1 \ 0 \ 1 \ 0 \ 1]$, são podadas duas OPTs. Portanto, o tamanho total do novo conjunto de treinamento Z'_1 de um indivíduo candidato é equivalente a soma de todos os nós das OPTs não podadas. A ideia por trás dessa modelagem de cada indivíduo é apresentada na Figura 14.

Figura 14 – Exemplo de representação genética para o MOGAP-OPF considerando um indivíduo com 4 genes, ou seja, 4 OPTs. Inicialmente, é apresentado (a) o modelo sem poda e, em seguida, (b) o indivíduo com metade das OPTs podadas pelo processo de otimização do AG.



Fonte: Próprio autor.

4.3.3.2 Função de *fitness*

Considerando um indivíduo específico (solução possível) $\mathbf{c}$, é proposta uma função de aptidão multiobjetivo que visa tanto maximizar a acurácia quanto a taxa de poda. Nesse contexto, almeja-se alta precisão e poucas OPTs. Em outras palavras, também pode-se afirmar que a função de *fitness* é baseada na minimização da taxa de erro e da razão entre o número de nós das OPTs após o processo de poda e o número total de padrões de treinamento (taxa de nós não podados). Para se obter uma função objetivo simples de minimização a priori (DUBOIS-LACOSTE; LÓPEZ-IBÁÑEZ; STÜTZLE, 2013), foi ponderada a relação entre o erro e a taxa de nós não podados por um fator $\alpha \in [0, 1]$. A

função de *fitness* é descrita como segue na Equação 4.7.

$$fitness(\mathbf{c}) = \alpha * error_rate(\mathbf{c}) + (1 - \alpha) * node_rate(\mathbf{c}), \quad (4.7)$$

onde $error_rate(\mathbf{c})$ é o valor complementar da taxa de acerto⁵ obtido pelo classificador OPF, $\alpha \in [0, 1]$ é um fator de importância que se dá a redução na taxa de erro⁶; e

$$node_rate(\mathbf{c}) = \frac{\sum_{i=1}^N f(g_i)[g_i = 1]}{|Z_1|}, \quad (4.8)$$

em que $f(g_i)$ denota a quantidade de nós da i -ésima OPT e $|Z_1|$ o tamanho do conjunto de treinamento.

4.3.3.3 Operadores genéticos

O método de seleção utilizado foi o da roleta (Seção 3.6.1.1). Tal mecanismo consiste basicamente em computar a aptidão relativa de um indivíduo em relação aos demais para medir seu fator de influência na roleta. Quanto melhor é a aptidão de um indivíduo, maior a chance dele ser selecionado para a reprodução e/ou mutação.

O operador de reprodução usado foi o cruzamento de único ponto (Seção 3.6.2.1). Nesse procedimento, um ponto de corte no intervalo $[1, (\text{tamanho do indivíduo} - 1)]$ é escolhido aleatoriamente para dividir em duas partes os dois cromossomos que serão cruzados. As partições dos pais geradas após esse processo são combinadas para produzir novos descendentes contendo informações genéticas de ambos os pais.

O operador de mutação usado nesta proposta foi o *bit-flip* (Seção 3.6.3.1), normalmente usado em indivíduos binários, onde uma pequena porção de genes estimada em 10% do total é selecionada para terem seus valores invertidos, ou seja, genes com valores iguais a 0 são alterados para 1 e vice-versa.

4.3.3.4 Algoritmo MOGAP-OPF

Na linha 1, o classificador OPF é projetado pelo Algoritmo 1 usando o conjunto inicial de treino sem poda Z_1 . A linha 2 inicializa o parâmetro t usado para controlar o número de gerações. Em seguida, nas linhas 3 – 4 é criado aleatoriamente a população inicial com s indivíduos. O *loop* nas linhas 5 – 7 avalia o *fitness* de cada indivíduo da população inicial, usando a Equação 4.7. No *loop* principal (linhas 8 – 19), os indivíduos são selecionados para serem reproduzidos e mutados através de seus operadores específicos (linhas 9 – 11). Então, na linha 12, uma parte dos melhores indivíduos é selecionada pelo processo de elitismo. Depois disso, todos os indivíduos selecionados são combinados

⁵ O método usado para calcular as taxas de acerto é aquele apresentado na Equação 4.4. Esta mesma equação foi utilizada para calcular a taxa de acerto no conjunto Z_3 .

⁶ Quanto menor o valor de α , menor o número de nós de treinamento.

Algoritmo 8 Algoritmo MOGAP-OPF

Entradas: Z_1 : conjunto de treino, Z_2 : conjunto de validação, t_{max} : o número máximo de gerações, s : tamanho da população, α : o custo de poda.

Saídas: Classificador OPF com o conjunto de treinamento reduzido i.e., $\mathbf{c}^1$.

```

1: Execute o Alg. 1 em  $Z_1$  para obter o classificador OPF
2:  $t \leftarrow 0$ 
3:  $P(t) \leftarrow geraPopulacaoInicial(s)$ 
4: Considere um conjunto  $\{\mathbf{c}\}_{i=1}^s$ , onde  $s$  é o número de indivíduos em  $P(t)$ , criado aleatoriamente
5: para todo indivíduo  $i \in P(t)$  faça
6:    $i.fitness \leftarrow funcaoFitness(\mathbf{c}^i, \alpha, Z_2)$ 
7: fim para
8: enquanto  $t \leq t_{max}$  ou outro critério de parada não é satisfeito faça
9:    $[ mutacaoSel, reproducaoSel ] \leftarrow P(t).selecao(taxaMutacao, taxaReproducao)$ 
10:   $individuosMutados \leftarrow mutacao(mutacaoSel)$ 
11:   $individuosReproduzidos \leftarrow reproducao(reproducaoSel)$ 
12:   $individuosElitizados \leftarrow elitismo(P(t), taxaElitismo)$ 
13:   $P(s)_{novo} \leftarrow individuosMutados \cup individuosReproduzidos \cup individuosElitizados$ 
14:  para todo individual  $j \in P(t)_{novo}$  faça
15:     $j.fitness \leftarrow funcaoFitness(\mathbf{c}^j, \alpha, Z_2)$ 
16:  fim para
17:   $P(t) \leftarrow P(t)_{novo}$ 
18:   $t \leftarrow t + 1$ 
19: fim enquanto
20: devolve  $P(t).melhorSolucao()$ 

```

para gerar uma nova população (linha 13). O *loop* nas linhas 14 – 16 calcula o valor de aptidão para cada indivíduo dessa nova população. A população é atualizada na linha 17 e o processo é repetido até que pelo menos um dos critérios de parada sejam atendidos. Após alcançado pelo menos um desses critérios, a melhor solução atual é retornada (linha 20), ou seja, o classificador OPF podado com o melhor valor de *fitness*.

4.4 Conclusão

Os três métodos propostos nesta Seção, GAP-OPF, FSGAP-OPF e MOGAP-OPF, são técnicas baseadas em GAs que buscam como principal objetivo reduzir o número de padrões no conjunto de treino e ao mesmo tempo manter ou até mesmo melhorar o desempenho do classificador. Essa redução de padrões no treinamento faz com que o tempo de classificação diminua consideravelmente, uma vez que, na etapa de teste, é necessário computar a Equação 2.3 para todos os padrões de treinamento.

Análogo aos métodos propostos no parágrafo anterior, as técnicas OPF-TPPA e ROB-OPF visam como principal objetivo eliminar padrões irrelevantes do conjunto de treinamento, a fim de diminuir o tempo de classificação enquanto mantém o desempenho

do classificador em um nível similar ou até melhor. Para isso, no OPF-TPPA, é adicionado um processo de marcação para as amostras relevantes, onde aqueles padrões não marcados são eliminados do conjunto de treinamento. No caso do ROB-OPF, existe também esse estágio de marcação e eliminação dos nós irrelevantes, no entanto, inicialmente, é realizado um procedimento de aprendizagem capaz de determinar as amostras mais representativas para compor os conjuntos de treinamento e de validação.

Do ponto de vista dos algoritmos de poda baseados em GAs, GAP-OPF caracteriza-se por ser uma técnica de único objetivo guiada por uma função que visa minimizar a taxa de erro de classificação sobre um conjunto de validação. Já no FSGAP-OPF, a função mono-objetivo guiada por minimizar o erro de classificação se mantém similar ao método anterior, no entanto, a taxa de poda desejada é estabelecida antes que o processo de otimização ocorra. No MOGAP-OPF, a função de *fitness* que deseja-se otimizar é multiobjetivo, propondo-se a minimizar tanto a taxa de erro de classificação sobre um conjunto de validação, como também o número de OPTs gerados pelo Algoritmo OPF. Além disso, uma grande vantagem na codificação do MOGAP-OPF comparado aos demais, está no seu tamanho reduzido para a representação dos indivíduos. Para esse algoritmo, o comprimento do indivíduo é igual ao número de OPTs geradas a partir do modelo treinado, enquanto que no GAP-OPF e no FSGAP-OPF, cada *bit* da solução representa um padrão de treinamento.

5 Simulações Computacionais

Este Capítulo busca explicar a metodologia utilizada para avaliar as etapas de classificação, validação e teste, bem como fazer uma comparação em termos de acurácia, poda e tempo de classificação para as diferentes abordagens propostas diante das técnicas presentes no estado da arte, além do próprio OPF original (sem poda). Todos os métodos exibidos neste Capítulo são baseados no classificador OPF.

5.1 Metodologia

A fim de averiguar a capacidade de generalização das técnicas propostas a seguir, foram utilizadas diversas bases para problemas de classificação do mundo real obtidas a partir do repositório de aprendizagem de máquina UCI ¹, são elas: *Australian Credit Approval* (ACA), *Credit Approval* (CA), *Breast Cancer Wisconsin* (BCW), *Vertebral Column* (VC), *Dermatology* (DM), *Pima Indians Diabetes* (PID), *Balance Scale* (BS), *Leaf* (LE), *Iris Flower* (DM) e *Haberman's Survival* (HS). Mais detalhes de cada base como quantidade de atributos, classes e etc, podem ser averiguadas nas subseções seguintes.

Para certificar a performance das propostas, note que foram utilizados bases com diferentes características, desde de problemas mais simples que podem ser resolvidos por classificadores que projetam apenas superfícies de decisão linear (*Iris Flower*), como também para problemas mais complexos não-lineares (*Vertebral Column*). Além disso, foram testadas bases balanceadas (*Iris Flower*), desbalanceadas (*Haberman's Survival*), de natureza binária (*Pima Indians Diabetes*), multiclasse (*Leaf*) e com dezenas de dimensões (*Dermatology*).

Todos os algoritmos foram implementados utilizando o ambiente de programação do *Matlab*. A Equação 5.1 (JAIN; NANDAKUMAR; ROSS, 2005) foi utilizada para efetuar a normalização dos padrões na etapa de pré-processamento dos dados. Esse processo foi realizado para cada base de dados.

$$x_{i_{norm}} = \frac{x_i - \mu_x}{\sigma_x}, \quad (5.1)$$

onde $x_{i_{norm}}$ é a i -ésima amostra normalizada, x_i o i -ésimo padrão, μ_x e σ_x a média e o desvio padrão do conjunto de dados, respectivamente, com i variando de 1 a N (total de padrões). Note que após transformado os dados, a média passa a ser 0 e a variância unitária. Dessa forma, a discrepância de valores entre os atributos de cada padrão é uniformizada e mantida na mesma escala, reduzindo assim o efeito dos *outliers* na distribuição dos dados (JAYALAKSHMI; SANTHAKUMARAN, 2011).

¹ <http://archive.ics.uci.edu/ml/>

A Equação 5.5 (PAPA; FALCAO; SUZUKI, 2009) foi utilizada para computar a taxa de acerto em todos os classificadores apresentados nesta dissertação. O resultado da operação leva em consideração classes de diferentes tamanhos. Portanto, são ponderadas situações em que o classificador atribui o rótulo da classe majoritária para a minoritária, devido ao comportamento tendencioso por conta do desbalanceamento de classes entre os padrões.

Considere $|Z_2^{c_i}|$ o número de elementos da classe i no conjunto Z_2 , com i variando de 1 até a quantidade de classes c , seguido pelo número total de padrões no conjunto Z_2 ($|Z_2|$). Além disso, observe a presença das variáveis **FP**(i) (Equação 5.2) e **FN**(i) (Equação 5.3), os falsos positivos e falsos negativos de cada classe i , respectivamente. O **FP**(i) consiste na contagem das amostras que foram classificadas como sendo da classe i , mas que verdadeiramente pertencem a outra classe diferente de i no conjunto Z_2 , enquanto que o **FN**(i) quantifica os padrões que são verdadeiramente da classe i , mas que foram classificados como sendo de outra classe diferente de i no mesmo conjunto Z_2 .

$$e_{i,1} = \frac{\mathbf{FP}(i)}{|Z_2| - |Z_2^{c_i}|} \quad (5.2)$$

$$e_{i,2} = \frac{\mathbf{FN}(i)}{|Z_2^{c_i}|} \quad (5.3)$$

A Equação 5.4 soma cada parcela $e_{i,1}$ e $e_{i,2}$ para compor o erro parcial de cada classe i .

$$E(i) = e_{i,1} + e_{i,2} \quad (5.4)$$

Por fim, a acurácia adaptada é calculada com base na Equação 5.5, onde c representa a quantidade de classes.

$$acc = \frac{2c - \sum_{i=1}^c E(i)}{2c} = 1 - \frac{\sum_{i=1}^c E(i)}{2c} \quad (5.5)$$

A Equação 5.6 apresenta o resultado da taxa de poda, dada pela razão entre o número de amostras podadas e pelo total de padrões no conjunto de treinamento:

$$pruning_rate = \frac{|Z_1 \setminus Z'_1|}{|Z_1|}, \quad (5.6)$$

onde $|Z_1|$ compreende ao tamanho do conjunto de treinamento e Z'_1 o conjunto de treinamento podado.

Na primeira proposta apresentada na Seção 5.2.1, os dados foram inicialmente particionados de forma igual (50%) para treinamento e teste, a fim de comprovar a robustez

do classificador OPF diante de algoritmos clássicos no estado da arte como a rede MLP, por exemplo. Posteriormente, os dados foram divididos em treino (Z_1), validação (Z_2) e teste (Z_3) com os valores de 60%, 20% e 20% dos dados, respectivamente. Essa mesma estratégia de particionar os dados em três partes nas proporções citadas anteriormente, foram também utilizadas nas duas propostas seguintes (Seções 5.2.2 e 5.2.3).

Um resumo geral da metodologia empregada para os modelos apresentadas, bem como a definição de parâmetros na composição do GA, podem ser consultadas na Tabela 2.

Tabela 2 – Resumo geral dos parâmetros empregadas nas abordagens propostas.

Normalização dos dados	Equação 5.1
Taxa de acerto	Equação 5.5
Taxa de poda	Equação 5.6
Divisão da base	60% para treino, 20% para validação e 20% para teste
Número de realizações	20
Número de gerações do GA	No máximo 150 ou até atingir o critério de parada (adotado apenas nas abordagens 5.2.2 e 5.2.3)
Tamanho da população do GA	100
Taxa de reprodução do GA	80%
Taxa de mutação do GA	10%
Taxa de elitismo do GA	10%

Fonte: Próprio autor.

As medidas de desempenho apuradas a partir dos valores médios para as 20 execuções independentes são:

- **taxa de poda** expressa em porcentagem, que consiste na redução obtida no conjunto de treinamento na etapa de validação.
- **acurácia** usada para avaliar o desempenho dos modelos, também exibida em porcentagem.
- **tempo de classificação** estimado para classificar os padrões na etapa de teste, representado em segundos.

Para cada uma das 20 realizações, foram aproveitados os mesmos conjuntos Z_1 , Z_2 e Z_3 para cada modelo, no intuito de tornar mais fiel a comparação entre os mesmos.

5.2 Resultados

Esta seção tem como objetivo apresentar os resultados alcançados pelos métodos propostos nesta dissertação, diante das abordagens existentes no estado da arte. Como será apresentado a seguir, esses resultados mostraram uma constante evolução, partindo da primeira proposta, intitulada GAP-OPF, até o modelo mais recentemente

proposto MOGAP-OPF. Os valores em negrito expostos nas tabelas de resultados a seguir, representam os melhores valores alcançados para uma determinada métrica e base.

5.2.1 Para o classificador GAP-OPF (Proposta 1)

As simulações realizadas nesta proposta foram divididas em duas partes: (i) certificação de robustez do método OPF, ao ser comparado com métodos clássicos, tais como k vizinhos mais próximos (k -NN) e Perceptron Multicamadas (MLP). Nessa etapa, apresentada na Seção 5.2.1.1, 50% dos dados foram selecionados aleatoriamente para o conjunto de treinamento Z_1 , enquanto os 50% restantes utilizados para compor os dados de teste Z_3 . As métricas utilizadas foram a acurácia e o desvio padrão medidos a partir de 20 taxas de acerto, para fins de comparação entre os classificadores. A configuração da rede MLP foi definida como $x : y : z$, onde $x = n$ (número de atributos), $y = \frac{n+c}{2}$ e $z = c$ (número de classes) representam o número de neurônios das camadas de entrada, oculta e saída, respectivamente (HEATON, 2008). Para a implementação do k -NN, foi definido $k = \sqrt{\frac{|Z_1|}{2}}$ (HASSANAT et al., 2014); (ii) na segunda parte (Seção 5.2.1.2), são apurados os resultados obtidos pelos métodos de poda GAP-OPF, OPF-TPPA (Seção 4.2.1), bem como o OPF original sem poda. Para esse procedimento, foi necessário particionar os dados de tal modo que fosse gerado um novo conjunto de validação Z_2 . Deste modo, 60% dos padrões de dados foram selecionados aleatoriamente para compor Z_1 (treinar o modelo OPF), 20% para Z_2 e os 20% restantes para o conjunto de teste Z_3 (avaliado com Z_1 podado). Nessa metodologia, similar a etapa anterior, foram executadas 20 realizações para calcular não só a acurácia, mas também a taxa de poda média (*Pruning Rate*, #PR) e o tempo de classificação médio (*Classification Time*, #CT) para cada algoritmo.

A Floresta de Caminhos Ótimos é construída para o GAP-OPF, utilizando o conjunto de treinamento previamente definido, Z_1 . Então, o GA cria aleatoriamente uma população de soluções factíveis, cada uma delas contendo uma cadeia de valores binários, em que cada *bit* representa a presença (1) ou ausência (0) de uma amostra em Z_1 . A composição de novos indivíduos para as gerações subsequentes é dada por: 80%, gerados pelo operador de cruzamento de único ponto (Seção 3.6.2.1); 10% dos melhores indivíduos da população atual (elitismo); e o restante resultado do processo de mutação *bit-flip* (Seção 3.6.3.1). Cada população possui 100 indivíduos e o número de gerações é 150. Após finalizada essa quantidade de gerações, é retornada a melhor solução encontrada, que corresponde ao indivíduo com a menor taxa de erro de classificação em Z_2 . Dessa forma, o modelo OPF é reformulado, que é representado pelo subconjunto de Z_1 . Por fim, o classificador GAP-OPF introduz esse novo conjunto podado Z_1 para ser testado em Z_3 , no intuito de comparar sua capacidade de generalização frente aos outros dois métodos OPF-TPPA e OPF tradicional.

Foram utilizados seis conjuntos de dados obtidos do repositório de aprendizagem

de máquina UCI²: *Breast Cancer Wisconsin* (BCW), *Dermatology* (DM), *Iris Flower* (IF), *Haberman's Survival* (HS), *Pima Indians Diabetes* (PID) e *Vertebral Column* (VC). A Tabela 3 apresenta algumas informações sobre os conjuntos de dados avaliados, como nome do conjunto de dados, abreviação, quantidade de padrões (# Padrões), atributos (# Atributos) e classes (# Classes).

Tabela 3 – Lista das bases de dados avaliadas nesta proposta.

Base de dados	Abreviação	# Padrões	# Atributos	# Classes
<i>Breast Cancer Wisconsin</i>	BCW	683	10	2
<i>Dermatology</i>	DM	366	34	6
<i>Iris Flower</i>	IF	150	4	3
<i>Haberman's Survival</i>	HS	306	3	2
<i>Pima Indians Diabetes</i>	PID	768	8	2
<i>Vertebral Column</i>	VC	310	6	3

Fonte: Próprio autor.

5.2.1.1 Efetividade do Algoritmo Floresta de Caminhos Ótimos

Esta seção tem como principal finalidade apresentar a robustez do classificador OPF (sem poda), comparado-o com técnicas clássicas do estado da arte, tais como k -NN e MLP. As métricas de desempenho avaliadas foram a acurácia e desvio padrão da taxa de acerto, calculadas sobre 20 execuções independentes.

Tabela 4 – Valores de acurácia e desvio padrão para os classificadores OPF, K -NN e MLP com metade da base (50%) utilizada para treino e o restante (50%) usado para teste.

Base de dados	OPF	k -NN	MLP
BCW	94,74% $\pm$ 1,41	95,16% $\pm$ 1,24	96,50% $\pm$ 1,28
DM	96,28% $\pm$ 0,84	95,86% $\pm$ 1,46	93,80% $\pm$ 3,28
IF	95,78% $\pm$ 1,83	94,72% $\pm$ 2,25	96,30% $\pm$ 2,34
HS	57,37% $\pm$ 2,89	53,95% $\pm$ 2,21	55,07% $\pm$ 3,52
PID	65,86% $\pm$ 2,26	65,46% $\pm$ 1,88	72,76% $\pm$ 2,08
VC	81,10% $\pm$ 2,25	81,50% $\pm$ 2,36	74,99% $\pm$ 2,81

Fonte: Próprio autor,

Analisando a Tabela 4, note que o classificador OPF obteve resultados semelhantes aos obtidos pelos outros classificadores, sendo que, em alguns casos, para os conjuntos de dados DM e HS, a acurácia do OPF foi superior.

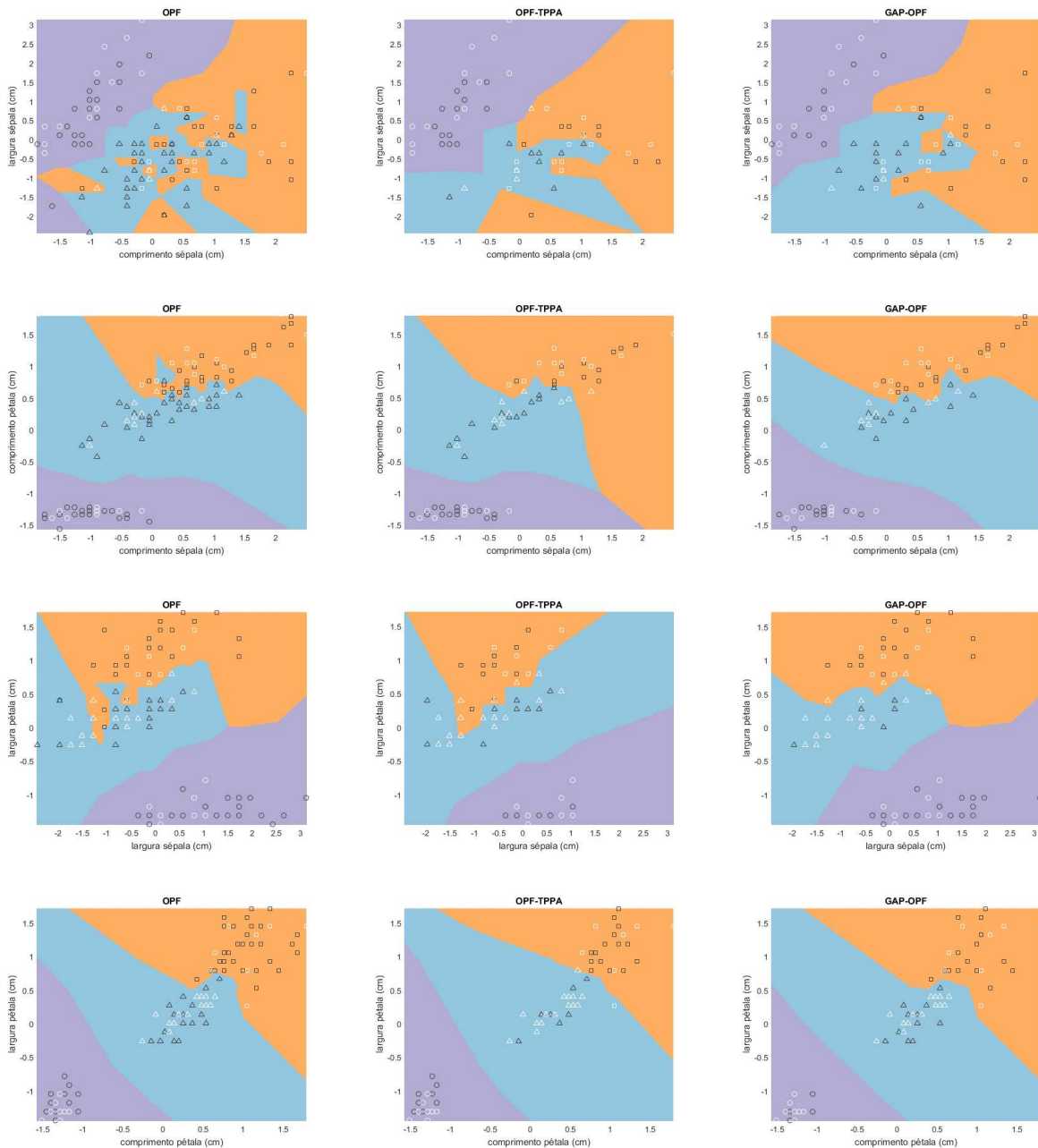
5.2.1.2 Efetividade do GAP-OPF

Antes de realizar os testes computacionais nas bases de dados utilizando todos os atributos, na Figura 15, é apresentado o mapa de cores em uma única realização para

² <http://archive.ics.uci.edu/ml/>

o problema da *Iris Flower*, levando em consideração apenas os atributos **comprimento sépala** *vs* **largura sépala** seguida de mais outras três combinações descritas em cada linha. Os padrões em preto representam o conjunto de treinamento e as amostras destacadas em branco o conjunto de teste. Além disso, os padrões circulares representam a classe **setosa**, as amostras quadriculadas a classe **virgínica** e as retangulares a classe **versicolor**. Para uma visualização mais detalhada dos dados, acesse o link <<https://www.dropbox.com/sh/dstg1uhf5p38enh/AADdvjIw-oH7WVNLkW5FioWja?dl=0>>.

Figura 15 – Mapa de cores para a base de dados *Iris Flower*. Em cada linha, são apresentados os três métodos OPF, OPF-TPPA e GAP-OPF, bem como suas distribuições para cada par de atributos.



Fonte: Próprio autor.

Ainda na Figura 15, note que a quantidade de padrões de treinamento presentes nos métodos de poda (OPF-TPPA e GAP-OPF) é inferior ao método OPF puro sem poda. Além do mais, esses métodos otimizados tendem a suavizar suas superfícies de decisão, bem como eliminar sobreposição entre as classes. Deste modo, no geral, OPF-TPPA e GAP-OPF alcançaram resultados melhores que o próprio OPF em termos de taxa de acerto. Com relação a taxa de poda, GAP-OPF e OPF-TPPA obtiveram níveis similares. Mais detalhes com respeito a valores específicos para todos os classificadores utilizando as bases de dados completa, podem ser consultados na Tabela 5.

A seguir, é apresentada a capacidade que o modelo GAP-OPF tem de gerar conjuntos compactos de treinamento, mantendo ou até mesmo melhorando a precisão do classificador. Seu desempenho é comparado com aqueles obtidos pelo OPF-TPPA (Seção 4.2.1) e o OPF antes da poda (OPF original, Seção 2).

A divisão dos dados é disposta por selecionar aleatoriamente 60% das amostras para o conjunto Z_1 , 20% para Z_2 e os 20% restantes para Z_3 . Em seguida, o classificador OPF é treinado utilizando o conjunto Z_1 , onde esse modelo gerado é repassado como entrada para o GAP-OPF e o OPF-TPPA. Com isso, é garantido uma comparação mais justa entre as técnicas de poda, uma vez que, cada método recebe como entrada a mesma instância do modelo OPF.

Na Tabela 5, são apresentados os resultados obtidos pelos classificadores OPF, OPF-TPPA e GAP-OPF. Esses valores são a acurácia e o desvio padrão em relação ao conjunto de testes Z_3 , a taxa de poda média ($\#PR$) em Z_1 e o tempo médio de classificação ($\#CT$), em segundos, gasto para classificar Z_3 .

Tabela 5 – Resultados para OPF antes da poda (OPF), OPF-TPPA e GAP-OPF com 60%, 20% e 20% do conjunto de dados particionados para treinamento, validação e teste, respectivamente. O valor da métrica #CT está em segundos.

Base de dados	Modelo	Acurácia	# PR	# CT
BCW	OPF	94,32% $\pm$ 1,93	—	0,2035
BCW	OPF-TPPA	94,45% $\pm$ 1,51	69,95%	0,0686
BCW	GAP-OPF	95,94% $\pm$ 2,07	49,75%	0,1067
DM	OPF	96,31% $\pm$ 1,64	—	0,0703
DM	OPF-TPPA	95,93% $\pm$ 1,74	60,73%	0,0310
DM	GAP-OPF	96,00% $\pm$ 1,76	50,80%	0,0376
IF	OPF	94,31% $\pm$ 3,18	—	0,0120
IF	OPF-TPPA	93,37% $\pm$ 2,93	57,89%	0,0061
IF	GAP-OPF	93,91% $\pm$ 2,90	49,78%	0,0068
HS	OPF	55,97% $\pm$ 8,00	—	0,0393
HS	OPF-TPPA	57,93% $\pm$ 9,16	60,74%	0,0179
HS	GAP-OPF	58,12% $\pm$ 6,33	52,73%	0,0205
PID	OPF	65,13% $\pm$ 3,34	—	0,2753
PID	OPF-TPPA	65,94% $\pm$ 4,36	62,26%	0,1115
PID	GAP-OPF	66,49% $\pm$ 4,47	51,67%	0,1381
VC	OPF	81,27% $\pm$ 4,71	—	0,0383
VC	OPF-TPPA	78,54% $\pm$ 5,64	57,45%	0,0178
VC	GAP-OPF	81,95% $\pm$ 3,56	51,80%	0,0196

Fonte: Próprio autor.

Como pode ser visto na Tabela 5, o método proposto nessa seção, GAP-OPF, obteve resultados superiores em termos de acurácia para todas as bases, quando comparado apenas com o outro método de poda OPF-TPPA. Na maioria dos casos, GAP-OPF alcançou melhores taxas de acerto do que próprio modelo original OPF, sendo nas bases BCW, HS, PID e CV. Além disso, o tempo de classificação e o tamanho de Z_1 diminuíram em torno da metade de seus valores anteriores produzidos pelo OPF. No entanto, o OPF-TPPA alcançou melhores resultados do que o GAP-OPF, quando comparado aos valores de # PR e # CT.

5.2.2 Para o classificador FSGAP-OPF (Proposta 2)

Esta seção busca avaliar a capacidade do FSGAP-OPF de gerar um conjunto compacto de treinamento, mantendo a precisão do classificador em um nível similar ou até mesmo melhorando. Seu desempenho foi comparado com aqueles obtidos pelo OPF-TPPA (Seção 4.2.1), ROB-OPF (Seção 4.2.2) e OPF antes da poda. As bases de dados utilizadas para fazer essa análise comparativa foram *Balance Scale* (BS), *Australian Credit Approval* (ACA), *Haberman's Survival* (HS), *Leaf* (LE), *Pima Indians Diabetes* (PID) e *Vertebral Column* (VC) apresentadas na Tabela 6.

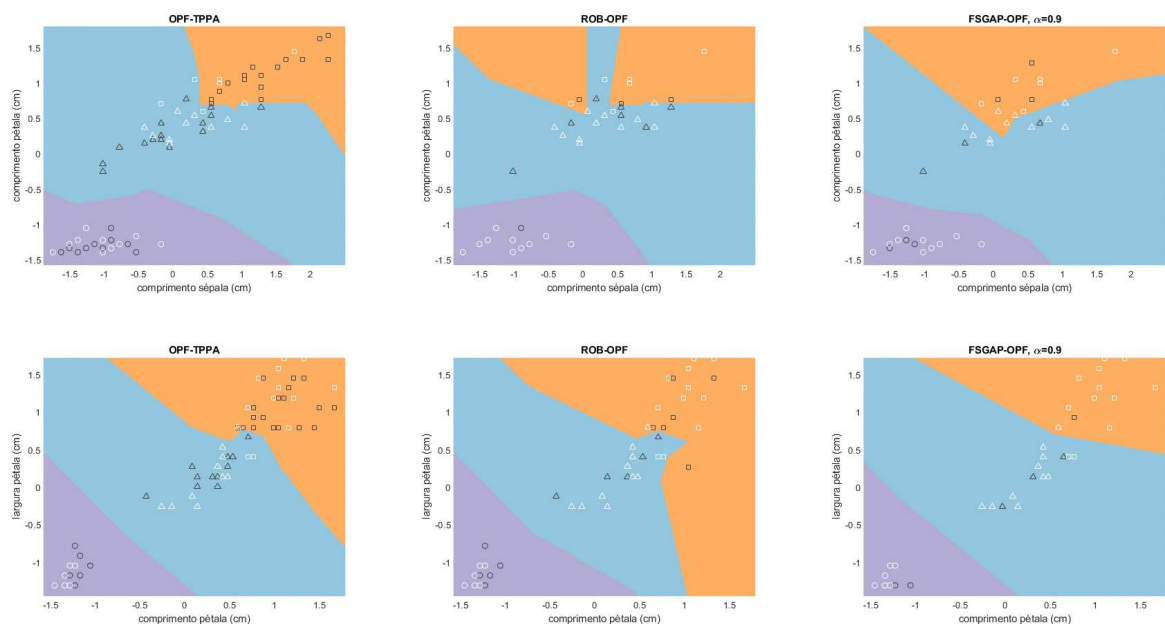
Tabela 6 – Lista das bases de dados avaliadas nesta proposta.

Base de dados	Abreviação	# Padrões	# Atributos	# Classes
<i>Balance Scale</i>	BS	625	4	3
<i>Australian Credit Approval</i>	ACA	690	14	2
<i>Haberman's Survival</i>	HS	306	3	2
<i>Leaf</i>	LE	340	15	15
<i>Pima Indians Diabetes</i>	PID	768	8	2
<i>Vertebral Column</i>	VC	310	6	3

Fonte: Próprio autor.

Análogo a metodologia inicial empregada na Subseção 5.2.1.2, a Figura 16 mostra as superfícies de decisão projetadas pelos classificadores OPF-TPPA, ROB-OPF e FSGAP-OPF em duas diferentes condições na base de dados *Iris Flower*: para os pares de atributos **comprimento sépala vs comprimento pétala** e **comprimento pétala vs largura pétala**. Note que, mesmo após uma redução significativa de 90% no conjunto de dados para o classificador FSGAP-OPF, ainda assim, a superfície de decisão projetada mantém um nível adequado de classificação. Na primeira condição, FSGAP-OPF supera os demais métodos de poda, enquanto que na segunda condição, FSGAP-OPF ainda é o melhor se igualando a taxa de acerto obtido pelo ROB-OPF. Observe também que a poda efetivada pelo FSGAP-OPF sempre foi a maior de todas. Para uma visualização mais detalhada dos dados, acesse o link <<https://www.dropbox.com/sh/8oqnrpuvnmqpu5/AAAJdQeK73luaUDdP7Gau5eBa?dl=0>>.

Figura 16 – Mapa de cores para a base de dados *Iris Flower*. Cada linha dispõe dos três métodos de poda OPF-TPPA, ROB-OPF e FSGAP-OPF com $\alpha = 0,9$, bem como suas distribuições para cada par de atributos.



Fonte: Próprio autor.

Os dados são particionados de tal modo que 60% das amostras são selecionadas aleatoriamente para integrar o conjunto de treinamento (Z_1), 20% para serem utilizadas na validação (conjunto Z_2) e os 20% restantes para constituir a partição de teste (conjunto Z_3). Após essa etapa, o classificador OPF é treinado utilizando o conjunto Z_1 , para que seu modelo treinado seja repassado como entrada para os classificadores com poda FSGAP-OPF, OPF-TPPA e ROB-OPF. Note que a instância do objeto treinado OPF é a mesma para cada classificador, assim garantindo uma maior fidelidade de comparação entre os métodos.

A população inicial do GA é criada de maneira aleatória, no qual cada indivíduo é representando por um vetor de bits binário, onde cada gene indica a presença (1) ou ausência (0) de um determinado padrão do conjunto de treinamento Z_1 . As proporções utilizadas para construir gerações futuras são dadas por aplicar 80% de reprodução (Seção 4.3.2.3), 10% de elitismo e os 10% restantes pelo processo de mutação (Seção 4.3.2.4). Cada população possui um tamanho fixo de 100 indivíduos e a quantidade máxima de geração é 150. Nem sempre, esse valor máximo é executado pelo algoritmo, visto que a melhor solução atual encontrada deve ser retornada com antecedência, caso o módulo da diferença entre a média dos valores de *fitness* da população anterior com a população corrente seja inferior ou igual a 10^{-5} , indicando uma possível convergência nos valores de *fitness*. Quando ao menos um dos critérios de parada for satisfeito, seja pelo número máximo de gerações ou pela condição citada anteriormente, o melhor indivíduo atual é retornado ($\mathbf{c}^1$ que representa um subconjunto de Z_1 com a maior taxa de acerto na classificação em Z_2). Por fim, o modelo reduzido Z_1 está pronto para ser testado no conjunto Z_3 .

Na Tabela 7 são mostrados os resultados relacionados ao OPF, OPF-TPPA, ROB-OPF com $T = 50$ e FSGAP-OPF quando aplicados aos conjuntos de dados BS, ACA, HS, LE, PID e VC para 20 realizações. Esses resultados são os valores da acurácia (*Accuracy*, ACC) medido no conjunto de teste Z_3 , a taxa média de poda (*Pruning Rate*, #PR) em Z_1 e o tempo médio de classificação (*Classification Time*, #CT), em segundos, requerido para classificar Z_3 . Além disso, o desvio padrão para cada métrica avaliada foi calculado também, no intuito de mensurar a variância dos resultados. O símbolo α no algoritmo FSGAP-OPF denota a taxa de poda desejada antes de iniciar o processo de redução, observe por exemplo, que para um $\alpha = 82$, o valor da #PR é igual ou muito próxima a 82%, e assim sucessivamente, indicando uma redução de mais de 80% nos dados de treinamento.

Tabela 7 – Resultados obtidos nesta proposta. Os valores das métricas α , ACC e #PR estão em porcentagem e #CT em segundos.

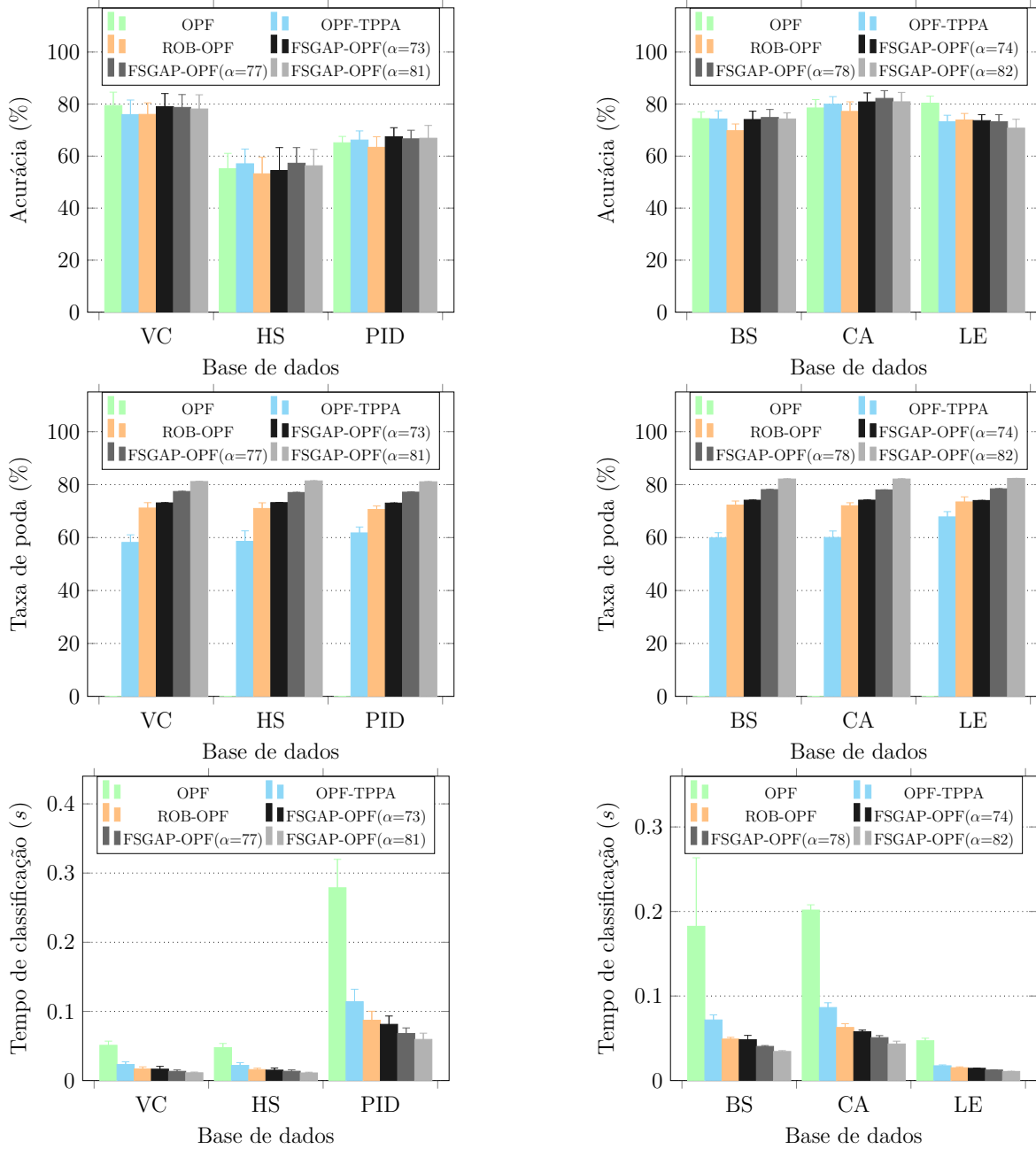
Base	Métrica	OPF	OPF-TPPA	ROB-OPF		FSGAP-OPF	
VC	ACC	79,45 $\pm$ 5,13	75,98 $\pm$ 5,58	76,05 $\pm$ 4,43	79,00 $\pm$ 5,05	78,72 $\pm$ 4,94	78,12 $\pm$ 5,46
	α	—	—	—	73,00	77,00	81,00
	#PR	—	58,17 $\pm$ 2,87	71,16 $\pm$ 2,04	73,12 $\pm$ 0,00	77,42 $\pm$ 0,00	81,18 $\pm$ 0,00
	#CT	0,0510 $\pm$ 0,006	0,0232 $\pm$ 0,004	0,0168 $\pm$ 0,003	0,0165 $\pm$ 0,004	0,0134 $\pm$ 0,002	0,0113 $\pm$ 0,001
HS	ACC	55,15 $\pm$ 5,92	57,07 $\pm$ 5,66	53,21 $\pm$ 6,34	54,49 $\pm$ 8,79	57,28 $\pm$ 6,00	56,29 $\pm$ 6,31
	α	—	—	—	73,00	77,00	81,00
	#PR	—	58,58 $\pm$ 3,96	71,01 $\pm$ 2,13	73,22 $\pm$ 0,00	77,05 $\pm$ 0,00	81,42 $\pm$ 0,00
	#CT	0,0476 $\pm$ 0,006	0,0219 $\pm$ 0,004	0,0159 $\pm$ 0,002	0,0151 $\pm$ 0,003	0,0135 $\pm$ 0,002	0,0111 $\pm$ 0,001
PID	ACC	65,13 $\pm$ 2,43	66,19 $\pm$ 3,49	63,39 $\pm$ 4,06	67,44 $\pm$ 3,48	66,65 $\pm$ 3,29	66,88 $\pm$ 4,91
	α	—	—	—	73,00	77,00	81,00
	#PR	—	61,78 $\pm$ 2,22	70,63 $\pm$ 1,31	73,04 $\pm$ 0,00	77,17 $\pm$ 0,00	81,09 $\pm$ 0,00
	#CT	0,2788 $\pm$ 0,041	0,1140 $\pm$ 0,018	0,0872 $\pm$ 0,013	0,0813 $\pm$ 0,012	0,0681 $\pm$ 0,008	0,0595 $\pm$ 0,009
BS	ACC	74,41 $\pm$ 2,62	74,29 $\pm$ 3,15	69,28 $\pm$ 4,33	74,08 $\pm$ 3,19	74,85 $\pm$ 3,08	74,29 $\pm$ 2,32
	α	—	—	—	74,00	78,00	82,00
	#PR	—	59,89 $\pm$ 1,96	72,75 $\pm$ 1,33	74,13 $\pm$ 0,00	78,13 $\pm$ 0,00	82,13 $\pm$ 0,00
	#CT	0,1825 $\pm$ 0,081	0,0717 $\pm$ 0,006	0,0491 $\pm$ 0,002	0,0484 $\pm$ 0,005	0,0404 $\pm$ 0,001	0,0344 $\pm$ 0,001
ACA	ACC	78,51 $\pm$ 3,23	79,97 $\pm$ 2,89	77,19 $\pm$ 3,68	80,85 $\pm$ 3,45	82,20 $\pm$ 2,97	80,89 $\pm$ 3,60
	α	—	—	—	74,00	78,00	82,00
	#PR	—	60,05 $\pm$ 2,44	72,05 $\pm$ 1,11	74,15 $\pm$ 0,00	78,02 $\pm$ 0,00	82,13 $\pm$ 0,00
	#CT	0,2017 $\pm$ 0,006	0,0864 $\pm$ 0,006	0,0628 $\pm$ 0,004	0,0578 $\pm$ 0,002	0,0508 $\pm$ 0,002	0,0433 $\pm$ 0,003
LE	ACC	80,35 $\pm$ 2,70	73,22 $\pm$ 2,49	73,89 $\pm$ 2,51	73,65 $\pm$ 2,27	73,22 $\pm$ 2,71	70,75 $\pm$ 3,41
	α	—	—	—	74,00	78,00	82,00
	#PR	—	67,82 $\pm$ 1,99	73,48 $\pm$ 1,91	74,02 $\pm$ 0,00	78,43 $\pm$ 0,00	82,35 $\pm$ 0,00
	#CT	0,0471 $\pm$ 0,003	0,0175 $\pm$ 0,001	0,0152 $\pm$ 0,001	0,0145 $\pm$ 0,000	0,0125 $\pm$ 0,000	0,0108 $\pm$ 0,000

Fonte: Próprio autor.

Analisando com mais detalhes a Tabela 7, o método proposto, FSGAP-OPF, conseguiu obter bons resultados em termos de #PR, e consequentemente #CT, quando comparado com as outras técnicas de poda. A redução no tempo de classificação e o tamanho de Z_1 foram expressivos, alcançando os melhores resultados frente aos outros algoritmos. FSGAP-OPF também foi capaz de obter melhores valores de acurácia na maioria das bases de dados, embora em alguns casos o OPF sem poda tenha se saído melhor. Isso mostra que o método proposto pode ser aplicado com segurança em problemas de classificação de maneira eficiente. Outra característica interessante, é que o desvio padrão para a taxa de poda obtida pelo FSGAP-OPF, é igual a zero para todos os casos, indicando subconjuntos de tamanhos iguais partindo de Z_1 . Fato esse, decorrente do α ser constante para as 20 realizações. Por fim, vale ressaltar também que, mesmo quando o valor de α aumenta, ou seja, aumenta a poda, a acurácia do FSGAP-OPF se mantém em um nível similar.

Uma outra alternativa de visualização geral dos resultados apresentados na Tabela 7, pode ser vista na Figura 17, por meio de um gráfico de barras.

Figura 17 – Acurácia, taxa de poda e tempo de classificação obtidos pelos métodos empregados neste trabalho aplicados nos conjuntos de dados apresentados na Tabela 6.



Fonte: Próprio autor.

5.2.3 Para o classificador MOGAP-OPF (Proposta 3)

As simulações computacionais foram experimentadas em 6 bases de dados do mundo real obtidas a partir do repositório de aprendizagem de máquina UCI³, são elas: *Balance Scale* (BS), *Breast Cancer Wisconsin* (BCW), *Credit Approval* (CA), *Vertebral Column* (VC), *Pima Indians Diabetes* (PID) e *Haberman's Survival* (HS). A Tabela 8 apresenta os respectivos valores para cada conjunto de dados usado nesta proposta, tais como

³ <http://archive.ics.uci.edu/ml/>

abreviação, quantidade de atributos (#Atributos), padrões (#Padrões), classes (#Classes) e nome da base.

Tabela 8 – Lista das bases de dados avaliadas nesta proposta.

Base de dados	Abreviação	#Padrões	#Atributos	# Classes
<i>Balance Scale</i>	BS	625	4	3
<i>Breast Cancer Wisconsin</i>	BCW	683	10	2
<i>Credit Approval</i>	CA	690	15	2
<i>Haberman's Survival</i>	HS	306	3	2
<i>Pima Indians Diabetes</i>	PID	768	8	2
<i>Vertebral Column</i>	VC	310	6	3

Fonte: Próprio autor.

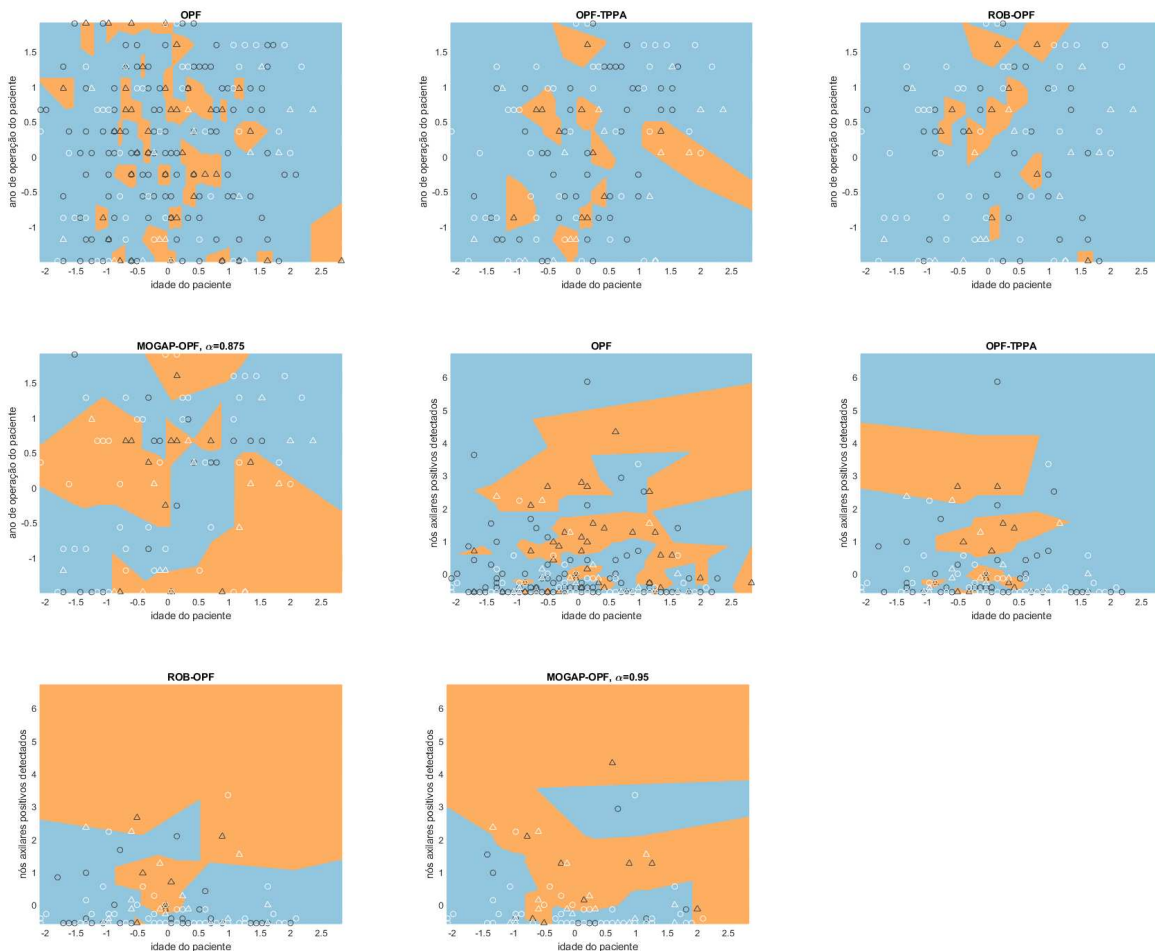
A divisão dos dados é concebida da seguinte forma: 60% selecionados aleatoriamente para constituir o conjunto de treinamento Z_1 , 20% para validação (Z_2) e os 20% restantes para integrar o conjunto de teste Z_3 . A fim de tornar mais justa a comparação entre as técnicas abordadas nesta Seção, foram utilizados os mesmos subconjuntos de dados em cada execução para todos os métodos, sendo no total 20 realizações. De um modo geral, o fluxo utilizando os conjuntos citados anteriormente, inicia pelo treinamento do modelo OPF com a partição Z_1 , ao passo que o conjunto Z_2 é utilizado para potencializar os modelos de poda, e finalmente, poderem ser testados em Z_3 .

A população inicial do GA é construída aleatoriamente, onde cada indivíduo é representado por uma solução factível. Por sua vez, cada solução é constituída por um indivíduo binário, no qual cada gene denota a ausência (0) ou a presença (1) de uma OPT na floresta de caminhos ótimos gerada a partir de Z_1 . Para as gerações seguintes, 80% de novos indivíduos são reproduzidos pelo operador *crossover* de único ponto (Seção 3.6.2.1), o elitismo é aplicado para preservar 10% dos melhores indivíduos e a parcela restante aplicada no processo de mutação *bit-flip* (Seção 3.6.3.1). O número de indivíduos por população é igual a 100 e a quantidade máxima de gerações 150. No entanto, o algoritmo MOGAP-OPF pode ser interrompido de maneira precoce caso o módulo da diferença entre a média dos valores de *fitness* da população anterior com a população corrente seja inferior ou igual a 10^{-5} , indicando que a aptidão dos indivíduos possivelmente convergiu. Após finalizada essa etapa de otimização no GA, a melhor solução encontrada é retornada, ou seja, o indivíduo com a melhor aptidão já pode ser usado para computar a acurácia no conjunto z_3 . Por fim, o modelo proposto nesta Seção, MOGAP-OPF, está pronto para ser validado mediante comparação com os outros métodos. No Apêndice A, é mostrada a evolução das soluções do GA ao longo de 150 gerações.

5.2.3.1 Efetividade do MOGAP-OPF

Esta seção analisa a capacidade do MOGAP-OPF de gerar conjuntos de treinamento representativos sem afetar ou mesmo melhorar sua acurácia. A fim de avaliar os níveis de eficiência fornecidos pela abordagem proposta, suas métricas coletadas foram confrontadas com o OPF sem poda, OPF-TPPA e ROB-OPF. Antes de realizar as simulações propriamente ditas utilizando as bases completamente, foram feitos alguns experimentos similares aos realizados nas Subseções 5.2.1 e 5.2.2. Na Figura 18, é apresentado o mapa de decisão para o problema *Haberman's Survival* utilizando alguns pares de atributos. Para uma visualização mais destacada dos dados, acesse o link <<https://www.dropbox.com/sh/rfg98p9bpz3n1w4/AACJLp79T-hqQyLoEu0B8suza?dl=0>>.

Figura 18 – Mapa de cores para a base de dados *Haberman's Survival*. Os métodos OPF, OPF-TPPA, ROB-OPF e MOGAP-OPF são apresentados, bem como suas distribuições para cada par de atributos.



Fonte: Próprio autor.

Diferentemente dos experimentos anteriores realizados nas subseções 5.2.1 e 5.2.2, a técnica proposta nesta seção, intitulada MOGAP-OPF, foi capaz de superar todos os outros métodos quando analisadas todas as métricas simultaneamente. Note que, principalmente

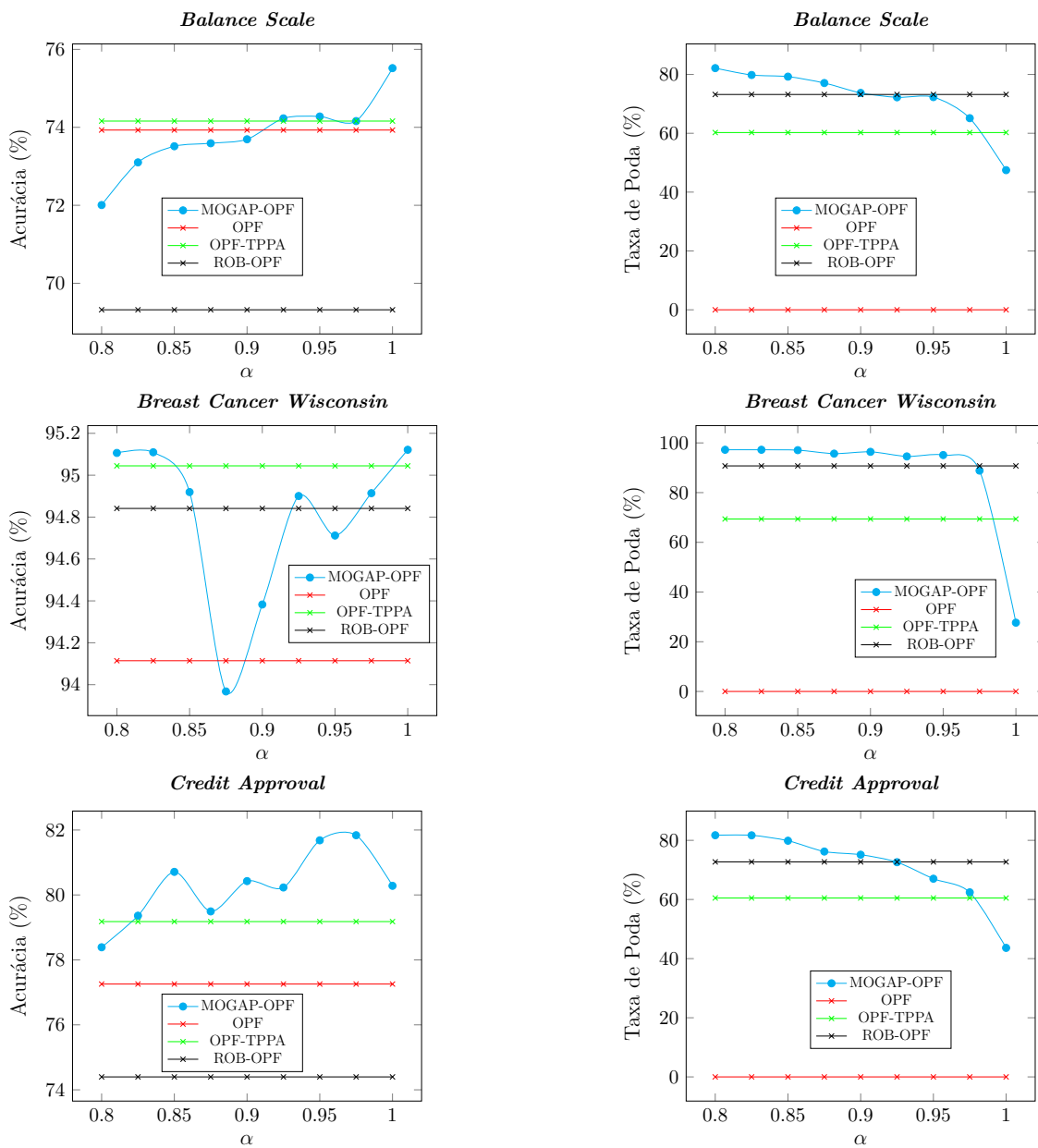
nos modelos OPF e OPF-TPPA, as superfícies de decisão projetadas por esses modelos tendem a estarem sobreajustadas. Para fins de experimentação, foram utilizadas duas combinações de atributos, sendo elas: **idade do paciente vs ano de operação do paciente** e **idade do paciente vs nós auxiliares positivos detectados**.

Os dados foram divididos aleatoriamente em 60% dos padrões para a construção de Z_1 , 20% para Z_2 e os outros 20% para Z_3 . Em seguida, o classificador OPF é treinado em Z_1 para que seu modelo seja repassado como entrada para os classificadores MOGAP-OPF, OPF-TPPA e ROB-OPF com $T = 50$, a fim de oferecer a mesma instância do objeto OPF treinado. Na Tabela 9, são apresentados alguns resultados obtidos pelo MOGAP-OPF, ROB-OPF, OPF-TPPA e o algoritmo tradicional do OPF. Esses resultados são a acurácia (ACC) sobre o conjunto de teste Z_3 , o tempo médio de classificação (#CT), em segundos, gasto para classificar Z_3 e a taxa de poda média (PR) em Z_1 . Os melhores valores obtidos de cada linha estão destacados em negrito.

Como pode ser visto na Tabela 9, o método proposto, MOGAP-OPF, foi capaz de obter bons resultados em termos de acurácia, quando comparado com os outros métodos. Sendo na maioria dos casos, o melhor, com exceção apenas no conjunto de dados VC. Além disso, a redução no tempo de classificação e o tamanho de Z_1 foram bem expressivos, alcançando os melhores resultados em todos os conjuntos de dados. Alternativamente, esses mesmos resultados podem ser vistos nas Figuras 19 e 20 com α varrendo o intervalo $[0, 8; 1]$ para 20 execuções independentes. Note que, a acurácia e taxa de poda geralmente caminham em sentidos contrários, ou seja, a medida que a acurácia aumenta, em contrapartida, a taxa de poda diminui e vice-versa. Fato esse, devido ao processo de otimização multiobjetivo guiado pelo parâmetro α .

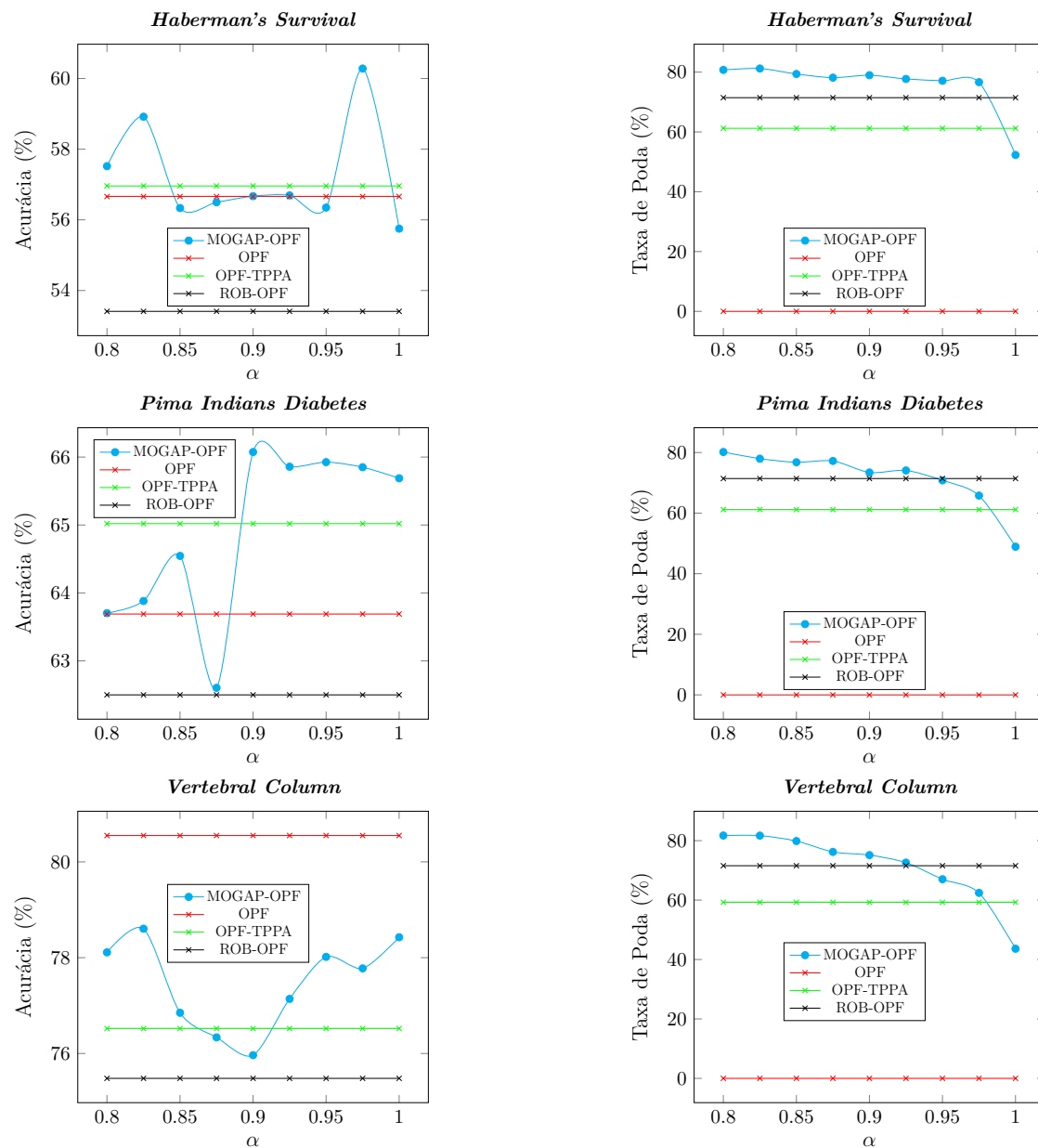
O fator α mede a importância dos parâmetros acurácia e poda no processo de otimização do classificador. Portanto, quanto maior é o α , maior deverá ser a acurácia do classificador e menos é dada importância para a métrica de poda. No entanto, embora o MOGAP-OPF seja sensível a α , ele ainda conseguiu manter um nível adequado de poda mesmo quando o α aumentou, garantindo assim uma qualidade de comparação adequada com os outros métodos.

Figura 19 – Comparação entre os métodos empregados nesse trabalho para as métricas acurácia e taxa de poda com α percorrendo o intervalo entre $[0, 8; 1]$, quando aplicados nas bases de dados *Balance Scale*, *Breast Cancer Wisconsin* e *Credit Approval*



Fonte: Próprio autor.

Figura 20 – Comparação entre os métodos empregados nesse trabalho para as métricas acurácia e taxa de poda com α percorrendo o intervalo entre $[0,8; 1]$, quando aplicados nas bases de dados *Haberman's Survival*, *Pima Indians Diabetes* e *Vertebral Column*



Fonte: Próprio autor.

Tabela 9 – Resultados para OPF, OP-TPPA, ROB-OPF e MOGAP-OPF. Os valores das métricas ACC e PR estão em porcentagem e #CT em segundos.

Base de dados	Métrica	OPF	OPF-TPPA	ROB-OPF	MOGAP-OPF				
					$\alpha = 0,80$	$\alpha = 0,85$	$\alpha = 0,90$	$\alpha = 0,95$	$\alpha = 1,00$
BS	ACC	73,93±2,50	74,16±2,67	69,32±3,28	72,01±2,76	73,51±2,67	73,69±3,86	74,28±2,68	75,52±3,62
	PR	—	60,25±1,82	73,19±1,41	82,15±3,67	79,25±4,72	73,72±6,16	72,29±5,81	47,47±7,04
	#CT	0,1851±0,072	0,0691±0,006	0,0475±0,004	0,0349±0,007	0,0438±0,020	0,0483±0,011	0,0520±0,011	0,0904±0,014
BCW	ACC	94,11±2,32	95,04±2,19	94,84±2,17	95,11±1,74	94,92±2,11	94,38±2,38	94,71±2,24	95,12±1,85
	PR	—	69,39±6,00	90,75±0,96	97,24±2,76	97,08±2,64	96,41±3,59	95,11±7,31	27,66±30,65
	#CT	0,1874±0,027	0,0668±0,024	0,0230±0,003	0,0099±0,006	0,0108±0,006	0,0140±0,011	0,0141±0,014	0,1515±0,082
CA	ACC	77,26±2,80	79,18±4,39	74,40±2,92	78,39±3,22	80,71±3,84	80,43±3,94	81,68±3,66	80,28±3,51
	PR	—	60,49±1,81	72,70±1,57	80,83±4,81	75,33±6,65	69,50±6,01	68,21±6,10	42,03±7,10
	#CT	0,1997±0,014	0,0848±0,008	0,0591±0,006	0,0447±0,011	0,0566±0,015	0,0684±0,012	0,0678±0,013	0,1184±0,019
HS	ACC	56,66±5,05	56,96±6,53	53,41±6,49	57,52±5,92	56,33±6,90	56,67±6,31	56,35±7,67	55,75±7,59
	PR	—	61,20±3,25	71,45±2,49	80,74±4,34	79,37±4,54	78,96±4,56	77,10±4,79	52,29±9,23
	#CT	0,0453±0,005	0,0207±0,004	0,0158±0,004	0,0112±0,003	0,0118±0,003	0,0124±0,003	0,0131±0,003	0,0240±0,005
PID	ACC	63,69±3,37	65,02±3,37	62,50±4,04	63,70±2,38	64,54±4,55	66,07±3,83	65,93±5,35	65,69±4,60
	PR	—	61,18±2,82	71,41±1,25	80,16±3,80	76,78±4,92	73,41±7,33	70,80±7,36	48,91±7,58
	#CT	0,2650±0,081	0,1160±0,057	0,0809±0,012	0,0642±0,025	0,0661±0,012	0,0797±0,019	0,0842±0,021	0,1399±0,034
VC	ACC	80,55±5,85	76,52±6,26	75,48±4,26	78,11±6,64	76,85±5,25	75,96±6,07	78,02±6,06	78,43±4,86
	PR	—	59,25±2,60	71,56±1,30	81,72±7,01	79,87±9,13	75,14±9,74	67,02±12,18	43,63±13,17
	#CT	0,0408±0,004	0,0195±0,004	0,0136±0,002	0,0098±0,003	0,0110±0,005	0,0125±0,005	0,0159±0,005	0,0246±0,007

Fonte: Próprio autor.

5.3 Conclusão

Esta seção apresentou a metodologia empregada nas propostas GAP-OPF, FSGAP-OPF e MOGAP-OPF, bem como uma avaliação comparativa entre os métodos de poda presentes no estado da arte e as propostas citadas anteriormente. As principais métricas avaliadas foram: acurácia medida sobre o conjunto de teste, percentual de redução no conjunto de treinamento ou taxa de poda, e tempo de classificação necessário para classificar todas as amostras de teste. O desempenho de cada modelo foi analisado em bases de dados do mundo real tais como *Dermatology*, *Pima Indians Diabetes*, *Vertebral Column*, dentre outras.

Tais experimentos reforçam a robustez e capacidade de generalização das propostas apresentadas nesta dissertação, exibindo resultados satisfatórios mesmo quando aplicadas em condições adversas. Tendo em vista que, esses modelos foram testados em diversas bases de dados, bem como para distintos valores de α , no caso do FSGAP-OPF e MOGAP-OPF.

6 Conclusões e trabalhos futuros

Esta seção tem como principal objetivo esclarecer as conclusões e considerações finais colhidas durante o desenvolvimento desta dissertação, bem como sugerir novas ideias de melhorias para as propostas apresentadas. Além disso, são apresentadas as contribuições deste trabalho, alcançando com satisfatoriedade os objetivos levantados nas seções que problematizam a pesquisa.

6.1 Conclusões

Como visto nos experimentos do Capítulo anterior, o OPF baseado em grafo completo oferece algumas vantagens interessantes, pois trata-se de um classificador simples, livre de parâmetros, lida com múltiplas classes sem modificações ou extensões em seu algoritmo e não faz suposições sobre a dispersão ou separabilidade das classes (PAPA; FALCAO; SUZUKI, 2009). Em contrapartida, o tempo de classificação demandado por esse algoritmo é diretamente proporcional ao tamanho do conjunto de treinamento armazenado.

Portanto, algumas propostas foram desenvolvidas no intuito de eliminar padrões irrelevantes ou redundantes do conjunto de treinamento, com o principal propósito de reduzir o tempo de classificação enquanto mantém ou até mesmo melhora o desempenho do classificador. Em Papa e Falcão (2009), é apresentado um modelo capaz de identificar e eliminar amostras irrelevantes do conjunto de treinamento. Por outro lado, em Papa et al. (2010), é elaborada uma melhoria da técnica anterior, por adicionar um processo de aprendizagem antes do estágio de poda.

Modelos baseados em GAs são amplamente utilizados para resolver problemas de otimização de um modo geral. Todavia, essa técnica tem sido pouco difundida para aplicações utilizando o classificador OPF. Em vista disso, foram criadas três propostas baseadas em GAs para o OPF, são elas: GAP-OPF, FSGAP-OPF e MOGAP-OPF.

A primeira proposta, intitulada GAP-OPF, foi avaliada em 6 bases de dados do mundo real e comparada com o OPF e OPF-TPPA. Todos os métodos foram examinados de acordo com os três critérios: acurácia, taxa de poda e tempo de classificação. Com base nos experimentos, pode-se afirmar que o GAP-OPF é uma estratégia factível e pode ser aplicado em tarefas de reconhecimento de padrões, pois obteve acurácias superiores quando comparado apenas com o OPF-TPPA, e ainda assim, foi capaz de reduzir o tamanho do conjunto de treinamento e tempo de classificação para cerca de metade dos valores obtidos pelo OPF antes da poda. Por outro lado, o OPF-TPPA removeu mais amostras do conjunto de treinamento e, conseqüentemente, realizou a etapa de classificação mais

rápido que o GAP-OPF.

Já para a segunda proposta, FSGAP-OPF, sua capacidade de generalização foi testada em conjuntos de dados públicos do repositório UCI para fins de comparação com os resultados obtidos pelo OPF-TPPA, ROB-OPF e OPF sem poda. Com base nas simulações, pode-se dizer que o FSGAP-OPF é uma alternativa viável para tarefas de classificação, visto que superou o OPF, OPF-TPPA e ROB-OPF em termos de taxa de poda e tempo de classificação, sem comprometer ou mesmo melhorando os valores de acurácia. Outra vantagem interessante nessa proposta, é o fato dela produzir modelos de tamanho fixo, o que ajudar a resolver problemas em que o consumo de memória é crucial, ao estabelecer previamente um limite de tamanho máximo para representar o conjunto de treinamento podado.

Para o terceiro e último modelo proposto, MOGAP-OPF, seu desempenho foi comparado com o OPF-TPPA, ROB-OPF e o OPF sem poda, por aplicá-los em seis conjuntos de dados obtidos do repositório UCI. Todos os algoritmos foram investigados seguindo três critérios: acurácia, tempo de classificação e taxa de poda. De acordo com os resultados, pode-se afirmar que o MOGAP-OPF é uma boa estratégia para problemas de classificação, uma vez que obteve os melhores resultados, considerando todas as métricas simultaneamente, quando confrontado com todas as outras técnicas.

Os modelos propostos nesta dissertação seguem uma ordem cronológica de evolução. Note que na primeira proposta, empregada pelo GAP-OPF, a única métrica capaz de apresentar equivalência quando comparada com os demais métodos foi a acurácia, enquanto que na segunda abordagem, elaborada pelo FSGAP-OPF, a poda e o tempo de classificação alcançaram os melhores resultados. Para finalizar, surge o MOGAP-OPF, um modelo capaz de vencer todos os algoritmos ao avaliar todas as métricas simultaneamente.

6.2 Contribuições deste trabalho

As contribuições apresentadas nesta dissertação estão relacionadas aos temas: floresta de caminhos ótimos, algoritmos de aprendizagem de máquina, métodos de poda no classificador OPF e algoritmos genéticos.

A Seção 2 apresenta o arcabouço necessário para a compreensão do algoritmo de estudo nesta dissertação, iniciando com uma breve introdução seguido de uma explicação nos procedimentos de treinamento e classificação do modelo utilizado: **o classificador de padrões supervisionado OPF baseado em grafo completo**. Além disso, é apresentada uma outra variação do modelo OPF baseado em um grafo k -NN, onde posteriormente poderá ser explorado para fins de pesquisa. No Capítulo 3, é revelada a meta-heurística utilizada para trabalhar juntamente com o classificador OPF, chamada algoritmos genéticos. A princípio, é contada a história inspirada na teoria da evolução natural que deu o

seu surgimento, bem como as características fundamentais no processo de otimização.

No Capítulo 4, é dado um aprofundamento na explicação de como funcionam as técnicas de poda para o classificador OPF, iniciando pelas adquiridas no estado da arte, e posteriormente os modelos criados nesta dissertação. Finalmente, na Seção 5, são revelados os resultados alcançados pelos classificadores. Desse modo, assegurando um bom embasamento na qualidade das técnicas propostas.

6.3 Trabalhos futuros

Como trabalhos futuros, almeja-se investigar uma nova modelagem para representar os indivíduos do GA, de modo que a solução possa ser simbolizada com uma quantidade menor de genes. Dessa forma, acredita-se que o esforço computacional solicitado para executar o algoritmo de otimização reduza consideravelmente. Além disso, ainda considerando essa nova codificação aperfeiçoada, grandes bases de dados que exigem mais memória poderiam se beneficiar ao serem exploradas com maior eficiência.

Outra indagação que pretende-se apurar nos dados, está em realizar um estudo direcionado nos protótipos, para averiguar o nível de impacto dos mesmos na performance do classificador, pois acredita-se que a ausência de um protótipo relevante pode ocasionar uma redução drástica no desempenho do classificador, assim como a permanência de um protótipo ruidoso pode defasar a qualidade do modelo.

Para as próximas versões baseadas em GAs, almeja-se explorar mais o conhecimento *a priori* do problema, para inicializar a população de indivíduos com soluções já melhoradas. Para tal efeito, será utilizada alguma meta-heurística capaz de realizar algum pré-ajuste nos parâmetros do GA, de tal modo que possam ser geradas soluções diversificadas no espaço de busca. Desse modo, pressupõe-se que a descoberta do melhor indivíduo seja alcançada mais rapidamente.

Outras meta-heurísticas de otimização, tais como recozimento simulado, colônia de formigas, dentre outras, deverão ser exploradas no intuito de desvendar qual técnica melhor se enquadra para um problema de natureza específico. Ainda nesse contexto, busca-se aproveitar as vantagens de cada uma dessas técnicas para poder combiná-las, com o propósito de construir algum método de codificação genérico para qualquer meta-heurística baseada em população. Por fim, deseja-se iniciar os estudos nas outras derivações do algoritmo OPF, sendo a versão não supervisionada e a supervisionada baseada em grafo k -NN, para que em um futuro próximo, novas proposições sejam difundidas no meio acadêmico.

Referências

- ALBAYRAK, M.; ALLAHVERDI, N. Development a new mutation operator to solve the traveling salesman problem by aid of genetic algorithms. *Expert Systems with Applications*, Elsevier, v. 38, n. 3, p. 1313–1320, 2011. Citado na página 54.
- ALENCAR, A. S.; ROCHA NETO, A. R. d.; GOMES, J. P. P. A new pruning method for extreme learning machines via genetic algorithms. *Applied Soft Computing*, Elsevier, v. 44, p. 101–107, 2016. Citado na página 31.
- ALICKOVIC, E.; SUBASI, A. Medical decision support system for diagnosis of heart arrhythmia using dwf and random forests classifier. *Journal of medical systems*, Springer, v. 40, n. 4, p. 108, 2016. Citado na página 29.
- ALLÈNE, C. et al. Some links between min-cuts, optimal spanning forests and watersheds. *Mathematical Morphology and its Applications to Image and Signal Processing*, p. 253–264, 2007. Citado na página 37.
- ALVES, S. S.; OLIVEIRA, S. A.; Rocha Neto, A. R. d. A novel educational timetabling solution through recursive genetic algorithms. In: IEEE. *2015 Latin America Congress on Computational Intelligence (LA-CCI)*. Curitiba, Brazil, 2015. p. 1–6. Citado na página 53.
- BACK, T. *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*. 198 Madison Ave. New York, NY: Oxford university press, 1996. Citado na página 43.
- BARBOSA, G. S.; COSTA, L. da S.; ROCHA NETO, A. R. d. A new genetic algorithm-based pruning approach for optimum-path forest. In: IEEE. *2018 7th Brazilian Conference on Intelligent Systems (BRACIS)*. Sao Paulo, Brazil, 2018. p. 13–18. Citado na página 63.
- BHOWMICK, A.; HAZARIKA, S. M. E-mail spam filtering: A review of techniques and trends. In: KALAM, A.; DAS, S.; SHARMA, K. (Ed.). *Advances in Electronics, Communication and Computing*. Singapore: Springer Singapore, 2018. p. 583–590. ISBN 978-981-10-4765-7. Citado na página 29.
- BRANKE, J. et al. *Multiobjective optimization: Interactive and evolutionary approaches*. Berlin Heidelberg: Springer Science & Business Media, 2008. v. 5252. Citado na página 56.
- CARDOSO, F. A. C. M. et al. Algoritmos geneticos para interpolação não-linear de imagem e decodificação de codigos lineares. Brazil, 1998. Citado na página 44.
- CASTRO, R. E. de. *Otimização de estruturas com multi-objetivos via algoritmos genéticos*. Tese (Doutorado) — Tese de Doutorado, Universidade Federal do Rio de Janeiro, Rio de Janeiro . . . , 2001. Citado na página 57.
- CHEN, J.; WANG, X.; ZHAI, J. Pruning decision tree using genetic algorithms. In: IEEE. *Artificial Intelligence and Computational Intelligence, 2009. AICI'09. International*

- Conference on*. Shanghai, China, 2009. v. 3, p. 244–248. Citado 2 vezes nas páginas 31 e 63.
- CORMAN, T. H. et al. *Introduction to algorithms*. USA: MIT press Cambridge, MA, 1990. Citado na página 40.
- CORMEN, T. H. et al. *Introduction to algorithms mit press. Cambridge MA*, 1990. Citado na página 37.
- COSTA, L. da S.; BARBOSA, G. S.; ROCHA NETO, A. R. d. A fixed-size pruning approach for optimum-path forest. In: SPRINGER. *International Work-Conference on Artificial Neural Networks*. Cham, 2019. p. 723–734. Citado na página 66.
- DAVIS, L. *Handbook of genetic algorithms*. CUMINCAD, 1991. Citado na página 47.
- DOMINGOS, P. *The master algorithm: How the quest for the ultimate learning machine will remake our world*. EUA: Basic Books, 2015. Citado na página 29.
- DRÉO, J. et al. *Metaheuristics for hard optimization: methods and case studies*. Berlin Heidelberg: Springer Science & Business Media, 2006. Citado 3 vezes nas páginas 50, 51 e 54.
- DUBOIS-LACOSTE, J.; LÓPEZ-IBÁÑEZ, M.; STÜTZLE, T. Combining two search paradigms for multi-objective optimization: Two-phase and pareto local search. In: TALBI, E.-G. (Ed.). *Hybrid Metaheuristics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013. p. 97–117. ISBN 978-3-642-30671-6. Disponível em: <https://doi.org/10.1007/978-3-642-30671-6_3>. Citado na página 70.
- EIBEN, A. E.; SMITH, J. E. et al. *Introduction to evolutionary computing*. Berlin, Heidelberg: Springer, 2003. v. 53. Citado na página 67.
- FALCÃO, A. X.; STOLFI, J.; LOTUFO, R. de A. The image foresting transform: Theory, algorithms, and applications. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 26, n. 1, p. 19–29, 2004. Citado 2 vezes nas páginas 29 e 37.
- FILHO, P. P. R. et al. Automated recognition of lung diseases in ct images based on the optimum-path forest classifier. *Neural Computing and Applications*, Springer, v. 31, n. 2, p. 901–914, 2019. Citado na página 30.
- FOGEL, L. J.; OWENS, A. J.; WALSH, M. J. *Artificial intelligence through simulated evolution*. John Wiley, 1966. Citado na página 43.
- FREITAS, F. G. de et al. Otimização em teste de software com aplicação de metaheurísticas. *Revista de Sistemas de*, v. 5, p. 3–13, 2010. Citado na página 56.
- GOLBERG, D. E. *Genetic algorithms in search, optimization, and machine learning*. Addison wesley, v. 1989, n. 102, p. 36, 1989. Citado 3 vezes nas páginas 45, 51 e 54.
- GRAVES, A.; MOHAMED, A.-r.; HINTON, G. Speech recognition with deep recurrent neural networks. In: IEEE. *2013 IEEE international conference on acoustics, speech and signal processing*. Vancouver, BC, Canada, 2013. p. 6645–6649. Citado na página 29.
- HASSANAT, A. B. et al. Solving the problem of the k parameter in the knn classifier using an ensemble learning approach. *arXiv preprint arXiv:1409.0919*, 2014. Citado na página 78.

- HEATON, J. *Introduction to Neural Networks with Java*. Heaton Research, 2008. ISBN 9781604390087. Disponível em: <<https://books.google.com.br/books?id=Swlcw7M4uD8C>>. Citado na página 78.
- HOLLAND, J. H. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. MIT Press, 1992. (Bradford book). ISBN 9780585038445. Disponível em: <<https://books.google.com.br/books?id=cyV7nQEACAAJ>>. Citado 3 vezes nas páginas 29, 43 e 46.
- HOSSAIN, M. et al. Forecasting the weather of nevada: A deep learning approach. In: IEEE. *2015 international joint conference on neural networks (IJCNN)*. Killarney, Ireland, 2015. p. 1–6. Citado na página 29.
- HUANG, C.-L.; WANG, C.-J. A ga-based feature selection and parameters optimization for support vector machines. *Expert Systems with applications*, Elsevier, v. 31, n. 2, p. 231–240, 2006. Citado 2 vezes nas páginas 31 e 63.
- JAIN, A.; NANDAKUMAR, K.; ROSS, A. Score normalization in multimodal biometric systems. *Pattern recognition*, Elsevier, v. 38, n. 12, p. 2270–2285, 2005. Citado na página 75.
- JANIKOW, C. Z.; MICHALEWICZ, Z. An experimental comparison of binary and floating point representations in genetic algorithms. In: *ICGA*. Morgan Kaufmann, 1991. p. 31–36. Disponível em: <<http://dblp.uni-trier.de/db/conf/icga/icga1991.html#JanikowM91>>. Citado na página 47.
- JAYALAKSHMI, T.; SANTHAKUMARAN, A. Statistical normalization and back propagation for classification. *International Journal of Computer Theory and Engineering*, v. 3, n. 1, p. 1793–8201, 2011. Citado na página 75.
- KORBICZ, J. et al. *Fault diagnosis: models, artificial intelligence, applications*. Berlin Heidelberg: Springer Science & Business Media, 2012. Citado na página 29.
- KOZA, J. R. Genetic programming ii: Automatic discovery of reusable subprograms. *Cambridge, MA, USA*, v. 13, n. 8, p. 32, 1994. Citado na página 44.
- KOZA, J. R.; KOZA, J. R. *Genetic programming: on the programming of computers by means of natural selection*. London, England: MIT press, 1992. v. 1. Citado na página 44.
- LAM, H. et al. Tuning of the structure and parameters of neural network using an improved genetic algorithm. In: IEEE. *Industrial Electronics Society, 2001. IECON'01. The 27th Annual Conference of the IEEE*. Denver, CO, USA, USA, 2001. v. 1, p. 25–30. Citado 2 vezes nas páginas 31 e 63.
- LINDEN, R. *Algoritmos genéticos: uma importante ferramenta da inteligência computacional*. Brasport, 2006. ISBN 9788574522654. Disponível em: <<https://books.google.com.br/books?id=1oUPOgAACAAJ>>. Citado 2 vezes nas páginas 46 e 48.
- LU, H. et al. Brain intelligence: go beyond artificial intelligence. *Mobile Networks and Applications*, Springer, v. 23, n. 2, p. 368–375, 2018. Citado na página 29.

- LUZ, E. J. D. S. et al. Ecg arrhythmia classification based on optimum-path forest. *Expert Systems with Applications*, Elsevier, v. 40, n. 9, p. 3561–3573, 2013. Citado na página 30.
- MIRANDA, P. M. et al. Object delineation by-connected components. *EURASIP Journal on Advances in Signal Processing*, Springer, v. 2008, n. 1, p. 467928, 2008. Citado na página 37.
- MITCHELL, M. *An Introduction to Genetic Algorithms*. Bradford Books, 1998. (A Bradford book). ISBN 9780262631853. Disponível em: <<https://books.google.com.br/books?id=0eznlz0TF-IC>>. Citado 3 vezes nas páginas 44, 46 e 50.
- MONTOYA-ZEGARRA, J. A. et al. Rotation-invariant texture recognition. In: BEBIS, G. et al. (Ed.). *Advances in Visual Computing*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. p. 193–204. ISBN 978-3-540-76856-2. Citado na página 30.
- NAKAMURA, R. et al. Optimum-path forest pruning parameter estimation through harmony search. In: IEEE. *2011 24th SIBGRAPI Conference on Graphics, Patterns and Images*. Maceio, Alagoas, Brazil, 2011. p. 181–188. Citado 2 vezes nas páginas 30 e 59.
- NILSSON, N. J. *Principles of artificial intelligence*. EUA: Morgan Kaufmann, 2014. Citado na página 29.
- NUNES, T. M. et al. Eeg signal classification for epilepsy diagnosis via optimum path forest—a systematic assessment. *Neurocomputing*, Elsevier, v. 136, p. 103–123, 2014. Citado na página 30.
- O’HARE, G. M.; JENNINGS, N. R.; JENNINGS, N. *Foundations of distributed artificial intelligence*. 605 Third Ave. New York, NY United States: John Wiley & Sons, 1996. v. 9. Citado na página 29.
- OLIVEIRA, G. M. B. de. *Dinâmica e evolução de autômatos celulares unidimensionais*. Tese (Doutorado) — Instituto Tecnológico de Aeronáutica, São José dos Campos, São Paulo, Brazil, 1999. Citado na página 55.
- OLIVEIRA, S. A. F.; BEZERRA, F. N.; ROCHA NETO, A. R. d. Genetic seam carving: A genetic algorithm approach for content-aware image retargeting. In: PAREDES, R.; CARDOSO, J. S.; PARDO, X. M. (Ed.). *Pattern Recognition and Image Analysis*. Cham: Springer International Publishing, 2015. p. 700–707. ISBN 978-3-319-19390-8. Citado na página 44.
- PAPA, J. P.; FALCÃO, A. X. A new variant of the optimum-path forest classifier. In: BEBIS, G. et al. (Ed.). *Advances in Visual Computing*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. p. 935–944. ISBN 978-3-540-89639-5. Citado 2 vezes nas páginas 29 e 35.
- PAPA, J. P.; FALCÃO, A. X. On the training patterns pruning for optimum-path forest. In: FOGGIA, P.; SANSONE, C.; VENTO, M. (Ed.). *Image Analysis and Processing – ICIAP 2009*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. p. 259–268. ISBN 978-3-642-04146-4. Citado 3 vezes nas páginas 30, 59 e 95.
- PAPA, J. P. et al. Efficient supervised optimum-path forest classification for large datasets. *Pattern Recognition*, Elsevier, v. 45, n. 1, p. 512–520, 2012. Citado 3 vezes nas páginas 30, 35 e 59.

- PAPA, J. P. et al. Robust pruning of training patterns for optimum-path forest classification applied to satellite-based rainfall occurrence estimation. *IEEE Geoscience and Remote Sensing Letters*, IEEE, v. 7, n. 2, p. 396–400, 2010. Citado 4 vezes nas páginas 30, 59, 60 e 95.
- PAPA, J. P. et al. Design of robust pattern classifiers based on optimum-path forests. *Mathematical Morphology and its Applications to Signal and Image Processing (ISMM)*, p. 337–348, 2007. Citado na página 30.
- PAPA, J. P.; FALCAO, A. X.; SUZUKI, C. T. Supervised pattern classification based on optimum-path forest. *International Journal of Imaging Systems and Technology*, Wiley Online Library, v. 19, n. 2, p. 120–131, 2009. Citado 5 vezes nas páginas 29, 30, 35, 76 e 95.
- PAPA, J. P. et al. A discrete approach for supervised pattern recognition. In: BRIMKOV, V. E.; BARNEVA, R. P.; HAUPTMAN, H. A. (Ed.). *Combinatorial Image Analysis*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. p. 136–147. ISBN 978-3-540-78275-9. Citado na página 30.
- PAPA, J. P. et al. Classificação supervisionada de padrões utilizando floresta de caminhos otimos. Brazil, 2008. Citado 2 vezes nas páginas 29 e 35.
- PEARL, J. Heuristics: intelligent search strategies for computer problem solving. Addison-Wesley Pub. Co., Inc., Reading, MA, 1984. Citado na página 29.
- PEREIRA, C. R. et al. An optimum-path forest framework for intrusion detection in computer networks. *Engineering Applications of Artificial Intelligence*, Elsevier, v. 25, n. 6, p. 1226–1234, 2012. Citado na página 30.
- RAHNAMAYAN, S.; TIZHOOSH, H. R.; SALAMA, M. M. A novel population initialization method for accelerating evolutionary algorithms. *Computers & Mathematics with Applications*, Elsevier, v. 53, n. 10, p. 1605–1614, 2007. Citado na página 47.
- RAMESH, A. et al. Artificial intelligence in medicine. *Annals of The Royal College of Surgeons of England*, Royal College of Surgeons of England, v. 86, n. 5, p. 334, 2004. Citado na página 29.
- RAZALI, N. M.; GERAGHTY, J. et al. Genetic algorithm performance with different selection strategies in solving tsp. In: INTERNATIONAL ASSOCIATION OF ENGINEERS HONG KONG. *Proceedings of the world congress on engineering*. London, U.K, 2011. v. 2, n. 1, p. 1–6. Citado na página 49.
- RECHENBERG, I. Cybernetic solution path of an experimental problem. *Royal Aircraft Establishment Library Translation 1122*, 1965. Citado na página 43.
- RECHENBERG, I. Evolution strategy: Optimization of technical systems by means of biological evolution. *Fromman-Holzboog, Stuttgart*, v. 104, p. 15–16, 1973. Citado na página 43.
- RODRIGUES, D. et al. A wrapper approach for feature selection based on bat algorithm and optimum-path forest. *Expert Systems with Applications*, Elsevier, v. 41, n. 5, p. 2250–2258, 2014. Citado na página 30.

- RODRIGUES, D.; SOUZA, A. N.; PAPA, J. P. Pruning optimum-path forest classifiers using multi-objective optimization. In: IEEE. *2017 30th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. Rio de Janeiro, Brazil, 2017. p. 127–133. Citado na página 59.
- ROZENBERG, G.; BÄCK, T.; KOK, J. N. *Handbook of natural computing*. Berlin Heidelberg: Springer, 2012. Citado na página 44.
- RUSSELL, S.; NORVIG, P. *Artificial Intelligence: A Modern Approach*. Pearson, 2016. (Always learning). ISBN 9781292153964. Disponível em: <<https://books.google.com.br/books?id=XS9CjwEACAAJ>>. Citado na página 29.
- SILVA, D. A.; ROCHA NETO, A. R. d. A genetic algorithms-based lssvm classifier for fixed-size set of support vectors. In: ROJAS, I.; JOYA, G.; CATALA, A. (Ed.). *Advances in Computational Intelligence*. Cham: Springer International Publishing, 2015. p. 127–141. ISBN 978-3-319-19222-2. Citado 3 vezes nas páginas 31, 63 e 68.
- SILVA, S. F. d. *Seleção de características por meio de algoritmos genéticos para aprimoramento de rankings e de modelos de classificação*. Tese (Doutorado) — Universidade de São Paulo, 2011. Citado 5 vezes nas páginas 45, 46, 47, 51 e 52.
- SIVANANDAM, S.; DEEPA, S. *Introduction to genetic algorithms*. Berlin Heidelberg: Springer Science & Business Media, 2007. Citado 2 vezes nas páginas 46 e 47.
- SIVAPATHASEKARAN, C. et al. Artificial neural network modeling and genetic algorithm based medium optimization for the improved production of marine biosurfactant. *Bioresource technology*, Elsevier, v. 101, n. 8, p. 2884–2887, 2010. Citado na página 44.
- SRINIVAS, M.; PATNAIK, L. M. Genetic algorithms: A survey. *computer*, IEEE, v. 27, n. 6, p. 17–26, 1994. Citado na página 63.
- TANG, K.-S. et al. Genetic algorithms and their applications. *IEEE signal processing magazine*, IEEE, v. 13, n. 6, p. 22–37, 1996. Citado na página 46.
- THEODORIDIS, S. *Koutroumbas K (2008) Pattern recognition*. MA, USA: Academic Press, 2008. Citado na página 29.
- THIERENS, D.; GOLDBERG, D. Convergence models of genetic algorithm selection schemes. In: DAVIDOR, Y.; SCHWEFEL, H.-P.; MÄNNER, R. (Ed.). *Parallel Problem Solving from Nature — PPSN III*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994. p. 119–129. ISBN 978-3-540-49001-2. Citado na página 49.
- WHITLEY, D.; BOGART, C. The evolution of connectivity: Pruning neural networks using genetic algorithm. In: *Proceedings of IJCNN-90*. San Diego, CA, USA: [s.n.], 1990. p. 134–137. Citado 2 vezes nas páginas 31 e 63.
- YANG, X.-S. *Nature-inspired optimization algorithms*. 32 Jamestown Road, London NW1 7BY: Elsevier, 2014. Citado na página 50.

Apêndices

APÊNDICE A – Evolução do GA para o MOGAP-OPF

Este Apêndice tem por objetivo apresentar a evolução do Algoritmo Genético (GA) ao longo do tempo, quando aplicado para 150 gerações. Foi utilizada a terceira proposta, MOGAP-OPF, com o valor de $\alpha = 0,8$ para avaliar o *fitness* do GA.

A.1 Resultados alcançados

As Figuras 21 e 22 apresentam os resultados para a evolução do GA ao longo de 150 gerações. O eixo das abscissas representa a quantidade de gerações, enquanto que o eixo das ordenadas indica a melhor aptidão e média de cada população.

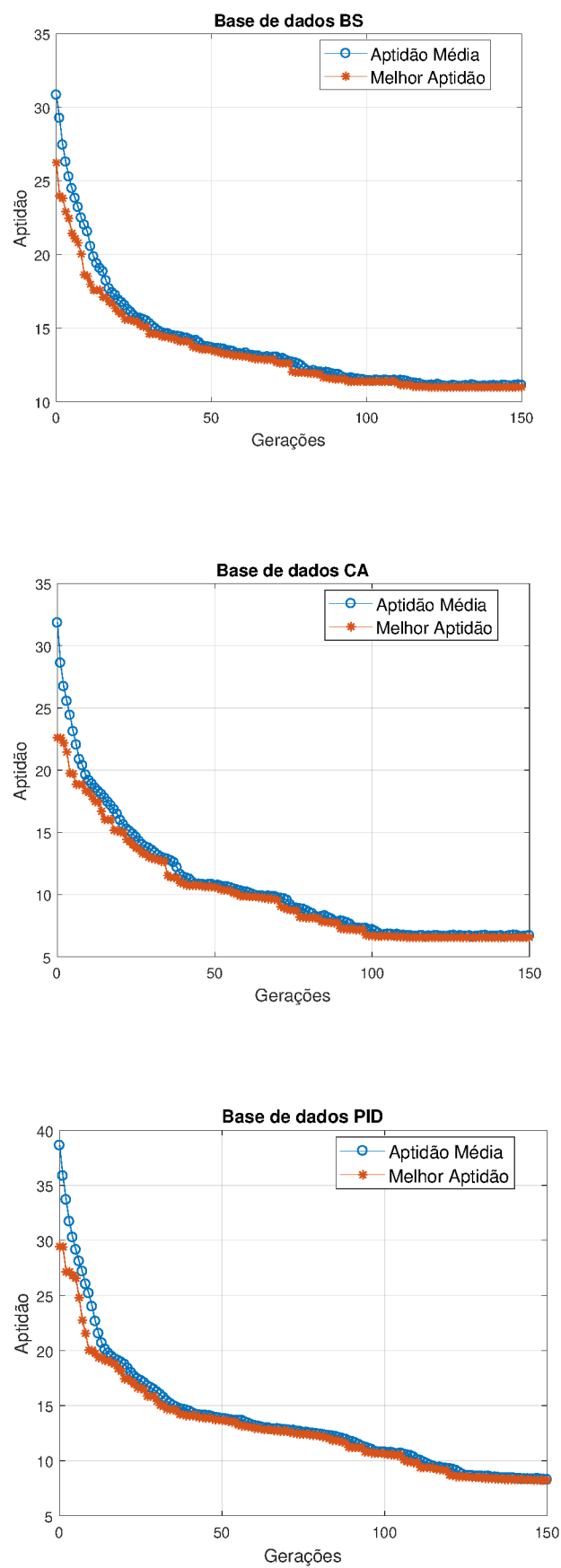
A aptidão de cada indivíduo é dada em função do erro, portanto quanto menor é o valor do *fitness* para a solução candidata, melhor é a qualidade da solução para o problema.

Observe na Figura 21, que a solução objetivo convergiu somente após executadas 100 gerações, enquanto que na Figura 22, houve estagnação na melhoria do *fitness* de cada população antes mesmo de alcançar 100 gerações.

O poder da capacidade de generalização do modelo MOGAP-OPF para a etapa de aprendizagem, pode ser certificado ao analisar a taxa de erro na etapa de validação, visto que essa taxa possui valores similares aos obtidos no estágio de teste, o que vem a indicar falta de *overfitting* no modelo gerado. Por outro lado, a diferença entre a media dos *fitness* e o melhor indivíduo por população, contada a partir da geração 50, não apresentou valores significantes, o que pode vir a indicar possível falta de diversidade nas soluções.

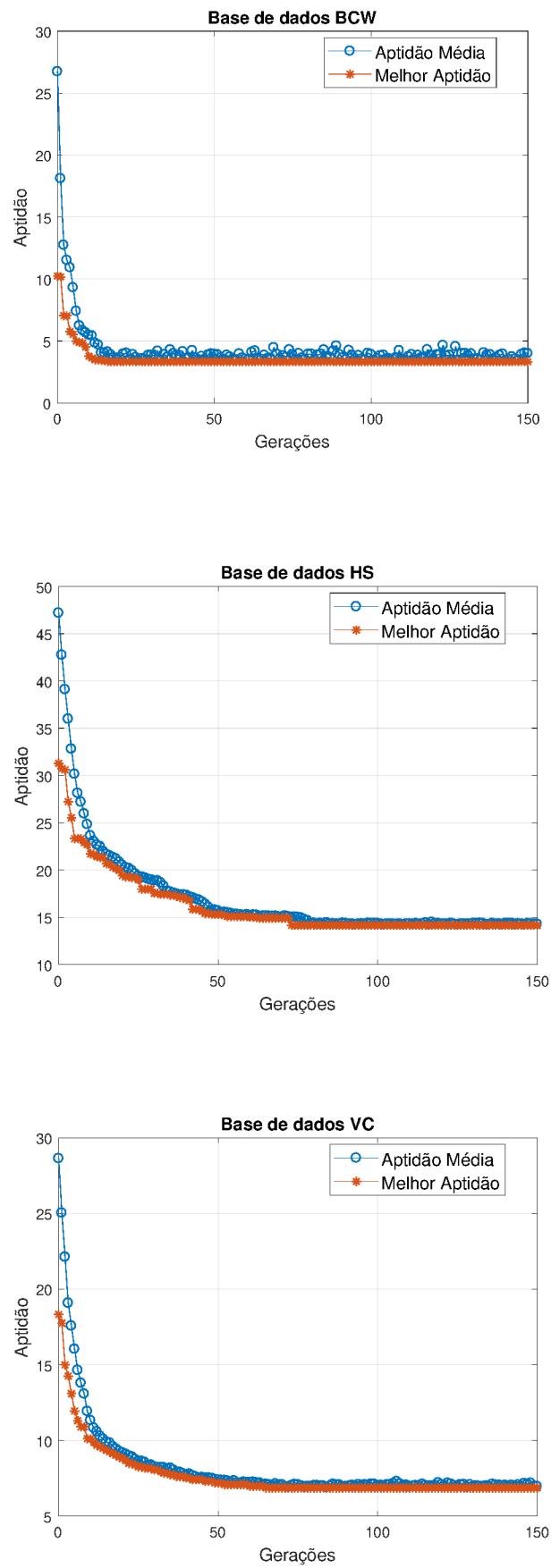
Os valores mostrados por cada gráfico nas Figuras 21 e 22, são resultados da aplicação de uma realização para cada base de dados utilizando o valor de $\alpha = 0,8$ da técnica MOGAP-OPF.

Figura 21 – Convergência da solução objetivo do GA para 150 gerações.



Fonte: Próprio autor.

Figura 22 – Convergência da solução objetivo do GA para 150 gerações.



Fonte: Próprio autor.