Tuning of Control Parameters of Grey Wolf Optimizer using Fuzzy Inference

A. Ferrari, G. Leandro, L. Coelho, C. da Silva, *Member, IEEE*, E. Lima, and C. Chaves

***Abstract*—The Grey Wolf Optimizer (GWO) is a recent metaheuristic that can be explored in many applications. This paper proposes a mechanism to tune the control parameters that influence the hunting process in the GWO in order to improve its efficiency. This adjustment is made by a fuzzy inference system that uses the normalized fitness value of each wolf and the hunting mechanism control parameters of the GWO. The proposed fuzzy mechanism is tested and compared with the conventional GWO and another version that uses a fuzzy system as input information the ratio of the current iteration number and the maximum number of iterations. For performance analysis of the proposed fuzzy mechanism, all tested optimizers in ten benchmark optimization functions ran 1000 times. Simulation results show that the proposed fuzzy mechanism improves the convergence of the conventional GWO and it is competitive in relation to another fuzzy version adopted in the GWO design.**

***Index Terms*—Metaheuristics, Optimization, Grey wolf optimizer, Benchmark functions, Fuzzy system.**

I. INTRODUÇÃO

AS metaheurísticas são, muitas vezes, utilizadas na resolução de problemas de otimização que são considerados complexos para as técnicas de otimização clássicas, tais como métodos baseados em informação do gradiente e abordagens de programação matemática. As metaheurísticas são geralmente inspiradas no comportamento de algum processo natural que pode ser representado por um comportamento biológico ou físico, ou mesmo por um processo químico [1], [2].

Um grupo de metaheurísticas que se destaca é a "inteligência de enxame". Os algoritmos que fazem parte deste grupo são caracterizados pela inspiração no comportamento natural de alguns tipos de animais que vivem na sociedade de forma cooperativa [3]. Uma das metaheurísticas de inteligência de enxame propostas recetemente é o GWO proposto em Mirjalili *et al*. [4]. Este algoritmo baseia-se no comportamento natural de liderança na hierarquia e mecanismos de caça dos lobos cinzentos. Em Mirjalili *et al*. [4], o algoritmo GWO quando é comparado com outras metaheurísticas bem conhecidas, o mesmo mostrou-se competitivo na otimização de funções testes unimodais, multimodais e compostas da competição IEEE CEC2005 (*Institute of Electrical and Electronic Engineers, Congress on Evolutionary Computation* 2005) [5]. O algoritmo GWO mostrou desempenho promissor na solução de problemas reais da engenharia que possuem espaço de busca desconhecido [4].

Nos trabalhos [6]–[10], encontram-se várias aplicações do algoritmo GWO em problemas de otimização e modificações do mesmo que visam melhorar a sua convergência. Por exemplo, foi introduzido a teoria do caos no algoritmo GWO com a finalidade de acelerar a velocidade de convergência [6]. Em [7], foi proposto uma nova versão do GWO voltada a otimização de problemas com a presença de múltiplos objetivos. Também pode ser encontrado na literatura, o GWO utilizando sistemas *fuzzy* para adaptação dos seus parâmetros de controle, estes cruciais na convergência [8]. Em outra aplicação do algoritmo GWO, foi apresentado uma nova variante do mesmo que utiliza os princípios de aprendizado por reforço no para o treinamento de uma rede neural artificial [9]. Em Jain *et al*. [10], o GWO é adotado para otimização de um controlador do tipo PID (Proporcional-Integral-Derivativo) aplicado em um sistema bola e aro.

Nos trabalhos citados anteriormente, percebe-se que o GWO, por ser uma metaheurística relativamente recente, pode ser ainda mais explorada para diversos tipos de aplicações de otimização, ou ainda, modificado e combinado com outras metodologias visando melhorar a sua aplicabilidade e eficiência. É importante mencionar o teorema *"no free lunch"* (NFL), pois não existe um tipo específico de metaheurística que seja superior na resolução de todos os problemas de otimização [11]. Em problemas de otimização em que a utilização do GWO se mostra adequada, a proposta deste artigo pode melhorar a eficiência do mesmo.

Este artigo tem como contribuição a proposição de um mecanismo de ajuste nos parâmetros de controle do processo de caça do GWO, por meio de sistema *fuzzy*. O modelo *fuzzy* proposto utiliza o valor do *fitness* normalizado de cada lobo (solução candidata) e os valores atuais dos parâmetros chaves do mecanismo de caça do GWO como informação de sinal de entrada. Logo, a metodologia proposta faz o ajuste dos parâmetros de controle do GWO sem depender do valor da iteração corrente da simulação, ao contrário do que acontece na versão convencional do algoritmo que faz o ajuste dos seus parâmetros de forma linear. O mecanismo proposto visa melhorar a convergência do GWO de forma que o algoritmo obtenha os valores ótimos que se aproximam melhor dos mínimos globais das funções testes utilizadas neste artigo.

O restante deste artigo está estruturado em três partes incluindo métodos, resultados e conclusão. Nos métodos são apresentados os fundamentos do GWO e o sistema *fuzzy*

A. C. K. Ferrari, Centro Universitário FACEAR (UNIFACEAR), Araucária, Paraná, Brasil, allanckferrari87@gmail.com.

G. V. Leandro, Universidade Federal do Paraná (UFPR), Curitiba, Paraná, Brasil, gede@eletrica.ufpr.br. L. S. Coelho, Pontifícia Universidade Católica do Paraná, Curitiba, Paraná, Brasil, leandro.coelho@pucpr.br.

C. A. G. da Silva, Universidade Federal do Paraná (UFPR), Curitiba, Paraná, Brasil, carlos.gouvea@ieee.org.

E. G. Lima, Centro Universitário FACEAR (UNIFACEAR), Araucária, Paraná, Brasil, edmilsonprofessor2018@gmail.com. C. R. Chaves, Universidade do Contestado Canoinhas (UNC), Canoinhas, Santa Catarina, Brasil, cr.chaves@superig.com.br.

proposto. Os resultados, são subdivididos em quatro seções. A primeira seção apresenta um sistema *fuzzy* que ajusta os parâmetros do GWO em função do número da iteração corrente, esta metodologia é utilizada na comparação com a proposta deste artigo. A segunda seção apresenta as funções testes utilizadas no processo de otimização dos algoritmos. A terceira seção mostra os critérios de análise e comparação adotados. Na quarta seção são mostrados os resultados obtidos. Finalmente na conclusão são apresentados as restrições e os desdobramentos para trabalhos futuros.

II. MÉTODOS

Nesta seção são apresentados os fundamentos básicos do algoritmo GWO e a proposta deste artigo que visa melhorar a sua convergência.

A. Fundamentos do Algoritmo GWO

O algoritmo GWO é inspirado na vida selvagem dos lobos cinzentos que são predadores naturais e estão no topo da cadeia alimentar. Os lobos cinzentos preferem viver em grupos onde existe uma hierarquia social dominante, conforme ilustrado na Fig. 1. O lobo dominante é denominado de lobo alfa (α) sendo responsável por liderar o grupo e no algoritmo representa a melhor solução. Na hierarquia, depois do lobo α vem o lobo beta (β) e lobo delta (δ) que representam respectivamente a segunda e terceira melhor solução. Estes lobos possuem a função de auxiliar o lobo α. A classe mais baixa do grupo é representada pelos lobos omega (ω) que se submetem aos lobos α, β e δ.

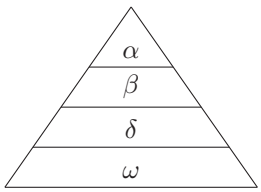

Fig. 1. Hierarquia do lobo cinzento.

A estrutura deste algoritmo baseia-se no mecanismo de caça dos lobos cinzentos. Os lobos α, β e δ lideram a caçada de uma presa, enquanto que os lobos ω seguem os três lobos na pesquisa de um ótimo global. Durante o processo de caça, os lobos cercam a presa até que ela pare de se mover. Este comportamento pode ser representado pelas seguintes equações:

$$\vec{D} = |\vec{C}\vec{X}_p(t) - \vec{X}(t)|, \tag{1}$$

$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A}\vec{D} \tag{2}$$

onde t indica a iteração atual, $\vec{A}$ e $\vec{C}$ são os vetores coeficientes, $\vec{X}_p(t)$ representa o vetor posição do melhor lobo, o vetor $\vec{D}$ representa a distância do movimento aleatório do melhor lobo em relação ao lobo ω na iteração atual, $\vec{X}(t)$ e $\vec{X}(t+1)$ indicam respectivamente a posição de um lobo ω na iteração atual e um passo à frente.

Os vetores $\vec{A}$ e $\vec{C}$ são calculados por:

$$\vec{A} = 2\vec{a}\vec{r}_1 - \vec{a}, \tag{3}$$

$$\vec{C} = 2\vec{r}_2, \tag{4}$$

onde o coeficiente $\vec{a}$ decresce linearmente a cada iteração de 2 até 0 e os vetores $\vec{r}_1$ e $\vec{r}_2$ são valores gerados aleatoriamente com distribuição uniforme no intervalo de 0 a 1.

Segundo Mirjalili *et al*. [4], o vetor $\vec{A}$ representa o comportamento de exploração territorial do lobo cinzento e decresce a cada iteração por estar em função do coeficiente $\vec{a}$. Quando $|\vec{A}| > 1$, os lobos estão divergindo e explorando o espaço de busca para encontrar uma presa em potencial para convergir e atacar. Quando $|\vec{A}| < 1$, os lobos estão convergindo e atacando uma presa. O vetor $\vec{C}$ pode ser considerado como o efeito dos obstáculos quando os lobos estão perseguindo sua presa. Ao contrário do vetor $|\vec{A}|$, o vetor $|\vec{C}|$ age de forma aleatória e isto pode facilitar ou dificultar a captura da presa.

Uma das características dos lobos cinzentos é a capacidade de reconhecer a localização de uma presa potencial, em seguida cercar e depois atacar. A posição dos lobos ω é atualizada em função da localização dos lobos α, β e δ que lideram a caçada, pois possuem melhor conhecimento sobre a localização de uma possível presa. As equações que representam este comportamento são as seguintes:

$$\vec{D}_\alpha = |\vec{C}_\alpha\vec{X}_\alpha(t) - \vec{X}(t)|, \tag{5}$$

$$\vec{D}_\beta = |\vec{C}_\beta\vec{X}_\beta(t) - \vec{X}(t)|, \tag{6}$$

$$\vec{D}_\delta = |\vec{C}_\delta\vec{X}_\delta(t) - \vec{X}(t)|, \tag{7}$$

$$\vec{X}_1 = \vec{X}_\alpha(t) - \vec{A}_\alpha\vec{D}_\alpha, \tag{8}$$

$$\vec{X}_2 = \vec{X}_\beta(t) - \vec{A}_\beta\vec{D}_\beta, \tag{9}$$

$$\vec{X}_3 = \vec{X}_\delta(t) - \vec{A}_\delta\vec{D}_\delta, \tag{10}$$

$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \tag{11}$$

As equações de 5, 6, 7, 8, 9 e 10 representam a implementação do mecanismo de caça dos lobos cinzentos no GWO. Estas equações são uma expansão das equações 1 a 4, e consideram a influência individual dos lobos α, β e δ.

A equação 11 atualiza a posição de todos os lobos ω para a próxima iteração sendo basicamente uma média dos valores $\vec{X}_1$, $\vec{X}_2$ e $\vec{X}_3$ que representam respectivamente a influência individual dos lobos α, β e δ. Os lobos ω tendem a se mover para o centro dos três melhores lobos. No final de cada iteração a hierarquia do grupo é atualizado. O GWO pode ser representado pelo pseudocódigo do Algoritmo 1.

Algoritmo 1: Pseudocódigo do algoritmo GWO

início
- Inicia a população aleatoriamente;
- Inicia $\vec{a}$, $\vec{A}$ e $\vec{C}$;
- Calcula o *fitness* de cada agente (lobo);
- $\vec{X}_\alpha$ melhor agente da população;
- $\vec{X}_\beta$ segundo melhor agente da população;
- $\vec{X}_\delta$ terceiro melhor agente da população;
- **while** *t ¡ Número_máximo_de_iterações* **do**
  - **para** *cada agente da população* **faça**
    - Atualiza a posição atual de cada agente pela equação 11;
  - **fim**
  - Atualiza o valor de $\vec{a}$, $\vec{A}$ e $\vec{C}$;
  - Calcula o *fitness* de todos os agentes;
  - Atualiza o valor de α, β e δ;
  - $t = t + 1$;
- **end**
- **retorna** $\vec{X}_\alpha$

fim

B. Proposta de Ajuste de Parâmetros por Lógica Fuzzy

A proposta deste artigo consiste em um sistema *fuzzy* para o ajuste simultâneo dos parâmetros $\vec{a}$ e $\vec{C}$ do GWO. Este modelo *fuzzy* é baseado na técnica proposta de Nafar *et al*. [12] que faz a adaptação do parâmetro de peso de inércia do algoritmo PSO (*Particle Swarm Optimization*). Este mesmo mecanismo de adaptação foi também aplicado no algoritmo híbrido que combina as metaheurísticas enxame de partículas PSO [13] e o algoritmo de busca gravitacional (*Gravitacional Search Algorithm*, GSA) [14], sendo denominado de PSOGSA (*Particle Swarm Optimization Gravitacional Search Algorithm*) no procedimento de identificação de um sistema multivariável [15].

Neste trabalho, o GWO é modificado de forma que cada lobo (indivíduo da população) está associado diretamente a um par próprio de valores $\vec{a}$ e $\vec{C}$ que são iniciados respectivamente com valores 2 e 0. Estes valores são atualizados durante o processo de otimização das funções testes.

O modelo *fuzzy* proposto recebe os valores atuais dos parâmetros $\vec{a}$ e $\vec{C}$, e também o valor referente ao desempenho mensurado do *fitness* normalizado NFV (*Normalized Fitness Value*) de cada lobo da população. A saída é representada pelas variações dos parâmetros essenciais do GWO, denominados de $\Delta\vec{a}$ e $\Delta\vec{C}$. A partir do *fitness* de cada lobo, o valor do respectivo NFV é calculado por:

$$\text{NFV} = \frac{\textit{fitness} - \textit{fitness}_{min}}{\textit{fitness}_{max} - \textit{fitness}_{min}} \tag{12}$$

O valor de NFV pertence ao intervalo [0,1]. O problema de otimização deste artigo é do tipo minimização, onde o *fitness* de cada lobo é obtido diretamente pelo valor ótimo das funções testes.

As equações que atualizam os parâmetros $\vec{a}$ e $\vec{C}$ de cada lobo são as seguintes:

$$\vec{a}^{t+1} = \vec{a}^{t} + \Delta\vec{a} \tag{13}$$

$$\vec{C}^{t+1} = \vec{C}^{t} + \Delta\vec{C} \tag{14}$$

O sistema *fuzzy* é responsável por atualizar os parâmetros $\vec{a}$ e $\vec{C}$ de cada indivíduo da população e o mesmo recebe três entradas: o valor atual dos parâmetros $\vec{a}$ e $\vec{C}$, e o valor do *fitness* normalizado NFV. Inicialmente, estes valores são "fuzzificados" por suas respectivas funções de pertinência e em seguida são obtidos os seus respectivos valores linguísticos com grau de pertinência μ. Estes valores são aplicados a um conjunto de regras e em seguida determinam-se os valores semânticos de $\Delta\vec{a}$, $\Delta\vec{C}$ e seus respectivos graus de pertinências. Depois de determinado estes valores, é aplicado o processo de "defuzzificação" para estimar os valores numéricos de $\Delta\vec{a}$ e $\Delta\vec{C}$. Finalmente, estes valores são aplicados nas equações 13 e 14 para a atualização dos parâmetros $\Delta\vec{a}$ e $\Delta\vec{C}$ de cada lobo.

O sistema *fuzzy* deste trabalho utiliza o modelo linguístico de Mamdani com funções de pertinência em formato triangular, operador de agregação de antecedentes mínimo, operador de agregação de regras máximo e "defuzzificação" por centro do máximo. Este sistema está representado na Fig. 2.

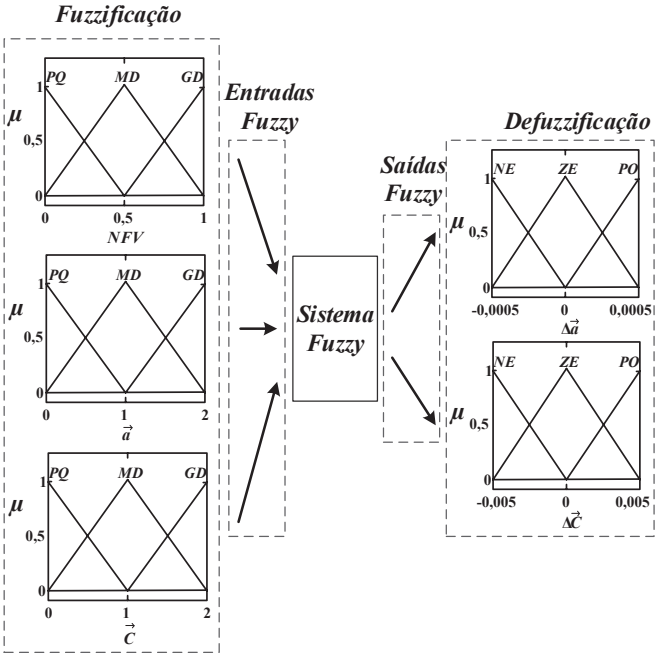

Fig. 2. Sistema *fuzzy* proposto.

Os termos linguísticos das funções de pertinências, utilizados para descrever os valores semânticos das variáveis de entrada (NFV, $\vec{a}$ e $\vec{C}$) são declarados como: PQ (Pequeno); MD (Médio) e GD (Grande). Os valores semânticos das variáveis de saída ($\Delta\vec{a}$ e $\Delta\vec{C}$) são declarados como: NE (Negativo); ZE (Zero) e PO (Positivo).

No modelo *fuzzy*, os parâmetros das funções de pertinência referentes as variáveis de entrada ($\vec{a}$ e $\vec{C}$) foram ajustados de forma linear, pois o mecanismo de adaptação da variável $\vec{a}$ do GWO original decresce linearmente de 2 a 0. O número de três funções pertinência foi o mesmo que Nafar *et al*. [12] adotou no seu sistema *fuzzy* que foi implementado no

algoritmo PSO. Os limites inferior e superior dessas funções de pertinência estão de acordo com o intervalo de valores recomendado por Mirjalili *et al*. [4], ou seja, $\vec{a} \in [0, 2]$ e $\vec{C} \in [0, 2]$. A configuração das funções de pertinência das variáveis de saída ($\Delta\vec{a}$ e $\Delta\vec{C}$) foram ajustadas empiricamente, os valores das mesmas estão representados na Tabela I.

TABELA I
PARÂMETROS DAS FUNÇÕES DE PERTINÊNCIAS

Função de pertinência da entrada NFV			
Funções de pertinência	Limite inferior	Valor de Pico	Limite superior
PQ	0,0	0,0	0,5
MD	0,0	0,5	1,0
GD	0,5	1,0	1,0
Função de pertinência da entrada $\vec{a}$			
Funções de pertinência	Limite inferior	Valor de Pico	Limite superior
PQ	0,0	0,0	1,0
MD	0,0	1,0	2,0
GD	1,0	2,0	2,0
Função de pertinência da entrada $\vec{C}$			
Funções de pertinência	Limite inferior	Valor de Pico	Limite superior
PQ	0,0	0,0	1,0
MD	0,0	1,0	2,0
GD	1,0	2,0	2,0
Função de pertinência da saída $\Delta\vec{a}$			
Funções de pertinência	Limite inferior	Valor de Pico	Limite superior
NE	-0,0005	-0,0005	0,0
ZE	-0,0005	0,0	0,0005
PO	0,0	0,0005	0,0005
Função de pertinência da saída $\Delta\vec{C}$			
Funções de pertinência	Limite inferior	Valor de Pico	Limite superior
NE	-0,005	-0,005	0,0
ZE	-0,005	0,0	0,005
PO	0,0	0,005	0,005

O conjunto de regras de produção do tipo "**Se** ¡ antecedente ou condição ¿ **Então** ¡ consequente ou ação ¿" foi elaborado seguindo os seguintes preceitos:

- O início do processo de otimização do algoritmo GWO representa o começo de uma caçada. Nesta etapa inicial, o parâmetro $\vec{a}$ inicia com um valor grande, enquanto que o parâmetro $\vec{C}$ inicia um valor relativamente "pequeno". Para os lobos da população que apresentarem um valor NFV grande, médio ou pequeno, então o valor do coeficiente $\vec{a}$ dos mesmos deverá ser diminuído, enquanto que o valor do parâmetro $\vec{C}$ deverá ser aumentado. Isto significa que esses lobos estão explorando um território e escolhendo uma presa em potencial para cercar e atacar;
- O final do processo de otimização é representado pelo fim da perseguição da presa. Em geral, o parâmetro $\vec{a}$ apresenta um valor pequeno, enquanto que o parâmetro $\vec{C}$ apresenta um valor "grande". Isto significa que os lobos cercaram uma presa em potencial e estão se preparando para o ataque. Para os lobos da população que apresentarem um valor NFV pequeno, logo, os valores dos parâmetros $\vec{a}$ e $\vec{C}$ relacionados aos mesmos devem ter uma variação desprezível, pois isto significa que esses lobos convergiram para uma solução;
- No decorrer das iterações um lobo pode apresentar um valor NVF "médio" ou "grande", porém seus parâmetros $\vec{a}$ e $\vec{C}$ podem apresentar valores que são mais condizentes com final do processo de otimização (valor de $\vec{a}$ pequeno e valor de $\vec{C}$ grande). Nesta situação pode ocorrer problemas de convergência prematura, onde o algoritmo pode ficar preso em algum mínimo local. Para evitar esta situação, é necessário aumentar o valor do parâmetro $\vec{a}$ e diminuir o valor do parâmetro $\vec{C}$ de forma a aumentar o território de exploração do lobo.

O conjunto de regras que relacionam as entradas NFV, $\vec{a}$ e $\vec{C}$ com as saídas $\Delta\vec{a}$ e $\Delta\vec{C}$ do modelo *fuzzy* estão expressas na Tabela II.

TABELA II
CONJUNTO DE REGRAS

Inferência para $\Delta\vec{a}$ e $\Delta\vec{C}$	
Se: NFV é PQ e $\vec{a}$ é PQ	**Então:** $\Delta\vec{a}$ é ZE
Se: NFV é PQ e $\vec{a}$ é MD	**Então:** $\Delta\vec{a}$ é NE
Se: NFV é PQ e $\vec{a}$ é GD	**Então:** $\Delta\vec{a}$ é NE
Se: NFV é MD e $\vec{a}$ é PQ	**Então:** $\Delta\vec{a}$ é PO
Se: NFV é MD e $\vec{a}$ é MD	**Então:** $\Delta\vec{a}$ é ZE
Se: NFV é MD e $\vec{a}$ é GD	**Então:** $\Delta\vec{a}$ é NE
Se: NFV é GD e $\vec{a}$ é PQ	**Então:** $\Delta\vec{a}$ é PO
Se: NFV é GD e $\vec{a}$ é MD	**Então:** $\Delta\vec{a}$ é ZE
Se: NFV é GD e $\vec{a}$ é GD	**Então:** $\Delta\vec{a}$ é NE
Se: NFV é PQ e $\vec{C}$ é PQ	**Então:** $\Delta\vec{C}$ é PO
Se: NFV é PQ e $\vec{C}$ é MD	**Então:** $\Delta\vec{C}$ é PO
Se: NFV é PQ e $\vec{C}$ é GD	**Então:** $\Delta\vec{C}$ é ZE
Se: NFV é MD e $\vec{C}$ é PQ	**Então:** $\Delta\vec{C}$ é PO
Se: NFV é MD e $\vec{C}$ é MD	**Então:** $\Delta\vec{C}$ é ZE
Se: NFV é MD e $\vec{C}$ é GD	**Então:** $\Delta\vec{C}$ é NE
Se: NFV é GD e $\vec{C}$ é PQ	**Então:** $\Delta\vec{C}$ é PO
Se: NFV é GD e $\vec{C}$ é MD	**Então:** $\Delta\vec{C}$ é ZE
Se: NFV é GD e $\vec{C}$ é GD	**Então:** $\Delta\vec{C}$ é NE

III. RESULTADOS E DISCUSSÃO

Para este artigo, a implemetação dos algoritmos e a obtenção dos seus respectivos resultados foi realizada por meio da utilização do ambiente computacional Matlab® [16]. A seguir serão apresentados os critérios de análise e comparação e após os resultados obtidos.

A. Ajuste Fuzzy dos Parâmetros do Algoritmo GWO em Função do Número de Iteração Corrente da Simulação

O método proposto deste trabalho foi comparado com a metodologia proposta por Rodriguez *et al*. [8] que também utiliza um sistema *fuzzy*. O mecanismo *fuzzy* proposto por Rodriguez *et al*. [8] utiliza o número da iteração normalizada corrente da simulação para o ajuste simultâneo dos parâmetros de controle $\vec{a}$ e $\vec{C}$ do GWO.

O sistema *fuzzy* de Rodriguez *et al*. [8] utiliza o modelo linguístico do tipo Mamdani com funções de pertinência triangular e método de defuzzificação por centroide. Os termos linguísticos adotados foram: "*low*" (baixo), "*medium*" (médio) e "*high*" (alto). O sistema *fuzzy*, o conjunto de regras e os parâmetros das funções pertinências estão representadas respectivamente na Fig. 3, nas Tabelas III e IV.

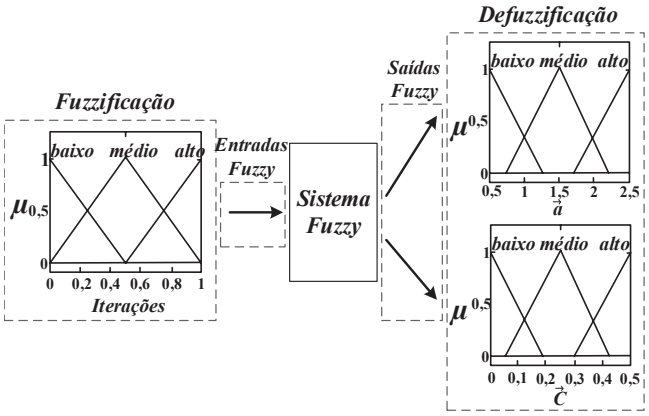

Fig. 3. Sistema *fuzzy* proposto pelos autores [8].

TABELA III
PARÂMETROS DAS FUNÇÕES DE PERTINÊNCIAS DO SISTEMA FUZZY PROPOSTO PELOS AUTORES [8]

Função de pertinência de entrada "iterações"			
Funções de pertinência	Limite inferior	Valor de Pico	Limite superior
baixo	0	0	0,4
médio	0,1	0,5	0,9
alto	0,6	1	1
Função de pertinência da saída $\vec{a}$			
Funções de pertinência	Limite inferior	Valor de Pico	Limite superior
baixo	0,5	0,5	1,3
médio	0,7	1,5	2,3
alto	1,7	2,5	2,5
Função de pertinência da saída $\vec{C}$			
Funções de pertinência	Limite inferior	Valor de Pico	Limite superior
baixo	0	0	0,2
médio	0,05	0,25	0,45
alto	0,3	0,5	0,5

TABELA IV
CONJUNTO DE REGRAS DO SISTEMA *fuzzy* PROPOSTO PELOS AUTORES [8]

	Conjunto de regras
1	**Se:** (iterações é baixo) **Então:** ($\vec{a}$ é alto e $\vec{C}$ é baixo)
2	**Se:** (iterações é médio) **Então:** ($\vec{a}$ é médio e $\vec{C}$ é médio)
3	**Se:** (iterações é alto) **Então:** ($\vec{a}$ é baixo e $\vec{C}$ é alto)

B. Funções Testes

Para avaliar a convergência do mecanismo *fuzzy* proposto foram utilizadas dez funções testes de otimização que são muito conhecidas na literatura e utilizadas para avaliar o desempenho de metaheurísticas de otimização [17]. As funções testes utilizadas para validar os otimizadores neste artigo estão descritas na Tabela V, e podem ser encontrados nos seguintes trabalhos [3], [18]–[20].

Nas funções testes (detalhes na Tabela V) a variável x_i representa os parâmetros que serão otimizados e n é a dimensão das funções testes. Para o processo de otimização, foi definida a dimensão n das funções testes iguais a 30.

Em competições cujo objetivo é avaliar o desempenho de algoritmos de otimização, estas funções testes são utilizadas na obtenção de problemas de otimização complexos e desafiadores cuja a solução requer um custo computacional elevado [21]. Na Tabela V, a metade das funções testes são do tipo unimodal (*Sphere*, *Schwefel 2.22*, *Schwefel 1.2*, *Schwefel 2.21* e *Quartic*), enquanto que a outra metade é do tipo multimodal (*Rastrigin*, *Ackley*, *Griewank*, *Weierstrass* e *HGbat*). Independentemente do tamanho de n, todas as funções testes possuem mínimo global igual à zero.

TABELA V
FUNÇÕES TESTE

Nome da função	Equação	Intervalo de busca
Sphere	$f(x) = \sum_{i=1}^{n} x_i^2$	[-100:100]
Schwefel 2.22	$f(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	[-10:10]
Schwefel 1.2	$f(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2$	[-100:100]
Schwefel 2.21	$f(x) = \max_i \{\lvert x_i \rvert, 1 \leq i \leq n\}$	[-100:100]
Quartic	$f(x) = \sum_{i=1}^{n} i x_i^4 + \text{random}[0,1]$	[-1,28:1,28]
Rastrigin	$f(x) = \sum_{i=1}^{n} (x_i^n - 10\cos(2\pi x_i) + 10)$	[-5,12:5,12]
Ackley	$f(x) = 20\exp\left(-0,2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	[-32:32]
Griewank	$f(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	[-600:600]
Weierstrass	$f(x) = \sum_{i=1}^{n}\left(\sum_{k=0}^{\mathbf{kmax}}[a^k\cos(2\pi b^k(x_i+0,5))]\right) - n\sum_{k=0}^{\mathbf{kmax}}[a^k\cos(2\pi b^k.2,5)]$ **onde:** $a = 0,5$**,** $b = 3$ **e kmax** $= 20$	[-0,5:0,5]
HGBat	$f(x) = \left\lvert \left(\sum_{i=1}^{n} x_i^2\right)^2 - \left(\sum_{i=1}^{n} x_i\right)^2 \right\rvert^{\frac{1}{2}} + \frac{1}{n}\left(0,5\sum_{i=1}^{n} x_i^2 + \sum_{i=1}^{n} x_i\right) + 0,5$	[-100:100]

C. Critérios de Análise e Comparação

Com o intuito de avaliar a convergência durante o processo de otimização das funções testes, os algoritmos GWO utilizados neste trabalho foram submetidos a uma série de 1000 simulações. Cada simulação foi realizada com 500 iterações. Cada algoritmo utiliza uma população de lobos composta por 30 lobos. O valor médio do ótimo de cada função teste obtido por meio da realização de 1000 simulações é utilizada como critério para avaliar cada algoritmo. Para a comparação dos resultados obtidos pelos algoritmos, foi utilizado como método estatístico o teste de Friedman com um de nível de confiança α de 5% ($\alpha = 0,05$).

O teste Friedman é um tipo de teste estatístico não paramétrico que é muito utilizado em comparações múltiplas [22]. O procedimento deste teste consiste em organizar o

desempenho dos algoritmos por meio de uma classificação (*ranking*) e depois é feita uma análise das variâncias dos seus valores [23].

Neste trabalho, o teste de Friedman foi realizado seguindo o tutorial de Derrac *et al*. [23] que orienta o uso de testes estatísticos não paramétricos com a metodologia voltada a comparação de algoritmos de inteligência de enxame. O teste de Friedman foi aplicado de forma a fazer uma comparação entre todos os algoritmos, onde um par de algoritmos é comparado por vez. Neste caso, o teste de Friedman realiza um número m comparações por pares de algoritmos. Para este teste foi utilizado o método de controle *post-hoc* de Holm de forma a melhorar a comparação feita pelo teste Friedman, ajustando o valor do nível de confiança α. O valor da probabilidade p é determinado diretamente pela área da curva de distribuição normal. Para determinar esta área é utilizado o valor da estatística teste Z_t que compara o valor do *ranking* de dois algoritmos diferentes, este valor é calculado por:

$$Z_t = \frac{(R_i - R_j)}{\sqrt{\frac{k(k+1)}{6N}}} \tag{15}$$

onde R_i e R_j são respectivamente os *ranks* do algoritmo i (o pior da comparação) e do algoritmo j (o melhor da comparação). A variável k representa a quantidade de algoritmos utilizados no teste de Friedman, enquanto que N representa a quantidade de funções testes que são utilizadas na comparação.

Neste trabalho, o método de Holm faz o ajuste do valor de α de maneira *stepdown* (abaixamento). Os valores das probabilidades $p_1, p_2, ..., p_m$ obtidas pelas comparações são ordenadas de forma crescente ($p_1 \leq p_2 \leq ... \leq p_m$) junto com suas hipóteses correspondentes $H_1, H_2, ..., H_m$ que representam as hipóteses nulas. O método de Holm rejeita H_1 para H_m se i é o menor valor inteiro tal que $p_i > \alpha/(m - i + 1)$. O procedimento *stepdown* de Holm começa com o valor p mais significativo. Se p_1 está abaixo de α/m, então a hipótese correspondente H_1 é rejeitada. Em seguida, é permitido comparar p_2 com $\alpha/(m - 1)$. Se a segunda hipótese H_2 é rejeitada, o teste prossegue com o terceiro e assim por diante. Se uma certa hipótese nula não puder ser rejeitada, as hipóteses restantes também não serão rejeitadas. Quando a hipótese nula de uma comparação de dois algoritmos é rejeitada, significa que o algoritmo de menor *ranking* médio converge melhor para o mínimo global das funções testes, conforme apresentado na Tabela V. Caso contrário, o algoritmo de menor *ranking* médio não é estatisticamente melhor que o outro algoritmo da comparação.

D. Resultados do Mecanismo Fuzzy Proposto

Os resultados obtidos estão representados nas Tabelas VI, VII e VIII, e também na Fig. 4. Nos resultados, o mecanismo *fuzzy* proposto para o algoritmo GWO é identificado por "*fuzzy_NFV*" enquanto que o modelo *fuzzy* proposto por Rodriguez *et al*. [8] é identificado por "*fuzzy_iter*". Já o algoritmo otimizador do lobo cinzento convencional é identificado por GWO.

TABELA VI
VALORES DE DISPERSÃO OBTIDOS NAS 1000 SIMULAÇÕES

Algoritmo	Função teste	Valores obtidos min	max	μ	σ
GWO	*Sphere*	2,39e-30	4,72e-26	1,39e-27	3,08e-27
	Schwefel 2.22	9,53e-18	8,28e-16	9,81e-17	8,58e-17
	Schwefel 1.2	6,22e-10	0,0022	2,18e-5	1,24e-4
	Schwefel 2.21	1,88e-8	2,22e-5	8,27e-7	1,21e-6
	Quartic	2,52e-4	0,0066	0,0019	0,0010
	Rastrigin	0	65,639	2,9167	4,8802
	Ackley	6,13e-14	2,07e-13	1,02e-13	1,71e-14
	Griewank	0	0,0731	0,0042	0,0731
	Weierstrass	0	1,55e-15	1,25e-16	2,03e-16
	HGBat	0,2858	1,0337	0,4938	0,0791
fuzzy _iter	*Sphere*	4,24e-54	5,31e-46	8,98e-48	3,99e-47
	Schwefel 2.22	3,25e-30	8,55e-27	3,03e-28	4,81e-28
	Schwefel 1.2	9,38e-18	1,36e-7	1,19e-9	8,83e-9
	Schwefel 2.21	2,48e-17	1,96e-13	7,22e-15	1,67e-14
	Quartic	2,29e-5	0,0036	6,62e-4	4,56e-4
	Rastrigin	2,92e-5	0,0045	7,07e-4	5,13e-4
	Ackley	4,44e-15	1,51e-14	1,03e-14	2,90e-15
	Griewank	0	0,0773	1,45e-4	0,0026
	Weierstrass	0	0	0	0
	HGBat	0,3354	0,5914	0,4767	0,0290
fuzzy _NFV	*Sphere*	1,13e-63	1,22e-56	9,89e-59	5,55e-58
	Schwefel 2.22	3,42e-36	3,16e-32	1,12e-33	2,09e-33
	Schwefel 1.2	9,71e-20	5,27e-7	6,73e-10	1,68e-8
	Schwefel 2.21	2,60e-21	2,22e-16	3,36e-18	1,17e-17
	Quartic	2,92e-5	0,0045	7,06e-4	5,13e-4
	Rastrigin	0	35,5252	0,0570	1,3122
	Ackley	4,44e-15	1,51e-14	8,98e-15	2,14e-15
	Griewank	0	0,0421	1,81e-4	0,0020
	Weierstrass	0	0	0	0
	HGBat	0,3907	0,5855	0,4815	0,0264

TABELA VII
Ranks DO TESTE DE FRIEDMAN

Função Teste	GWO	*fuzzy_iter*	*fuzzy_NFV*
Sphere	3	2	1
Schwefel 2.22	3	2	1
Schwefel 1.2	3	2	1
Schwefel 2.21	3	2	1
Quartic	3	1	2
Rastrigin	3	1	2
Ackley	3	2	1
Griewank	3	1	2
Weierstrass	3	1,5	1,5
HGBat	3	1	2
Ranking médio	3	1,55	1,45

TABELA VIII
RESULTADO OBTIDO PELO TESTE DE FRIEDMAN

Hipóteses	Z_t	p	$p_i > \frac{\alpha}{m-i+1}$
H_1: *GWO vs fuzzy_ NFV*	3,4659	0,000264	Rejeitada
H_2: *GWO vs fuzzy_iter*	3,2423	0,000593	Rejeitada
H_3: *fuzzy_iter vs fuzzy_NFV*	0,2236	0,411534	Aceita

A Tabela VI mostra os resultados referentes aos valores ótimos das funções testes obtidas nas 1000 simulações. Nesta

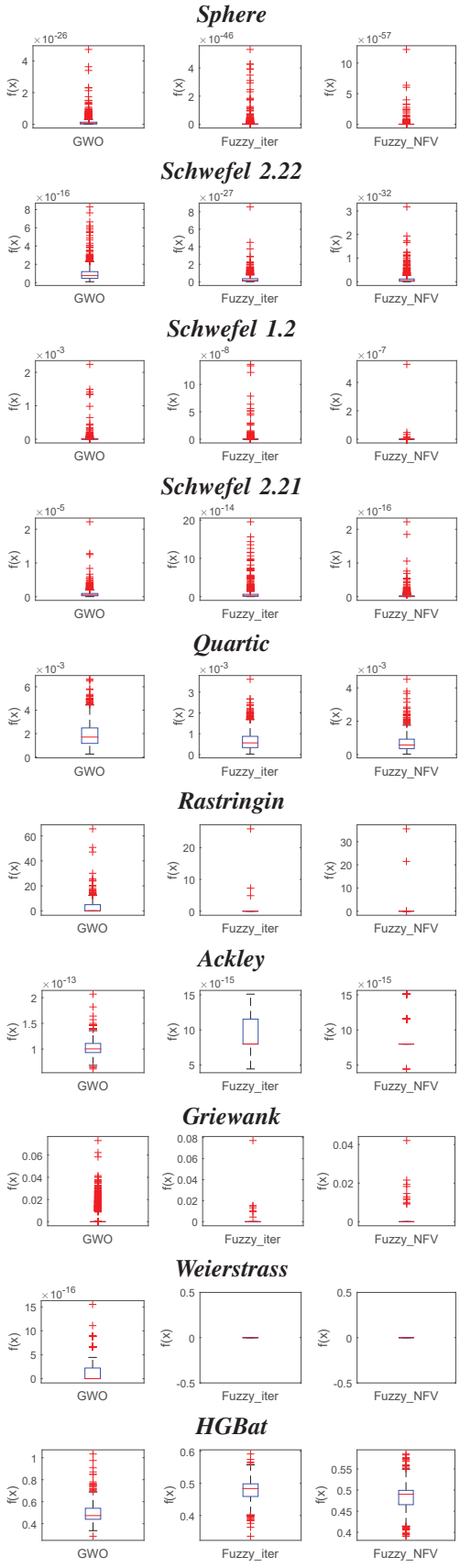

Fig. 4. Diagrama de caixa referente aos valores ótimos obtidos durante o processo de otimização das funções testes nas 1000 simulações.

tabela, o melhor e o pior resultado são representados respectivamente por "min" e "max". Os valores referentes a média e desvio padrão são representados respectivamente por μ e σ. Pode-se observar ainda na mesma tabela que na maior parte dos casos, o mecanismo *fuzzy* proposto apresentou os menores valores médios. O diagrama de caixa mostrado na Fig. 4 ajuda a reforçar os resultados obtidos na Tabela VI mostrando que o mecanismo *fuzzy* proposto apresentou valores menos discrepantes na maioria dos casos quando comparado com o algoritmo GWO convencional e o mecanismo *fuzzy* proposto por Rodriguez *et al*. [8].

A Tabela VII de *ranks* foi elaborada a partir dos valores médios (μ) dos algoritmos da Tabela VI. Na otimização de uma determinada função teste, o algoritmo com o melhor resultado foi atribuído um valor de *rank* igual a 1, para o segundo e terceiro, respectivamente 2 e 3. Em relação aos outros algoritmos, observa-se na Tabela VIII que o mecanismo *fuzzy* proposto apresentou o menor valor do *rank* médio no processo de otimização das dez funções testes.

Na Tabela VIII, o teste de Friedman mostra que o mecanismo *fuzzy* proposto se mostrou superior estatisticamente superior em relação ao algoritmo GWO convencional (a hipótese nula H_1 foi rejeitada). Em relação a metodologia apresentada por Rodriguez *et al*. [8], o desempenho do mecanismo proposto mostrou-se competitivo. Isto pode ser verificado pelo teste de Friedman, pois o mesmo não se mostrou estatisticamente superior (a hipótese nula H_3 foi aceita).

IV. Conclusão

Este trabalho propôs um mecanismo de ajuste de parâmetros para o algoritmo otimizador do lobo cinzento baseado em lógica nebulosa que utiliza como informação de entrada os valores dos parâmetros do processo de caça do GWO e o valor do *fitness* normalizado de cada indivíduo da população. Ao contrário das outras variantes do algoritmo GWO, o modelo *fuzzy* proposto não utiliza o valor da iteração atual para ajustar os parâmetros da equação que é responsável por imitar o comportamento de caça dos lobos cinzentos.

O mecanismo proposto melhorou a convergência do algoritmo GWO convencional. No processo de otimização das dez funções testes, o modelo *fuzzy* proposto neste trabalho mostrou-se superior em relação a versão convencional. O modelo proposto se mostrou competitivo em comparação com outro modelo *fuzzy* que utiliza como informação de entrada o valor da iteração normalizada. Isto foi confirmado via teste de Friedman.

Em trabalhos futuros, outros autores podem utilizar o mecanismo *fuzzy* proposto com a utilização do GWO voltado a aplicação de problemas de otimização multiobjetivo. Este mecanismo *fuzzy* pode ainda ser explorado e implementado nos outros algoritmos de inteligência de enxame.

Referências

[1] K.-L. Du, M. Swamy *et al.*, "Search and optimization by metaheuristics," *Birkhaüser) July*, 2016.

[2] S. Salcedo-Sanz, "Modern meta-heuristics based on nonlinear physics processes: A review of models and design procedures," *Physics Reports*, vol. 655, pp. 1–70, 2016.

[3] E. H. Silva and C. J. Bastos Filho, "PSO efficient implementation on GPUs using low latency memory," *IEEE Latin America Transactions*, vol. 13, no. 5, pp. 1619–1624, 2015.
[4] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in engineering software*, vol. 69, pp. 46–61, 2014.
[5] J.-J. Liang, P. N. Suganthan, and K. Deb, "Novel composition test functions for numerical global optimization," in *Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE*. IEEE, 2005, pp. 68–75.
[6] M. Kohli and S. Arora, "Chaotic grey wolf optimization algorithm for constrained optimization problems," *Journal of Computational Design and Engineering*, vol. 5, no. 4, pp. 458–472, 2018.
[7] S. Mirjalili, S. Saremi, S. M. Mirjalili, and L. d. S. Coelho, "Multi-objective grey wolf optimizer: a novel algorithm for multi-criterion optimization," *Expert Systems with Applications*, vol. 47, pp. 106–119, 2016.
[8] L. Rodriguez, O. Castillo, M. Garcia, J. Soria, F. Valdez, and P. Melin, "Dynamic simultaneous adaptation of parameters in the grey wolf optimizer using fuzzy logic," in *Fuzzy Systems (FUZZ-IEEE), 2017 IEEE International Conference on*. IEEE, 2017, pp. 1–6.
[9] E. Emary, H. M. Zawbaa, and C. Grosan, "Experienced gray wolf optimization through reinforcement learning and neural networks," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 3, pp. 681–694, 2018.
[10] N. Jain, G. Parmar, R. Gupta, and I. Khanam, "Performance evaluation of GWO/PID approach in control of ball hoop system with different objective functions and perturbation," *Cogent Engineering*, vol. 5, no. 1, p. 1465328, 2018.
[11] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE transactions on evolutionary computation*, vol. 1, no. 1, pp. 67–82, 1997.
[12] M. Nafar, G. B. Gharehpetian, and T. Niknam, "Using modified fuzzy particle swarm optimization algorithm for parameter estimation of surge arresters models," *International Journal of Innovative Computing, Information and Control*, vol. 8, no. 1, pp. 567–581, 2012.
[13] R. Storn and K. Price, "Differential evolution: a simple and efficient heuristic for global optimization over continuous spaces," *Journal of global optimization*, vol. 11, no. 4, pp. 341–359, 1997.
[14] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "GSA: a gravitational search algorithm," *Information sciences*, vol. 179, no. 13, pp. 2232–2248, 2009.
[15] A. C. K. Ferrari, G. V. Leandro, and G. Oliveira, "Mecanismos de Ajuste do Parâmetro Peso de Inercia do Algoritmo PSOGSA na Identificação de um Sistema Multivariável Não Linear," in *XXI Congresso Brasileiro de Automática - CBA2016 UFES, Vitória - ES*. CBA, 2016, pp. 495–500.
[16] Math Works, Inc. MATLAB. [Online]. Available: http://www.mathworks.com. Acessado em 01/10/2018.
[17] M. Jamil and X.-S. Yang, "A literature survey of benchmark functions for global optimization problems," *arXiv preprint arXiv:1308.4008*, 2013.
[18] J. David Velasquez, "A simple monte carlo optimizer based on adaptive coordinate sampling," *IEEE Latin America Transactions*, vol. 12, no. 2, pp. 236–243, March 2014.
[19] F. A. Procopio Paiva, J. A. Ferreira Costa, and C. Rodrigues Muniz Silva, "A serendipity-based approach to enhance particle swarm optimization using scout particles," *IEEE Latin America Transactions*, vol. 15, no. 6, pp. 1101–1112, June 2017.
[20] L. H. Reis Jesus and L. Cunha Brito, "Interactive evolution strategies for minimizing single-objective functions," *IEEE Latin America Transactions*, vol. 15, no. 5, pp. 981–987, May 2017.
[21] Q. Chen, B. Liu, Q. Zhang, J. Liang, P. Suganthan, and B. Qu, "Problem definitions and evaluation criteria for CEC 2015 special session on bound constrained single-objective computationally expensive numerical optimization," *Technical Report, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Technical Report, Nanyang Technological University*, 2014.
[22] L. Ledo, M. Regattieri Delgado, and J. Valente de Oliveira, "Synthesis of probabilistic fuzzy classifiers using gk clustering and bayesian estimation," *IEEE Latin America Transactions*, vol. 15, no. 3, pp. 550–556, March 2017.
[23] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 3–18, 2011.

Allan Christian Krainski Ferrari Mestre em Engenharia Elétrica na área de sistemas eletrônicos pela Universidade Federal do Paraná - UFPR (2015). Graduado em Engenharia Elétrica pela UFPR (2011). Atualmente é professor do Centro Universitário UNIFACEAR e do Centro Universitário UNISOCIESC. Suas principais áreas de interesse são automação, estudo de metaheurísticas, sistemas de controle e modelagem matemática.

Gideon Villar Leandro Possui graduação em Engenharia Elétrica pela Universidade Estadual Paulista Júlio de Mesquita Filho (1989), Ilha Solteira, mestrado em Engenharia Elétrica pela Universidade Federal da Paraíba (1992), Campina Grande e doutorado em Engenharia Elétrica pela Universidade Estadual de Campinas (2000). Atualmente é professor associado I da Universidade Federal do Paraná. Tem experiência na área de Engenharia Elétrica, com ênfase em Automação e Controle, atuando principalmente nos seguintes temas: identificação de sistemas, sistemas de controle, sistemas a eventos discretos e modelagem matemática.

Leandro dos Santos Coelho possui graduação em Informática pela Universidade Federal de Santa Maria (1994), graduação em Engenharia Elétrica pela Universidade Federal de Santa Maria (1999), mestrado em Ciências da Computação pela Universidade Federal de Santa Catarina (1997) e doutorado em Engenharia Elétrica pela Universidade Federal de Santa Catarina (2000). Atualmente, é professor titular da Pontifícia Universidade Católica do Paraná (Engenharia Mecatrônica / Produção) e professor adjunto na Universidade Federal do Paraná (Engenharia Elétrica). Atua como professor permanente do PPGEPS/PUCPR (Pós-graduação em Engenharia de Produção e Sistemas - mestrado e doutorado) e também é professor colaborador do PGEE/UFPR (Pós-Graduação em Engenharia Elétrica - mestrado, Grupo de Pesquisa de Operação e Planejamento de Sistemas Elétricos de Potência) e da UnB (Pós-Graduação em Sistemas Mecatrônicos da Universidade de Brasília - doutorado). Suas áreas de interesse incluem aprendizado de máquina, identificação de sistemas, controle avançado, previsão de séries temporais, biomecânica e metaheurísticas.

Carlos Alexandre Gouvea da Silva é Engenheiro de Computação pela PUCPR em 2012, Mestre em 2015 e doutorando em Engenharia Elétrica pela UFPR. É professor 20 horas no Departamento de Engenharia Elétrica UFPR, membro da Sociedade Brasileira de Telecomunicações e do IEEE. Suas principais áreas de interesse são telecomunicações, redes de computadores, educação e qualidade de ensino em engenharias.

Edmilson Gabriel de Lima Mestre em Engenharia Mecânica pela UFPR, na área de Fenômenos de Transportes e Mecânica dos Sólidos com ênfase em engenharia de produto. Especialista em Desenvolvimento e Processo de Componentes Plásticos pela UNISOCIESC. Especialista em Gestão da Educação pela Faculdades Integradas Camões. Bacharel em Administração de Empresas pela Fundação de Estudos Sociais do Paraná e Técnico em Mecânica Industrial pela UTFPR. Atualmente é professor do Centro Universitário UNIFACEAR e do Centro Universitário UNISOCIESC.

Carlos Roberto Chaves Doutorando pela Universidade de São Paulo (USP), graduado em Engenharia Eletrônica com Ênfase em Telecomunicações pela Universidade do Contestado (UnC), Bacharel em Informática, Tecnólogo em Processamento de Dados, Especialização em Automação Industrial UFPR, Mestre em Engenharia Elétrica pela UFPR. Trabalha na área de Automação Industrial - Petróleo Brasileiro, Professor Universitário da UnC e San-Martin. Experiência nas áreas de Engenharia Elétrica, Eletrônica Industrial, Sistemas e Controles Eletrônicos.