Universidade de Pernambuco
Escola Politécnica de Pernambuco
Programa de Pós-Graduação em Engenharia da Computação

Murilo Rebelo Pontes

Otimização por Enxames de Partículas utilizando Clãs com Comportamento Adaptativo

Dissertação de Mestrado

Recife, Agosto de 2010

Universidade de Pernambuco
Escola Politécnica de Pernambuco
Programa de Pós-Graduação em Engenharia da Computação

Murilo Rebelo Pontes

Otimização por Enxames de Partículas utilizando Clãs com Comportamento Adaptativo

s

Dissertação de Mestrado

DISSERTAÇÃO APRESENTADA AO PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DA COMPUTAÇÃO DA UNIVERSIDADE DE PERNAMBUCO COMO REQUISITO PARCIAL PARA OBTENÇÃO DO TÍTULO DE MESTRE EM ENGENHARIA DA COMPUTAÇÃO.

Orientador: Prof. Dr. Carmelo José Bastos Filho
Co-Orientador: Prof. Dr. Fernando Buarque de Lima Neto

Recife, Agosto/2010

P811o Pontes, Murilo Rebelo

Otimização por Enxames de Partículas utilizando Clãs com Comportamento Adaptativo / Murilo Rebelo Pontes / – Recife : O Autor, 2010
VII, 100 pág , 27 fig. 11 Tab.

Dissertação (mestrado) – Universidade de Pernambuco. Programa de Pós-Graduação em Engenharia da Computação, Recife, BR-PE 2010. Orientador: Carmelo José Bastos Filho; Co-orientador: Fernando Buarque de Lima Neto;

1. Comportamento Adaptativo, 2.Otimização por enxame de partículas, 3.Clãs cooperativos, 4.Inteligência de enxames.

CDU 681.51:007.52

Dissertação de Mestrado apresentada por **Murilo Rebelo Pontes**, à Pós-Graduação em Engenharia da Computação da Escola Politécnica de Pernambuco da Universidade de Pernambuco, sob o título "**Otimização por Enxames de Partículas utilizando Clãs com Comportamento Adaptativo**", orientado por Carmelo José Albanez Bastos Filho — Doutor e aprovada pela Banca Examinadora formada pelos professores:

Prof. Francisco Madeiro Bernardino Júnior - Doutor
(primeiro examinador)

Prof. Leandro Nunes de Castro Silva - Doutor
(segundo examinador)

Prof. Carmelo José Albanez Bastos Filho- Doutor
(terceiro examinador)

Visto e permitida a impressão.

Recife 13 de agosto de 2010

Prof. Fernando Buarque de Lima Neto
Coordenador da Pós-Graduação em Engenharia da Computação da
Escola Politécnica de Pernambuco da Universidade de Pernambuco

Agradecimentos

Quero agradecer a Deus por ter criado todos nós. A minha família por ter me criado e educado até agora e pelo resto da vida. Aos amigos, por fazer do mundo um lugar legal para viver e passar o tempo. Aos colegas de profissão pelo reconhecimento do meu trabalho e oportunidades oferecidas. A todos os desconhecidos por aumentarem a entropia do mundo.

Agradeço a meus orientadores professores Carmelo Bastos e professor Fernando Buarque. Sem dúvida posso afirmar que todas as críticas e questionamentos foram para enriquecer o trabalho, e também meu desenvolvimento pessoal e profissional. Além disso, preciso dizer que desde que os conheci na graduação são profissionais, que "vestem a camisa e abraçam a causa", trabalhando duro para manter um nível de excelência nos projetos que se envolvem.

Aos amigos do Quartel General de Pesquisa (QG): Robson Santana e Salomão Madeiro. Foram vários meses compartilhando experiências das mais diversas possíveis. Desejo sucesso a esses companheiros de guerra na próxima etapa da vida acadêmica.

Aos meus amigos Marcos Álvares e Hugo Serrano, no desenrolar de várias histórias.

A minha amiga Sílvia Fabiane por todas as participações e apoio em momentos importantes da minha vida.

Aos colegas do grupo de pesquisa de inteligência computacional (CIRG) e todos os professores da instituição que contribuíram direta e indiretamente na minha formação profissional.

Resumo

Os algoritmos de otimização por enxames de partículas têm sido amplamente utilizados para resolver problemas reais, principalmente quando existem muitas variáveis a serem otimizadas e estas, por sua vez, são contínuas. Nos enxames encontrados na natureza podem ser observados comportamentos cooperativos. Uma das formas mais comuns de cooperação é a colaboração, onde os integrantes de uma mesma população se agrupam para resolver partes de um problema maior, semelhante à estratégia dividir para conquistar.

Existem algoritmos de otimização por enxames de partículas que incorporam cooperação colaborativa, como *Clan Particle Swarm Optimization*, *Cooperative Particle Swarm Optimization* e *Multi-Swarm Cooperative Particle Swarm Optimization*. No entanto, os algoritmos de otimização por enxames de partículas apresentam algumas deficiências como: baixa velocidade de convergência e limitada capacidade de escapar de ótimos locais. Para mitigar esses problemas, foi proposto, no algoritmo *Adaptive particle swarm optimization*, um processo de adaptação de parâmetros.

Inspirada pelos conceitos de cooperação colaborativa e adaptação automática de parâmetros, esta dissertação, propõe o algoritmo *Clan Adaptive Particle Swarm Optimization*. Partiu-se da hipótese de que o emprego de ambos conceitos em um único algoritmo pode produzir um ganho de desempenho maior do que a simples execução intercalada de algoritmos possuem apenas um destes conceitos.

Para avaliar o desempenho do algoritmo proposto foram utilizados problemas de teste conhecidos na literatura especializada e realizadas análises referentes à adaptação de parâmetros, disposição das soluções geradas no espaço de busca e a influência da quantidade de clãs. Também foram realizadas comparações com outras técnicas colaborativas baseadas em inteligência de enxames e com outras técnicas de inteligência de enxames, como busca por cardumes e otimização por colônia de abelhas artificiais. Os resultados obtidos indicam que o algoritmo proposto possui desempenho superior, principalmente em problemas de mais alta dimensionalidade, *e.g.* problemas de otimização com 300 dimensões.

Palavras-chave: Comportamento Adaptativo, Otimização por enxame de partículas, Clãs cooperativos, Inteligência de enxames.

Particle swarm optimization with adaptive behavior using Clans

Abstract

Swarm intelligence algorithms are usually composed by simple entities that interact among themselves. Complex behavior patterns may emerge from this interaction. This feature can be used to solve search and optimization problems. A common characteristic presented by swarm intelligence algorithms is the ability to solve problems in a distributed way, with no prior knowledge about the entire problem model. There are several optimization algorithms based on swarms, such as ant colony, fish schools, flocks of birds, among others. Particle Swarm Optimization is one of these techniques and have been widely used to solve real problems, mainly when there are many variables to be optimized and these variables are continuous.

In nature, some cooperative behaviors can be observed. One of the most common form of cooperation is collaboration. There are some Particle Swarm Optimization variations that incorporate cooperative features, such as Clan Particle Swarm Optimization, Cooperative Particle Swarm Optimization and Cooperative Multi-Swarm Particle Swarm Optimization. Moreover, the basic Particle Swarm Optimization algorithm presents some shortcomings such as low convergence speed and a limited ability to escape from local optima. To mitigate these problems, the Adaptive particle swarm optimization algorithm was proposed, in which the parameter adaptation ability was included.

This work was motivated by the idea that it is possible to include the autoadaptation capacity into cooperative Particle Swarm Optimization algorithms. For this, we assume that the hybridization of these techniques can produce a better performance when compared to the interleaved execution of these techniques.

The main idea is to include both concepts of collaboration and adaptation of parameters in Particle Swarm Optimization. The proposed algorithm is called Clan Adaptive Particle Swarm Optimization. To evaluate the performance of this algorithm, we used some well known benchmark problems in the literature. Some analysis concerning the adaptation of parameters, distribution of solutions generated in the search space along algorithm execution and the influence of the clans were done. We also performed comparisons with other colaborative techniques and other swarm intelligence techniques, such as fish school search and artificial bee colony. The results showed that the proposed algorithm achived better performance than the other algorithms, specially in problems with higher dimensionality, *e.g.* optimization problems with 300 dimensions.

Keywords: Adaptive behavior, Particle swarm optimization, Clans cooperative, Swarm intelligence.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

ABC	Artificial Bee Colony
APSO	Adaptive Particle Swarm Optimization
ClanAPSO	Clan Adaptive Particle Swarm Optimization
ClanPSO	Clan Particle Swarm Optimization
CPSO	Cooperative Particle Swarm Optimization
CPSO-H_k	Cooperative Particle Swarm Optimization hídrido k-dimensional
CPSO-S	Cooperative Particle Swarm Optimization unidimensional
CPSO-S_k	Cooperative Particle Swarm Optimization k-dimensional
FSS	Fish School Search
GCPSO	Guaranteed Convergence Particle Swarm Optimization
MCPSO	Multi-Swarm Cooperative Particle Swarm Optimization
MCPSO-COL	Multi-Swarm Cooperative Particle Swarm Optimization Colaborativo
MCPSO-COM	Multi-Swarm Cooperative Particle Swarm Optimization Competitivo
PSO	Particle Swarm Optimization

Lista de Símbolos e Parâmetros

$\vec{x}(t)$ Vetor de posição no instante de tempo t

$\vec{v}(t)$ Vetor de velocidade no instante de tempo t

ω Coeficiente de inércia para PSO

c_1 Coeficiente de aceleração cognitivo para PSO

c_2 Coeficiente de aceleração social para PSO

φ Soma dos fatores cognitivos e sociais do PSO

χ Fator de constrição para a abordagem *Constricted* PSO

f_{evol} Fator evolucionário do APSO

δ Taxa de aceleração do APSO

μ Média da mutação do APSO

σ Desvio padrão da mutação do APSO

ρ Escalar de ajuste da perturbação da melhor partícula no algoritmo GCPSO

W Peso do peixe para FSS

$\Delta\vec{x}$ Variação de posição para FSS

Δf Variação da função objetivo para FSS

s_{ind} Passo do operador individual do FSS

s_{vol} Passo do operador volitivo do FSS

$\vec{I}$ Vetor de direção resultante para FSS

$\vec{B}$ Vetor de baricentro para FSS

$Q(\vec{x})$ Quantidade de néctar da fonte para ABC

P_i Probabilidade de seleção de uma fonte para ABC

Φ Fator de migração do MCPSO

Sumário

Capítulo 1

Introdução

Otimização se refere a escolha do melhor elemento dentre um conjunto de elementos disponíveis. Problemas de otimização consistem em encontrar os valores das variáveis de uma função de forma a obter o valor mínimo ou máximo da função.

Problemas de otimização podem ser classificados de diferentes formas. Quanto à natureza das variáveis podem ser classificados como: contínuos (números reais) e discretos (números inteiros). Quanto ao número de funções objetivo podem ser classificados como: mono-objetivo (uma função objetivo) ou multi-objetivo (duas ou mais funções objetivo). Quanto ao número de pontos ótimos podem ser classificados como: unimodais (único ponto ótimo) ou multimodais (diversos pontos ótimos). Quanto à restrições impostas podem ser classificados como: sem restrições (a única restrição são os limites do espaço de variáveis) e com restrições (alguns critérios precisam ser satisfeitos para que a solução seja aceita). Quanto ao números de variáveis podem ser classificados como: baixa dimensionalidade (até 3 dimensões) e alta dimensionalidade (a partir de 4 dimensões). Quanto ao variação da função objetivo no tempo podem ser classificados como: estáticos (a função objetivo é constante no tempo) e dinâmicos (a função objetivo varia com o tempo). Esta dissertação trata especificamente de problemas de otimização contínuos mono-objetivo estáticos sem restrições de alta dimensionalidade. Tais problemas são comumente encontrados em diversas áreas da engenharia, por exemplo: otimização de áreas e volumes, modelagem de antenas, redes neurais artificias, e outros.

A matemática aplicada é o ramo da matemática que estuda técnicas de otimização global. As técnicas de otimização global podem ser agrupadas em: determinísticas, estocásticas e heurísticas [4]. As estratégias determinísticas são caracterizadas pelo uso de "força bruta" para encontrar soluções. As estratégias estocásticas são caracterizadas pelo uso de componentes aleatórios para encontrar soluções. As estratégias heurísticas são caracterizadas pelo uso de comportamentos inteligentes para encontrar soluções. Entre principais estratégias heurísticas estão: algoritmos evolucionários (algoritmos genéticos), otimização baseada em enxames (*Particle Swarm Optimization* e *Ant Colony Optimization*), algoritmos meméticos, evolução diferencial e outros. Esta dissertação concentra-se na utilização de estratégias heurísticas de otimização baseada em enxames.

Inteligência de enxames foi uma expressão introduzida por Gerardo Beni e Jing Wang [5], no contexto de sistemas robóticos celulares. Enxames são sistemas formados por entidades que seguem regras simples e que interagem em seu ambiente. Da interação entre as entidades emergem padrões de comportamento globais, que não são observados individualmente. Em computação, o termo se refere ao conjunto de vários algoritmos que

incorporam estes princípios. No campo da matemática, Stephen Wolfram observou que comportamentos complexos podem surgir de regras simples, levantou a hipótese de que a complexidade da natureza também poderia ser explicada através de regras simples. No livro "A New Kind of Sience" [6], Wolfram argumenta que suas descobertas não são um fato isolado, e que possuem significância para todas as áreas da ciência.

Os algoritmos de inteligência de enxames inicialmente eram utilizados para simular comportamentos coletivos de animais e insetos da natureza. Posteriormente, foi observada a possibilidade de utilizá-los na resolução de problemas de busca e otimização. Sendo particularmente interessantes porque possuem a capacidade de adaptação ao problema e solução de forma distribuída, sem controle centralizado e sem um modelo completo do problema. Entre os algoritmos mais populares em inteligência de enxames para solução de problemas de otimização em espaços de variáveis contínuas sem restrições estão: otimização por enxame de partículas (PSO, *Particle Swarm Optimization*), busca baseada em cardumes (FSS, *Fish School Search*) e otimização por colônia de abelhas (ABC, *Artificial Bee Colony*), entre outros. É possível também encontrar diversas variações e hibridizações dessas técnicas aplicadas a problemas complexos.

O algoritmo de PSO [7] [8] foi inspirado no comportamento de bandos de pássaros (Ver ilustração na Figura 1.1). O PSO consiste em um sistema de partículas, onde a

Figura 1.1: Ilustração de um enxame de pássaros [1].

posição de cada partícula representa uma possível solução para o problema. A cada partícula está associada uma velocidade, que é calculada levando em consideração a melhor posição encontrada pela própria partícula (fator cognitivo) e a melhor posição encontrada pelas partículas vizinhas (fator social).

O algoritmo FSS [9] é inspirado no comportamento gregário de peixes (Ver ilustração na Figura 1.2). Nessa abstração, a posição de um peixe representa uma solução e o peso do peixe indica a qualidade da solução. Cada peixe pode olhar ao seu redor, procurando por "alimento". Ao avistar um local com mais alimento que o atual, o peixe se move para esse local mais promissor. Cada peixe tem seu peso variado de acordo com seu sucesso na

Figura 1.2: Ilustração de um cardume de peixes em busca por alimento [2].

movimentação dentro do aquário. A partir do movimento individual do peixe, é gerado um movimento global coletivo para uma região mais promissora. Após isso, mais um movimento global é gerado onde o cardume como um todo se compacta à medida que seus membros ganham peso ou se dispersa à medida que perde peso.

O algoritmo ABC [10] é inspirado no comportamento de abelhas (Ver ilustração na Figura 1.3). Neste modelo, as fontes de alimento representam as soluções. Uma colônia

Figura 1.3: Ilustração de uma colônia de abelhas [3].

consiste de três grupos de abelhas: empregadas, investigadoras e escoteiras. As abelhas empregadas vão para a fonte de alimento e voltam para colmeia para deixar alimento e dançar. A dança das abelhas é um processo de comunicação em equipe, onde ocorre troca de informações a respeito das fontes de alimento. As abelhas investigadoras assistem à dança das empregadas e escolhem uma fonte de alimento dependendo da dança. Quando uma abelha empregada abandona uma fonte de alimento, ela passa a ser uma escoteira e procura por uma nova fonte de alimento.

Alguns pontos positivos das técnicas de inteligência de enxames são: a não necessidade do uso de derivadas da função objetivo e a possibilidade de execução distribuída. Entretanto, podem existir pontos negativos como: a necessidade de ajustar manualmente os parâmetros de configuração dos algoritmos.

1.1 Motivações e Objetivos

Existem grandes desafios relacionados aos problemas de otimização, por exemplo: o efeito "*Curse of Dimensionality*" e o teorema "*Not free lunch*". Outro fator motivador é alta complexidade dos métodos de otimização tradicional, como Newton-Raphson [11] e Quasi-Newton [11].

O efeito "*Curse of Dimensionality*" [12] [13] [14], também conhecido como "Efeito de Hughes" [15], é causado pelo aumento exponencial no volume associado as dimensões adicionais do espaço de busca, ou seja, a dificuldade do problema cresce exponencialmente com o aumento linear do número dimensões. O ideal é desenvolver algoritmos de otimização que funcionem bem em baixa dimensionalidade e que tenham complexidade linear em relação ao número de dimensões. Na prática, quando o número de avaliações da função objetivo cresce exponencialmente com o aumento linear do número de dimensões, implica em uma redução exponencial na velocidade de convergência dos algoritmos de otimização.

O nível de conhecimento em relação ao problema representa outro desafio. Problemas simples podem ser resolvidos obtendo a função inversa. Algumas famílias de problemas possuem métodos específicos para obter soluções satisfatórias. Quando nada é conhecido a respeito do problema, o teorema "*Not free lunch*" [16], estabelece que qualquer método de otimização apresenta desempenho médio igual, pois se um algoritmo apresenta bons resultados para uma classe de problemas, então terá o desempenho reduzido no conjunto de todos os problemas restantes. Isto é importante para definir quais metas devem ser obtidas por um novo algoritmo de otimização, visto que, segundo o teorema não existe um método capaz de adaptar a todos os problemas e obter bons resultados. Um desdobramento do teorema "*Not Free lunch*" demonstra que é possível obter algoritmos melhores que outros, quando é definido um conjunto limitado de problemas e existe algum mecanismo de co-evolução presente no algoritmo [17]. Ciente da implicações do teorema, esta dissertação, realiza estudos com algoritmos que utilizam o conceito de co-evolução em um conjunto limitado de problemas.

O método de Newton-Raphson é uma das formas mais clássica de otimização numérica e, sua principal limitação é otimizar apenas problemas de uma dimensão, além do uso de derivadas da função objetivo (que podem não existir). A cada iteração do algoritmo é produzido um valor $x \in \mathbb{R}$, na tentativa de encontrar a raiz de uma função $h : \mathbb{R} \to \mathbb{R}$, utilizando a Equação 1.1, em que $x(0)$ deve ser um "chute razoável", ou seja, um valor próximo a solução (senão o algoritmo falha). Para otimizar uma função de uma dimensão $f : \mathbb{R} \to \mathbb{R}$, $h(x)$ é substituída por $f'(x)$, conforme a Equação 1.2. A raiz de f' é chamado de ponto crítico de f e também é um ótimo local de f ou ponto de sela. Conseguir encontrar o ponto crítico de f depende do $x(0)$. Este método requer que f' e f'' existam, e que $f''(x) \neq 0$ para todo x, também é possível usar aproximações para f' e f'' utilizando o método de diferenças finitas [18].

$$x(t+1) \leftarrow x(t) - \frac{h(x(t))}{h'(x(t))}. \tag{1.1}$$

$$x(t+1) \leftarrow x(t) - \frac{f'(x(t))}{f''(x(t))}. \tag{1.2}$$

O método Quasi-Newton é uma generalização do método Newton-Raphson para funções multi dimensionais. Para uma função $f : \mathbb{R}^n \to \mathbb{R}$, derivável duas vezes, a solução $\vec{x}$ é gerada utilizando a Equação 1.3. Em que ∇f é chamado de gradiente, é o

vetor das derivadas parciais de primeira ordem de f, conforme a Equação 1.4. A matriz $[Hf(\vec{x}(t))]^{-1}$ é o inverso da matriz hessiana de f. Cada posição da matriz é a derivada parcial de segunda ordem de f, conforme Equação 1.5. O gradiente ∇f pode ser difícil de obter e a matriz $[Hf(\vec{x}(t))]$ é obviamente mais difícil ainda.

$$\vec{x}(t+1) \leftarrow \vec{x}(t) - [Hf(\vec{x}(t))]^{-1} \nabla f(\vec{x}(t)). \tag{1.3}$$

$$\nabla f = \left[\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \cdots, \frac{\partial f}{\partial x_n} \right]. \tag{1.4}$$

$$\mathbf{H}f = \begin{bmatrix} \frac{\partial^2 f}{\partial^2 x_1, x_1} & \frac{\partial^2 f}{\partial^2 x_1, x_2} & \cdots & \frac{\partial^2 f}{\partial^2 x_1, x_n} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial^2 f}{\partial^2 x_n, x_1} & \frac{\partial^2 f}{\partial^2 x_n, x_2} & \cdots & \frac{\partial^2 f}{\partial^2 x_n, x_n} \end{bmatrix}. \tag{1.5}$$

O objetivo geral é estudo de técnicas de otimização numérica contínua baseadas em inteligência enxames, mais especificamente em técnicas baseadas em enxames. Os objetivos específicos são: o estudo de processos de adaptação de parâmetros em técnicas de otimização por enxame de partículas e o estudo de processos de cooperação colaborativa em técnicas de otimização por enxame de partículas.

1.2 Organização do texto

Para um melhor entendimento desta dissertação é recomendada a leitura de forma sequencial. O texto foi organizado da seguinte forma:

- O capítulo 1 apresentou as motivações e objetivos desta dissertação;
- O capítulo 2 apresenta a fundamentação teórica sobre inteligência de enxames;
- O capítulo 3 apresenta a fundamentação teórica sobre multi-enxames cooperativos;
- O capítulo 4 apresenta a proposta do algoritmo Clã PSO Adaptativo;
- O capítulo 5 apresenta os problemas de teste e os resultados das simulações;
- O capítulo 6 apresenta as considerações finais.

1.3 Contribuições

A principal contribuição desta dissertação é o algoritmo de Clã PSO adaptativo. Inspirado nos seguintes trabalhos encontrados na literatura: *Clan Particle Swarm Optimization* [19] e *Adaptive particle swarm optimization* [20]. O algoritmo proposto lança a ideia de que é possível dotar os enxames de um sistema multi enxame com comportamento adaptativo. Os resultados experimentais demonstraram que o algoritmo proposto obteve os seguintes resultados em relação aos algoritmos que lhe serviram de inspiração: (i) maior velocidade de convergência, (ii) maior capacidade de espacar de ótimos locais, (iii) menor tempo de execução, (iv) menor perda desempenho com aumento da dimensionalidade. Além disso, os resultados obtidos nos experimentos demonstraram

ser estatisticamente estáveis nas diversas repetições de cada experimento, ou seja, é possível reproduzir os experimentos.

Uma outra contribuição são os Capítulos 2 e 3. Nestes capítulos há referência completa e didática aos principais algoritmos de inteligencia de enxames. Todos os algoritmos descritos possuem pseudo-código. Além disso, são descritos detalhes dos parâmetros de cada algoritmo, bem como os valores recomendados para serem usados como padrão de cada parâmetro.

Os resultados obtidos no Capítulo 5, estão sendo preparados para publicação em um periódico. Durante o desenvolvimento do mestrado foram geradas duas publicações em conferências internacionais em temas relacionados aos objetivos desta dissertação. Os resumos das publicações encontram-se no Apêndice A.

Capítulo 2

Fundamentos de inteligência de enxames

A natureza oferece vários exemplos de enxames que inspiraram o desenvolvimento de algoritmos inteligentes. Por exemplo: colônia de formigas, cardumes de peixes, agrupamento de aves em voo, entre outros. Nas próximas seções serão introduzidas as principais variações e aplicações dos algoritmos de inteligência de enxames. Esta dissertação explora aplicações em problemas de otimização contínua sem restrições, ou seja, problemas que possuem variáveis contínuas e a única restrição para essas variáveis são os limites mínimo e máximo que cada variável pode assumir. Tendo em vista essas aplicações, tem-se que os principais algoritmos de inteligência de enxames encontrados na literatura especializada para otimização contínua sem restrição são: Otimização por enxame de partículas (PSO), Busca baseada em cardumes (FSS) e Otimização por colônia de abelhas (ABC).

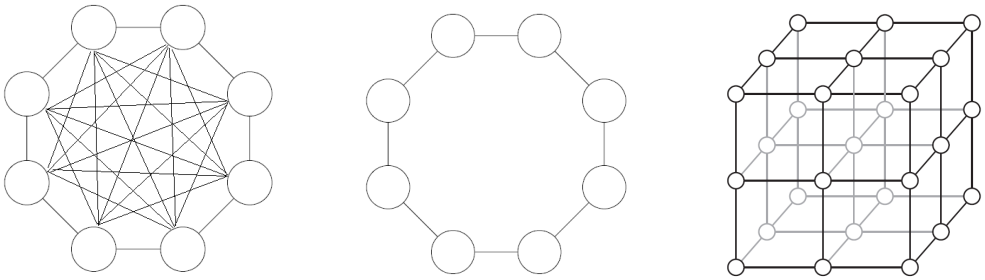
2.1 Algoritmo PSO

Otimização por enxame de partículas (PSO, *Particle Swarm Optimization*) [7] [8] [21] [22] é um algoritmo de otimização inspirada em bandos de pássaros. É considerada uma das técnicas mais importantes de inteligência de enxames [23], sendo utilizada na solução de diversos problemas reais.

Um enxame é formado por uma população de partículas, onde cada partícula é composta de: um vetor de posição ($\vec{x}(t)$) que representa uma possível solução para um problema; o valor obtido a partir da valoração da função objetivo nesta posição $f[\vec{x}(t)]$; uma memória cognitiva que armazena a melhor posição já visitada pela partícula ao longo das iterações do algoritmo, e o valor da função objetivo para essa posição $(\vec{x}_{pbest}(t), f[\vec{x}_{pbest}(t)])$; uma vizinhança, que é entendida como o conjunto de partículas do enxame com as quais é possível haver comunicação; um líder social que é a melhor partícula na vizinhança $\vec{x}_{nbest}(t)$, e o valor da função objetivo para essa posição $(\vec{x}_{nbest}(t), f[\vec{x}_{nbest}(t)])$; e um vetor de velocidade ($\vec{v}(t)$).

A vizinhança da partícula é definida pela topologia de comunicação entre as partículas. A topologia de comunicação utiliza apenas os índices das partículas dentro do enxame, ou seja, normalmente não está relacionado à disposição das partículas no espaço de busca. Entre as topologias já propostas na literatura [24] [25] [26], podem ser citadas: topologia global, onde cada partícula se comunica com todas as outras (Figura 2.1(a)); topologia em anel, onde cada partícula se comunica com seus vizinhos adjacentes (Figura

2.1(b)); topologia de Von Neumann, onde as partículas são organizadas em forma de cristal (Figura 2.1(c)).



(a) (b) (c)

Figura 2.1: Ilustração das topologias de comunicação mais utilizads em PSO: (a) Global, (b) Anel e (c) Von Newmann.

A quantidade de partículas no enxame normalmente é fixa ao longo das iterações, e geralmente usa-se 30 partículas. Algumas publicações demonstraram que variações do tamanho do enxame entre 20 e 100 partículas não apresentaram diferença significativa no desempenho do algoritmo sobre um conjunto de problemas de teste bastante utilizado na literatura de inteligência enxames [26]. De fato, para a maioria das aplicações, o algoritmo não é muito sensível ao tamanho da população [27].

Durante o processo de busca, cada partícula pertencente ao enxame é atualizada a cada iteração utilizando a Equação (2.1) para ajustar a velocidade e a Equação (2.2) para ajustar a posição.

$$\vec{v}(t+1) = \omega\vec{v}(t) + U_1(0,1)c_1\left[\vec{x}_{pbest}(t) - \vec{x}(t)\right] + U_2(0,1)c_2\left[\vec{x}_{nbest}(t) - \vec{x}(t)\right], \quad (2.1)$$

$$\vec{x}(t+1) = \vec{x}(t) + \vec{v}(t), \quad (2.2)$$

em que $U_1(0,1)$ e $U_2(0,1)$ são números aleatórios obtidos de uma distribuição uniforme entre 0 e 1. É importante ressaltar que para cada partícula, em cada dimensão, um par de números aleatórios diferentes é gerado. ω é o coeficiente de inércia, c_1 é o coeficiente de aceleração cognitivo e c_2 é o coeficiente de aceleração social.

Inicialmente, a posição $\vec{x}(t)$ e a velocidade $\vec{v}(t)$ são gerados aleatoriamente para cada partícula. Na abordagem mais usada, o valor dos coeficientes de aceleração c_1 e c_2 é igual a $2,05$ e o coeficiente de inércia ω decresce linearmente de $\omega_{max} = 0,9$ até $\omega_{min} = 0,4$ em função do número de iterações [28]. O cálculo do decrescimento linear é realizado seguindo a Equação (2.3).

$$\omega = \omega_{max} - \left[(\omega_{max} - \omega_{min})\frac{g_{atual}}{g_{final}}\right], \quad (2.3)$$

em que g_{atual} é iteração atual e g_{final} é total de iterações a serem realizadas.

A estratégia de decrescimento linear do coeficiente de inércia permite balancear as capacidades de busca em amplitude e busca em profundidade. É desejado que a capacidade de busca em amplitude seja maior durante as iterações iniciais, quando provavelmente o algoritmo ainda não encontrou uma região próxima ao ótimo da função objetivo, e que a capacidade de busca em profundidade seja maior durante as iterações

finais, quando provavelmente o algoritmo já encontrou uma região próxima ao ótimo da função objetivo e está refinando as soluções encontradas.

Dependendo da abordagem utilizada, depois de calculada a velocidade, é necessário assegurar que esta está dentro de um limite máximo. Isto é feito para evitar o estado de explosão, ou seja, que a velocidade atinja valores maiores que a amplitude do espaço de busca. Uma boa prática é limitar o valor absoluto da velocidade a 20% do intervalo de busca [28][29]. Outro motivo para se controlar a velocidade é ter um controle da variação máxima de posição entre duas iterações consecutivas.

Após calculada a posição, também se faz necessário verificar se a partícula está dentro do espaço de busca factível. Um estratégia simples é deixar a partícula entrar no espaço infactível sem calcular o valor da função objetivo $f[\vec{x}(t)]$. O que se espera é que a partícula retorne ao espaço factível, uma vez que seu valor para função objetivo $f[\vec{x}(t)]$ permanece inalterado e existe uma atração da partícula pela posição armazenada na sua memória cognitiva. Algumas tópicos relevantes sobre o tratamento de soluções infactíveis podem ser encontrados em [30].

2.1.1 Pseudo código para algoritmo PSO

O pseudo código do algoritmo de PSO é apresentado no Algoritmo 1. Na linha 5, a localização da melhor partícula na vizinha, depende da topologia de comunicação escolhida.

Algoritmo 1: Pseudo código do algoritmo PSO.

```
1  iniciar as partículas do enxame com posição e velocidade aleatórias;
2  iniciar parâmetro de inércia com valor máximo;
3  enquanto não for iteração final faça
4      para cada partícula do enxame faça
5          Localizar a melhor partícula na vizinhança ;
6          Atualizar a velocidade utilizando a Equação 2.1 ;
7          Atualizar a posição utilizando a Equação 2.2 ;
8          Avaliar posição utilizando a função objetivo do problema ;
9          se posição atual melhor que memória cognitiva então
10             Atualizar a memória cognitiva ;
11         fim
12     fim
13     Atualizar parâmetro de inércia utilizando a Equação 2.3 ;
14 fim
15 retorna A melhor memória cognitiva do enxame
```

2.2 Algoritmo PSO Constricted

Uma análise dos parâmetros do algoritmo de PSO foi realizada por Clerc com a finalidade de investigar as propriedades de convergência e estabilidade [31] [32]. Como resultado da análise, foi determinado um "fator de constrição". Quando este coeficiente é aplicado à equação de velocidade do PSO, está garantida a convergência do enxame para uma região de ótimo local após um número suficiente de iterações e o ajuste da velocidade das partículas do enxame de forma estável, evitando assim que ocorra um estado de explosão. Neste caso, não se faz mais necessário impor limites de velocidade para as partículas do enxame.

O fator de constrição é obtido a partir das Equações (2.4) e (2.5). A equação de atualização de velocidade modificada para o algoritmo de PSO está apresentada na Equação (2.6).

$$\varphi = c_1 + c_2, \tag{2.4}$$

$$\chi = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|}, \tag{2.5}$$

$$\vec{v}(t+1) = \chi \left\{ \vec{v}(t) + U_1(0,1) \cdot c_1 \cdot \left[\vec{x}_{pbest}(t) - \vec{x}(t)\right] + U_2(0,1) \cdot c_2 \cdot \left[\vec{x}_{nbest}(t) - \vec{x}(t)\right] \right\}. \tag{2.6}$$

É demonstrado que só existe garantia de convergência para uma região de ótimo local se $\varphi > 4$. Caso $\varphi \leq 4$, as partículas se movem lentamente em espiral e não existe garantia de convergência para uma região de ótimo local. Para $c_1 = c_2 = 2,05$, tem-se que $\varphi = 4,1$ e $\chi \approx 0,7298$. É importante notar que as análises realizadas por Clerc foram feitas mantendo ω constante com valor igual a 1.

2.2.1 Pseudo código para algoritmo PSO Constricted

O pseudo código do algoritmo de PSO Constricted é apresentado no Algoritmo 2. Na linha 5, a localização da melhor partícula na vizinha, depende da topologia de comunicação escolhida.

Algoritmo 2: Pseudo código do algoritmo PSO Constricted.

```
1  iniciar as partículas do enxame com posição e velocidade aleatórias;
2  iniciar parâmetros utilizando as Equações 2.4 e 2.5 ;
3  enquanto não for iteração final faça
4      para cada partícula do enxame faça
5          Localizar a melhor partícula na vizinhança ;
6          Atualizar a velocidade utilizando a Equação 2.6 ;
7          Atualizar a posição utilizando a Equação 2.2 ;
8          Avaliar posição utilizando a função objetivo do problema ;
9          se posição atual melhor que memória cognitiva então
10             Atualizar a memória cognitiva ;
11         fim
12     fim
13 fim
14 retorna A melhor memória cognitiva do enxame
```

2.3 Algoritmo GCPSO

Existe um problema no PSO original que pode levar o enxame a convergir prematuramente para uma região de ótimo local. Isso pode ocorrer quando uma partícula se torna a melhor partícula do enxame. Neste momento, $\vec{x}(t) = \vec{x}_{pbest}(t) = \vec{x}_{nbest}(t)$. Sendo assim, a melhor partícula do enxame depende apenas da sua velocidade anterior para se mover, que decresce rapidamente com as iterações, até a melhor partícula do enxame parar totalmente. Como as outras partículas do enxame são atraídas pela melhor partícula, isso pode causar a convergência prematura para uma região de ótimo local.

Na proposta PSO com garantia de convergência (GCPSO, *Guaranteed Convergence PSO*) [33][34] é sugerida uma modificação na equação de velocidade apenas para a melhor partícula do enxame. Isto é feito para evitar que a velocidade da melhor partícula não se torne nula prematuramente. A equação de velocidade sugerida para atualização da melhor partícula no GCPSO está apresentada na Equação (2.7).

$$\vec{v}(t+1) = -\vec{x}(t) + \vec{x}_{pbest}(t) + \omega \cdot \vec{v}(t) + \rho(t) \cdot U(-1,1), \tag{2.7}$$

em que ρ é um escalar que tem valor inicial 1 e $U(-1,1)$ é um número aleatório obtido de uma distribuição uniforme entre -1 e 1.

O escalar ρ é atualizado em função do número de sucessos e falhas da melhor partícula, conforme a Equação (2.8).

$$\rho(t+1) = \begin{cases} 2,0 \cdot \rho(t) & \text{se \#sucessos} > s_c, \\ 0,5 \cdot \rho(t) & \text{se \#falhas} > f_c, \\ \rho(t) & \text{caso contrário,} \end{cases} \tag{2.8}$$

em que $s_c = 15$ e $f_c = 5$ por recomendação dos autores.

Os contadores #sucessos e #falhas são zerados quando a melhor partícula do enxame muda. A atualização dos contadores é feita da seguinte forma: quando ocorre um sucesso, o contador #sucessos é incrementado e o #falhas zerado; quando ocorre uma falha, o contador #falhas é incrementado e o #sucessos é zerado.

2.3.1 Pseudo código para algoritmo GCPSO

O pseudo código do algoritmo de PSO Constricted é apresentado no Algoritmo 3. Na linha 5, a localização da melhor partícula na vizinha, depende da topologia de comunicação escolhida. Na linha 21, caso ocorra a modificação do valor do parâmetro ρ os contadores #sucessos e #falhas devem ser reiniciados.

Algoritmo 3: Pseudo código do algoritmo GCPSO.

```
1  iniciar as partículas do enxame com posição e velocidade aleatórias;
2  iniciar parâmetros utilizando as Equações 2.4 e 2.5 ;
3  enquanto não for iteração final faça
4      para cada partícula do enxame faça
5          Localizar a melhor partícula na vizinhança ;
6          se partícula é a melhor partícula do enxame então
7              Atualizar a velocidade utilizando a Equação 2.7 ;
8          senão
9              Atualizar a velocidade utilizando a Equação 2.6 ;
10         fim
11         Atualizar a posição utilizando a Equação 2.2 ;
12         Avaliar posição utilizando a função objetivo do problema ;
13         se posição atual melhor que memória cognitiva então
14             Atualizar a memória cognitiva ;
15             Incrementar #sucessos ;
16             Reiniciar #falhas ;
17         senão
18             Incrementar #falhas ;
19             Reiniciar #sucessos ;
20         fim
21         Atualizar parâmetro ρ utilizando a Equação 2.8 ;
22     fim
23 fim
24 retorna A melhor memória cognitiva do enxame
```

2.4 Algoritmo APSO

A proposta do PSO adaptativo (APSO, *Adaptive PSO*) se deu quando Zhan et al. [20] identificaram as seguintes deficiências no PSO original: a baixa velocidade de convergência e a capacidade limitada de escapar de ótimos locais. Então, obter uma alta velocidade de convergência e escapar de ótimos locais são principais objetivos do APSO. Para obter sucesso em seus objetivos, o APSO apresenta um esquema de adaptação sistemática de parâmetros e uma estratégia de aprendizado elitista.

Os passos que são executados a cada geração do APSO consistem basicamente em:

- Avaliar a distribuição da população com um procedimento que faz a estimativa do estado evolucionário considerando uma métrica proposta pelos autores chamada de fator evolucionário;
- Classificar a população em um dos possíveis estados evolucionários;
- Adaptar os parâmetros de aceleração, melhorando a velocidade de convergência;
- Executar uma estratégia de aprendizado elitista para escapar de ótimos locais;
- Adaptar o parâmetro de inércia, visando melhorar a eficiência da busca.

Os detalhes de cada passo são apresentados nas subseções seguintes.

2.4.1 Estimativa do estado evolucionário

Para controlar os parâmetros do APSO, primeiro é preciso estimar o estado evolucionário do enxame, pois as características da população podem mudar com o tempo. O estado evolucionário é determinado pelo valor do fator evolucionário. O cálculo da estimativa do estado evolucionário é realizado em dois passos.

Primeiramente, calcula-se a distância média de cada partícula para todas as outras utilizando a Equação (2.9);

$$d_i = \frac{1}{N-1} \sum_{j=1, j\neq i}^{N} \sqrt{\sum_{k=1}^{D} (x_i^k - x_j^k)^2}, \tag{2.9}$$

em que N é o número total de partículas no enxame e D é o número de dimensões do problema.

Posteriormente, calcula-se o fator evolucionário utilizando a Equação (2.10).

$$f_{evol} = \frac{d_g - d_{min}}{d_{max} - d_{min}} \in [0,1], \tag{2.10}$$

em que f_{evol} é o fator evolucionário, d_g é a distância média entre a melhor partícula do enxame e as outras partículas, d_{min} é a menor distância média considerando todas partículas do enxame e d_{max} é a maior distância média considerando todas partículas do enxame. Devido a normalização do fator evolucionário, é necessário o uso de pelos menos 3 partículas no enxame para evitar $d_{min} = d_{max}$.

O fator evolucionário é um escalar entre "0" e "1" que representa a proximidade entre a melhor partícula do enxame e todas as outras partículas do enxame. Quanto mais próximo de "0", significa que as partículas do enxame estão muito próximas da melhor partícula do enxame. Quanto mais próximo de "1", significa que as partículas do enxame estão muito distantes da melhor partícula do enxame.

2.4.2 Classificação do enxame em um dos estados evolucionários

A classificação do enxame em um dos estados evolucionários pode ser feita de várias formas. Uma possibilidade é utilizar intervalos fixos para cada estado evolucionário. Mas na proposta original do APSO, é utilizado um processo de classificação *fuzzy*. Funções de pertinência *fuzzy* são utilizadas para classificar o fator evolucionário. Existem vários estados possíveis, a saber: convergência, busca em profundidade, busca em amplitude e escape.

Estado de convergência: ocorre quando o valor de f_{evol} é próximo do mínimo, ou seja, neste caso a distância da melhor partícula de enxame para as demais partículas do enxame é mínima. Neste caso, o algoritmo está refinando as boas soluções encontradas em uma região ótima. Calcula-se o valor *fuzzy* do estado utilizando a Equação (2.11).

$$E_{converge}(f_{evol}) = \begin{cases} 1, & \text{se } 0,0 \leq f_{evol} \leq 0,1, \\ -5f_{evol} + 1,5, & \text{se } 0,1 < f_{evol} \leq 0,3, \\ 0, & \text{se } 0,3 < f_{evol} \leq 1,0. \end{cases} \tag{2.11}$$

Estado de busca em profundidade: ocorre quando o valor de f_{evol} é pequeno, ou seja, neste caso a distância da melhor partícula de enxame para as demais partículas do enxame

é pequena. O algoritmo está buscando boas soluções em uma região limitada. Calcula-se o valor *fuzzy* do estado utilizando a Equação (2.12).

$$E_{profundidade}(f_{evol}) = \begin{cases} 0, & \text{se } 0,0 \leq f_{evol} \leq 0,2, \\ 10f_{evol} - 2, & \text{se } 0,2 < f_{evol} \leq 0,3, \\ 1, & \text{se } 0,3 < f_{evol} \leq 0,4, \\ -5f_{evol} + 3, & \text{se } 0,4 < f_{evol} \leq 0,6, \\ 0, & \text{se } 0,6 < f_{evol} \leq 1,0. \end{cases} \tag{2.12}$$

Estado de busca em amplitude: ocorre quando o valor de f_{evol} está compreendido entre médio e alto, ou seja, neste caso a distância da melhor partícula de enxame para as demais partículas do enxame está entre média e grande. O algoritmo está buscando por uma região ótima. Calcula-se o valor *fuzzy* do estado utilizando a Equação (2.13).

$$E_{amplitude}(f_{evol}) = \begin{cases} 0, & \text{se } 0,0 \leq f_{evol} \leq 0,4, \\ 5f_{evol} - 2, & \text{se } 0,4 < f_{evol} \leq 0,6, \\ 1, & \text{se } 0,6 < f_{evol} \leq 0,7, \\ -10f_{evol} + 8, & \text{se } 0,7 < f_{evol} \leq 0,8, \\ 0, & \text{se } 0,8 < f_{evol} \leq 1,0. \end{cases} \tag{2.13}$$

Estado de escape: ocorre quando o valor de f_{evol} é próximo do máximo, ou seja, neste caso a distância da melhor partícula de enxame para as demais partículas do enxame é grande. A melhor partícula do enxame escapou de uma região de ótimo local para uma nova região de ótimo mais promissor. Calcula-se o valor *fuzzy* do estado utilizando a Equação (2.14).

$$E_{escape}(f_{evol}) = \begin{cases} 0, & \text{se } 0,0 \leq f_{evol} \leq 0,7, \\ 5f_{evol} - 3,5, & \text{se } 0,7 < f_{evol} \leq 0,9, \\ 1, & \text{se } 0,9 < f_{evol} \leq 1,0. \end{cases} \tag{2.14}$$

Na Figura 2.2, são apresentadas as funções de pertinência *fuzzy* utilizadas na proposta original do APSO [20].

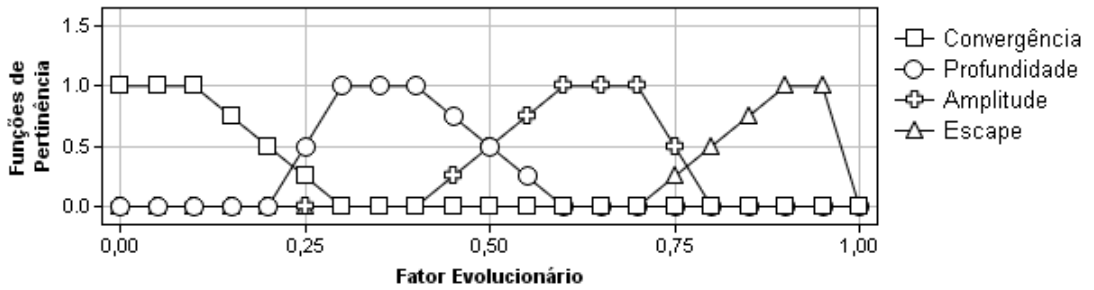


Figura 2.2: Funções de pertinência *fuzzy* utilizadas no APSO.

A cada iteração, os valores para cada estado são zerados. Após o cálculo dos estados, o que obter maior valor *fuzzy* é selecionado como o estado evolucionário atual do enxame.

2.4.3 Adaptação de parâmetros de aceleração

No APSO, os coeficientes de aceleração c_1 e c_2 são iniciados com valor igual a $2,0$ e, durante a execução do APSO, são controlados de forma adaptativa em função do estado evolucionário conforme a Tabela 2.1.

Tabela 2.1: Estratégias de controle dos coeficientes de aceleração.

Estado evolucionário	c_1	c_2
Busca em amplitude	incrementar	decrementar
Busca em profundidade	incrementar levemente	decrementar levemente
Convergência	incrementar levemente	incrementar levemente
Escape	decrementar	incrementar

No estado de busca em amplitude, o valor de c_1 é incrementado e c_2 é decrementado. A importância do estado de exploração é descobrir o maior número possível de ótimos. Essa estratégia ajuda as partículas a explorar individualmente e obter suas próprias posições $\vec{x}_{pbest}(t)$, ao invés de se associar a uma suposta melhor partícula que pode estar em um ótimo local.

No estado de busca em profundidade, o valor de c_1 é incrementado levemente e c_2 é decrementado levemente. Neste estado, as partículas fazem uso da informação local e se agrupam ao redor de possíveis ótimos locais indicados por suas posições $\vec{x}_{pbest}(t)$. Portanto, a ênfase é a busca em profundidade em volta do $\vec{x}_{pbest}(t)$. Espera-se que o estado de busca em profundidade ocorra depois de um estado de busca em amplitude e antes de um estado de convergência. Os ajustes de c_1 e c_2 devem conduzir de modo apropriado o algoritmo do estado de busca em amplitude para o estado de convergência.

No estado de convergência, o valor de c_1 é incrementado levemente e c_2 é incrementado levemente. O enxame possivelmente encontrou uma região que é um ótimo global. A influência social deve ser enfatizada para atrair as partículas para a melhor partícula da vizinhança. A influência cognitiva deveria ser diminuída para promover uma rápida convergência, mas durante os testes do APSO, foi verificado que esse tipo de ajuste causa convergência prematura, pois os parâmetros rapidamente saturam para os limites mínimos e máximos. A solução encontrada foi incrementar levemente os dois coeficientes de aceleração.

No estado de escape, o valor de c_1 é decrementado e c_2 é incrementado. Quando a melhor partícula deixa um ótimo local para outro melhor, é como se estivesse deixando o agrupamento principal do enxame que estava durante o estado de convergência. Após a descoberta dessa nova região, as outras partículas devem seguir a melhor partícula do enxame. Sendo assim, é dada uma maior importância ao novo líder social em detrimento da experiência cognitiva de cada partícula. Isso faz com que a partícula seja guiada para uma região promissora, caso a busca em profundidade da partícula não esteja obtendo progresso significativo.

O passo de incremento ou decremento dos coeficientes de aceleração é chamado de taxa de aceleração (δ). No APSO, o valor da taxa de aceleração (δ) é obtido de uma distribuição uniforme entre 0,05 e 0,10. Nas estratégias onde é necessário um incremento ou decremento leve, usa-se $0,5 \cdot \delta$.

No decorrer do processo de adaptação de parâmetros, os coeficientes de aceleração podem atingir valores extremante grandes ou pequenos, provocando instabilidades no processo de busca. Para solucionar este problema é necessário normalizar os valores

dos coeficientes de aceleração. Foi determinado que $c_{min} = 1,5$ e $c_{max} = 2,5$ são respectivamente os valores limites mínimo e máximo para os coeficientes de aceleração. Quando a soma dos coeficientes de aceleração c_1 e c_2 é maior que $c_{min} + c_{max}$, é realizada uma normalização conforme a Equação (2.15).

$$c_i = \frac{c_i(c_{min} + c_{max})}{c_1 + c_2}. \tag{2.15}$$

2.4.4 Estratégia de aprendizado elitista

Esta estratégia foi proposta para o APSO e visa melhorar a capacidade de busca global do algoritmo provocando um estado de escape. Foi concebida para ser aplicada apenas na melhor partícula do enxame para ajudá-la a escapar de um ótimo local durante o estado de convergência.

Diferentemente das outras partículas, o líder global não tem a quem seguir. Portanto, tem de encontrar um meio de melhorar por si própria. Uma perturbação é provocada para ajudar o líder a se movimentar para uma região possivelmente melhor. Se a nova região é melhor do que a atual, então o resto do enxame irá seguir o líder rapidamente e convergir para esta nova região.

É escolhida uma dimensão do $\vec{x}_{pbest}(t)$ da melhor partícula do enxame, que será denotada por $x_{pbestd}(t)$. Apenas uma dimensão é escolhida porque geralmente o ótimo global compartilha uma parte de sua estrutura com ótimos locais. Todas as dimensões têm a mesma probabilidade de serem escolhidas. Então, é gerada uma perturbação Gaussiana conforme a Equação (2.16).

$$x_{pbestd}(t+1) = x_{pbestd}(t) + (X^d_{max} - X^d_{min})G(\mu, \sigma^2), \tag{2.16}$$

em que (X^d_{min}, X^d_{max}) são os limites do espaço de busca, $G(\mu, \sigma^2)$ é um número aleatório obtido de um distribuição gaussiana com média (μ) zero e desvio padrão (σ).

A nova posição só é aceita se for melhor do que a atual posição da melhor partícula do enxame. Caso contrário, esta solução será usada para substituir a pior partícula do enxame.

No APSO, o desvio padrão da distribuição gaussiana utilizada na Equação (2.16) é chamado de taxa de aprendizado elitista. É sugerido decrescer linearmente o valor da taxa de aprendizado elitista ao longo das iterações, conforme a Equação (2.17).

$$\sigma = \sigma_{max} - (\sigma_{max} - \sigma_{min})\frac{g_{atual}}{g_{final}}, \tag{2.17}$$

em que σ_{max} e σ_{min} são os limites máximo e mínimo para a taxa de aprendizado elitista, g_{atual} é a iteração atual e g_{final} é o total de iterações a serem realizadas. Empiricamente, foi demonstrado que $\sigma_{max} = 1,0$ e $\sigma_{min} = 0,1$ geram boas soluções. Uma alta taxa de aprendizado elitista provê ao líder a possibilidade de deixar um ótimo local, uma baixa taxa de aprendizado elitista permite que o líder refine a melhor solução encontrada até então.

2.4.5 Adaptação do parâmetro inércia

O ajuste do fator de inércia ω é usado para balancear a capacidade de busca em amplitude e busca em profundidade. No APSO, o fator de inércia ω é obtido em função do fator evolucionário utilizadando a Equação (2.18).

$$\omega(f) = \frac{1}{1+1,5e^{-2,6f}} \in [0,4;0,9], \quad \forall f \in [0,1]. \tag{2.18}$$

2.4.6 Pseudo código para algoritmo APSO

O pseudo código do algoritmo APSO é apresentado no Algoritmo 4. Na linha 8, a classificação do enxame em um estado evolucionário é feita utilizando o estado que apresenta o maior valor de pertinência.

Algoritmo 4: Pseudo código do algoritmo APSO.

```
1  iniciar as partículas do enxame com posição e velocidade aleatórias;
2  enquanto não for iteração final faça
3      para cada partícula do enxame faça
4          Calcular distância média utilizando a Equação 2.9 ;
5      fim
6      Calcular fator evolucionário utilizando a Equação 2.10 ;
7      Calcular pertinências utilizando as Equações 2.11,2.12,2.13 e 2.14 ;
8      Classificar enxame em um estado evolucionário ;
9      Adaptar os coeficientes de aceleração conforme Tabela 2.1 ;
10     Normalizar os coeficientes de aceleração utilizando a Equação 2.15 ;
11     Atualizar parâmetro de inércia utilizando Equação 2.18 ;
12     se classificado em estado de convergência então
13         Gerar uma nova posição utilizando as Equações 2.16 e 2.17 ;
14         se nova a posição é melhor que a melhor memória cognitiva então
15             Atualizar a posição e memória cognitiva da melhor partícula;
16         senão
17             Atualizar a posição e memória cognitiva da pior partícula;
18         fim
19     fim
20     para cada partícula do enxame faça
21         Localizar a melhor partícula na vizinhança ;
22         Atualizar a velocidade utilizando a Equação 2.1 ;
23         Atualizar a posição utilizando a Equação 2.2 ;
24         Avaliar posição utilizando a função objetivo do problema ;
25         se posição atual melhor que memória cognitiva então
26             Atualizar a memória cognitiva ;
27         fim
28     fim
29 fim
30 retorna A melhor memória cognitiva do enxame
```

2.5 Variações do algoritmo PSO

Existem outros artigos e estudos que serviram de fundamentação teórica para o desenvolvimento desta dissertação. Para interessados no assunto, é recomendada a leitura das seguintes referências:

1. Soluções para lidar com problemas de otimização discretos [35];
2. Soluções para lidar com problemas de otimização dinâmicos [36];
3. Soluções para lidar com problemas de otimização multi-objetivo [37] [38] [39] [40] [41] [42] [43];
4. Soluções que utilizam operadores genéticos de seleção [44];
5. Análise de convergência e estabilidade [45];
6. Abordagem para clusterização de partículas [46];
7. Comparação entre o coeficiente de inércia e o fator de constrição [47];
8. Análise do fator de inércia com adaptação *fuzzy* [27];
9. Estudo da influência da topologia de comunicação no desempenho do PSO [48][49];
10. Apresentação de uma nova topologia com anéis giratórios [50];
11. Apresentação do conceito de partículas quânticas [51] [52] [53] [54];
12. PSO-TVAC: Variação do valor dos coeficientes de aceleração ao longo das iterações [55].
13. CLPSO: Utilização do histórico da memória cognitiva no ajuste de velocidade de partículas [56].

2.6 Aplicações do algoritmo PSO

O algoritmo de PSO e suas variações foram aplicados com sucesso em diversas aplicações reais. Abaixo segue uma lista de algumas aplicações:

1. Treinamento de redes neurais artificiais [57] [58] [59] [60] [61] [62] [63] [64] [65] [66] [67] [68] [69] [70] [71] [72] ;
2. Problemas de engenharia elétrica e eletrônica [73] [74] [75] [76] [77] [78] [79] [80] [81] [82] [83] [84] [85] [86] [87] [88] [89] [90] [91] [92] [93] [94] [95] [96] [97] [98] [99] [100] [101] ;
3. Problemas de engenharia com múltiplos objetivos como: demanda de carga elétrica e agendamento de tarefas [102] [103] [104] ;
4. Problemas aeroespaciais [105] ;
5. Projeto de antenas [106] [107] [108] [109]; [110]; ;
6. Aproximação polinomial [111] ;
7. Seleção de características em problemas de classificação [112] [113] ;
8. Engenharia mecânica [114] ;

9. Mineração de dados [115] ;
10. Otimização de áreas e volumes [116] ;
11. Planejamento de tarefas [117] [118] [119] [120] [121] [122] ;
12. Programação não-linear [123] ;
13. Previsão de séries temporais [124] ;
14. Tratamento de imagens [125] [126] [127] ;
15. Redes de sensores sem fio [128] .

2.7 Algoritmo FSS

Busca baseada em cardumes (FSS, *Fish School Search*) [9] [129] [130] é um algoritmo de busca e otimização inspirado no comportamento gregário de peixes. Em funções monomodais os resultados do FSS são comparáveis aos do PSO básico, mas em funções multimodais as versões mais recentes do FSS produzem resultados melhores do que as variações mais básicas do PSO [131].

No FSS, cada peixe possui um peso que representa o sucesso do peixe durante o processo de busca. Um conjunto de peixes forma um cardume. A posição de cada peixe representa uma possível solução. A função objetivo é mapeada como uma quantidade de alimento disponível. O aquário é mapeado como a região factível da função objetivo. Em problemas de minimização, a quantidade de alimento disponível é inversamente proporcional ao valor da função objetivo no ponto de avaliação.

O algoritmo é composto de operadores que são executados de forma sequencial sobre o cardume durante cada iteração: operador de movimento individual, que faz busca local; operador de alimentação, que funciona como indicador de sucesso ou fracasso do processo de busca local; operador de movimento coletivo instintivo, que gera um deslocamento em todo o cardume; operador de movimento coletivo volitivo, que regula a granularidade da busca do algoritmo. Os operadores são detalhados nas subseções seguintes.

2.7.1 Operador de Movimento Individual

Os peixes são atraídos pelo alimento que pode estar espalhado no aquário em diversas concentrações. Para encontrar alimento, cada peixe do cardume realiza movimentos individuais conforme a Equação (2.19).

$$\vec{x}(t+1) = \vec{x}(t) + U(-1,1)S_{individual}(t), \tag{2.19}$$

em que $\vec{x}(t)$ é a posição do peixe no instante t, $U(-1,1)$ é um vetor gerado por uma distribuição uniforme entre 1 e -1. O parâmetro $S_{individual}$ é uma porcentagem da amplitude de busca em determinada dimensão.

É sugerido que o parâmetro $S_{individual}$ decresça linearmente ao longo das iterações conforme a Equação (2.20) com a finalidade de obter uma maior capacidade de busca em profundidade no fim do processo de busca.

$$S_{individual}(t) = S_{individual\,inicial} - (S_{individual\,inicial} - S_{individual\,final})\frac{g_{atual}}{g_{final}}, \tag{2.20}$$

em que g_{atual} indica a iteração atual e g_{final} é o número total iterações a serem realizadas, $S_{individual\,inicial}$ é o passo individual no início da execução do algoritmo e $S_{individual\,final}$ é o passo individual no final da execução do algoritmo. Usualmente $S_{individual\,inicial} = 10\%$ e $S_{individual\,final} = 0,001\%$ do tamanho do aquário em uma determinada dimensão [9].

2.7.2 Variação de posição e variação da função objetivo

Para calcular os próximos operadores é necessário calcular para cada peixe: a variação de posição utilizando a Equação (2.21); e a variação da função objetivo utilizando a Equação (2.22).

$$\Delta\vec{x} = \vec{x}(t) - \vec{x}(t-1), \tag{2.21}$$

$$\Delta f = f[\vec{x}(t)] - f[\vec{x}(t-1)]. \tag{2.22}$$

2.7.3 Operador de Alimentação

O peixe pode ganhar ou perder peso dependendo do sucesso ou falha de seu movimento individual conforme a Equação (2.23).

$$W_i(t+1) = W_i(t) + \frac{\Delta f}{max(|\Delta f|)}, \tag{2.23}$$

em que W_i é o peso do peixe e $max(|\Delta f|)$ é a maior variação da função objetivo em absoluto do cardume na iteração atual. O valor inicial de W_i é o ponto médio de uma escala de valores que tem valor mínimo W_{min} e valor máximo W_{max}. Durante as iterações, o peso do peixe é limitado a essa escala de valores. Usualmente $W_{min} = 1$ e $W_{max} = 5000$ [9].

2.7.4 Operador de Movimento Coletivo Instintivo

Após a execução do operador individual, alguns peixes obtiveram sucesso na busca por alimento e outros não. É esperado que os peixes que obtiveram sucesso na busca por alimento influenciem o movimento de todo o cardume para uma região mais rica em alimento. Então, é gerado um movimento global na direção em que provavelmente existe uma maior concentração de alimento, a partir do movimento dos peixes que obtiveram sucesso na busca por alimento.

A direção obtida pelo cardume é calculada pela Equação (2.24).

$$\vec{I}(t) = \frac{\sum_{i=1}^{N} \vec{\Delta x_i} \Delta f_i}{\sum_{i=1}^{N} \Delta f_i}. \tag{2.24}$$

Em seguida, todos os peixes do cardume têm suas posições atualizadas por este movimento conforme a Equação (2.25).

$$\vec{x}(t+1) = \vec{x}(t) + \vec{I}(t). \tag{2.25}$$

2.7.5 Operador de Movimento Coletivo Volitivo

A variação de peso do cardume gera outro movimento global capaz de expandir ou contrair o cardume. Quando o cardume está ganhando peso significa que está tendo sucesso na busca. Então, é esperado que o cardume se compacte para aumentar a capacidade de busca em profundidade. Quando o cardume está perdendo peso significa que está tendo fracasso na busca. Então, é esperado que o cardume se disperse para aumentar a capacidade de busca em amplitude.

Antes de realizar a expansão ou contração do cardume, é necessário calcular o centro de massa do cardume conforme a Equação (2.26).

$$\vec{B}(t) = \frac{\sum_{i=1}^{N} \vec{x}_i W_i(t)}{\sum_{i=1}^{N} W_i(t)}. \tag{2.26}$$

Para realizar a expansão do cardume, usa-se a Equação (2.27). Para realizar a contração do cardume, usa-se a Equação (2.28).

$$\vec{x}(t+1) = \vec{x}(t) + S_{volitivo} U(0,1) \frac{(\vec{x}(t) - \vec{B}(t))}{DE(\vec{x}(t), \vec{B}(t))}, \tag{2.27}$$

$$\vec{x}(t+1) = \vec{x}(t) - S_{volitivo} U(0,1) \frac{(\vec{x}(t) - \vec{B}(t))}{DE(\vec{x}(t), \vec{B}(t))}, \tag{2.28}$$

em que $U(0,1)$ é uma distribuição aleatória uniforme entre 0 e 1, DE é uma função que calcula a distância euclidiana entre duas posições, $S_{volitivo}$ é o parâmetro do operador de movimento coletivo volitivo.

É sugerido que o parâmetro $S_{volitivo}$ decresça linearmente ao longo das iterações conforme a Equação (2.29) com a finalidade de obter uma maior capacidade de busca em profundidade no fim do processo de busca.

$$S_{volitivo}(t) = S_{volitivo\,inicial} - (S_{volitivo\,inicial} - S_{volitivo\,final}) \frac{g_{atual}}{g_{final}}, \tag{2.29}$$

em que g_{atual} é iteração atual e g_{final} é o total iterações a serem realizadas. Empiricamente a relação $S_{volitivo} = 2 \cdot S_{individual}$ produz bons resultados.

Após a atualização da posição de todos os peixes, é necessário recalcular o valor da função objetivo para que o cardume fique preparado para a próxima iteração do processo de busca.

2.7.6 Pseudo código para algoritmo FSS

O pseudo código do algoritmo FSS é apresentado no Algoritmo 5. Nas linhas 5 e 31, são feitas avaliações da função objetivo do problema. Portanto quando o algoritmo FSS for utilizado em comparações de desempenho, é importante ajustar a quantidade de peixes no cardume ou o números de iterações, de forma que a quantidade de avaliações da função objetiva seja igual para ambos algoritmos utilizados na comparação.

Algoritmo 5: Pseudo código do algoritmo FSS.

```
1  iniciar os peixes em posições aleatórias;
2  enquanto não for iteração final faça
3      para cada peixe do cardume faça
4          Realizar movimento individual utilizando a Equação 2.19;
5          Avilar posição utilizando a função objetivo ;
6          se nova posição é melhor então
7              Atualizar memória do peixe;
8          fim
9      fim
10     Atualizar passo individual utilizando a Equação 2.20 ;
11     para cada peixe do cardume faça
12         Calcular a variação de posição utilizando a Equação 2.21;
13         Calcular a variação da função objetivo utilizando a Equação 2.22;
14     fim
15     Calcular peso do cardume antes do ajuste de peso;
16     para cada peixe do cardume faça
17         Ajustar o peso utilizando a Equação 2.23;
18     fim
19     Calcular peso do cardume depois do ajuste de peso;
20     Calcular direção resultante do cardume utilizando a Equação 2.24 ;
21     para cada peixe do cardume faça
22         Realizar movimento instintivo utilizando a Equação 2.25;
23     fim
24     Calcular centro de massa do cardume utilizando a Equação 2.26 ;
25     para cada peixe do cardume faça
26         se Cardume ganhou peso então
27             Realizar movimento volitivo (compactação) utilizando a Equação 2.28 ;
28         senão
29             Realizar movimento volitivo (dispersão) utilizando a Equação 2.27 ;
30         fim
31         Avilar posição utilizando a função objetivo ;
32         se nova posição é melhor então
33             Atualizar memória do peixe;
34         fim
35     fim
36     Atualizar passo volitivo utilizando a Equação 2.29 ;
37 fim
38 retorna A melhor memória do cardume
```

2.8 Algoritmo ABC

Otimização por colônia de abelhas (ABC, *Artificial Bee Colony*) [10] [132] [133] [134] [135] é um algoritmo de busca e otimização inspirado no comportamento de abelhas. A colônia de abelhas artificial contém três grupos de abelhas: as abelhas empregadas, as investigadoras e as escoteiras. Geralmente, metade da colônia é constituída por abelhas empregadas e a outra metade é constituída por abelhas investigadoras. Para cada fonte de alimento é enviada uma abelha empregada para realizar busca local. As abelhas investigadoras são enviadas em quantidade proporcional a qualidade da fonte de alimento, para realizar busca local. Quando uma abelha empregada deixa de obter sucesso em suas buscas, passa a ser um abelha escoteira, para realizar busca aleatória.

Os passos realizados a cada ciclo do algoritmo de ABC são: as abelhas empregadas selecionam uma fonte de alimento na vizinhança da fonte de alimento em sua memória, determinam as quantidades de néctar das fontes selecionadas, e compartilham suas informações com as abelhas investigadoras dentro da colmeia; as abelhas investigadoras selecionam uma das fontes de alimento baseadas nas informações compartilhadas, investigam uma fonte de alimento na vizinhança da fonte de alimento escolhida, e determinam as quantidades de néctar das fontes investigadas; uma abelha empregada após abandonar uma fonte de alimento se torna uma abelha escoteira; as abelhas escoteiras selecionam uma fonte de alimento de forma aleatória e determinam suas quantidades de néctar; e a cada iteração, a melhor fonte de alimento encontrada até o momento é memorizada.

Uma fonte de alimento representa uma possível solução para o problema a ser otimizado. A quantidade de néctar de uma fonte de alimentação corresponde à qualidade da solução representada por essa fonte de alimento. As investigadoras escolhem as melhores fontes de alimentos utilizando seleção por roleta [10].

Toda colônia de abelhas tem escoteiras. Elas são as exploradoras da colônia. As escoteiras não têm qualquer orientação na busca por alimento. Sua principal função é encontrar qualquer tipo de fonte de alimento. Como resultado desse comportamento, as escoteiras são caracterizadas por baixos custos de busca e por achar fontes de alimento de qualidade média ou baixa. Ocasionalmente, as escoteiras podem descobrir acidentalmente uma rica fonte de alimentação inteiramente desconhecida. As escoteiras funcionam como um mecanismo de descoberta rápida de um grupo de soluções factíveis.

Em um processo de busca robusta, os processos de busca em amplitude e busca em profundidade devem ser realizadas em conjunto. No algoritmo ABC, enquanto as investigadoras e as empregadas realizam o processo de busca em profundidade no espaço de busca, as escoteiras realizam o processo de busca em amplitude.

2.8.1 Qualidade da fonte de alimento

A qualidade da fonte de alimento é calculada em relação ao valor da função objetivo conforme a Equação (2.30).

$$Q[\vec{x}(t)] = \begin{cases} \frac{1}{f[\vec{x}(t)]+1}, & \text{se } f[\vec{x}(t)] \geq 0, \\ 1 + abs(f[\vec{x}(t)]+1), & \text{se } f[\vec{x}(t)] < 0, \end{cases} \tag{2.30}$$

em que $\vec{x}(t)$ é a posição fonte de alimento, *abs* é uma função que retorna o valor absoluto de seu argumento, $Q[\vec{x}(t)]$ representa a quantidade de néctar da fonte de alimentação

localizado em $\vec{x}(t)$ e $f[\vec{x}(t)]$ é o valor função objetivo na posição $\vec{x}(t)$.

2.8.2 Geração de fonte de alimento na vizinhança

Para encontrar uma fonte de alimento com mais néctar em torno da fonte de alimento atual usa-se a Equação (2.31).

$$\vec{x}_i(t+1) = \vec{x}_i(t) + [\vec{x}_i(t) - \vec{x}_k(t)]U(-1,1), \tag{2.31}$$

em que $U(-1,1)$ é uma distribuição aleatória uniforme entre -1 e 1, $\vec{x}_i(t)$ é a posição da i-ésima de fonte de alimento, $\vec{x}_k(t)$ é a posição da k-ésima de fonte de alimento diferente $\vec{x}_i(t)$ escolhida aleatoriamente, com $i \neq k$.

Se a quantidade de néctar $Q[\vec{x}_i(t+1)]$ é maior do que a quantidade de néctar $Q[\vec{x}_i(t)]$, então a fonte de alimento tem sua posição atualizada. Caso contrário, a posição atual é mantida.

2.8.3 Seleção de fonte de alimento pelas investigadoras

A probabilidade de uma fonte de alimento localizada em $\vec{x}_i(t)$ ser escolhida por uma abelha investigadora durante a roleta é calculada conforme a Equação (2.32).

$$P_i(t) = 0,9\frac{Q[\vec{x}_i(t)]}{max\{Q[\vec{x}(t)] \in P\}} + 0,1, \tag{2.32}$$

em que $\vec{x}_i(t)$ é a posição da i-ésima de fonte de alimento, $Q[\vec{x}_i(t)]$ representa a quantidade de néctar da fonte de alimentação localizado em $\vec{x}_i(t)$, e que P representa a população de fontes de alimento que está sendo visitada por abelhas. $max\{Q[\vec{x}(t)] \in P\}$ é o maior valor de $Q[\vec{x}(t)]$ no enxame.

2.8.4 Abandono de fonte de alimento

No algoritmo ABC, uma das abelhas empregadas é selecionada e classificada como a abelha escoteira. A classificação é controlada por um parâmetro de controle chamado "limite". Se uma solução que é representada por uma fonte de alimento não é melhorada por um número predeterminado de ciclos, a fonte de alimento é abandonada pela abelha empregada. E a abelha empregada associada a essa fonte de alimento se torna uma escoteira. O número de ciclo para abandonar uma fonte de alimentação é igual ao valor do parâmetro "limite", que é importante no controle do algoritmo ABC. O valor usual do parâmetro "limite" é 100, por recomendação dos autores [135].

2.8.5 Pseudo código para algoritmo ABC

O pseudo código do algoritmo ABC é apresentado no Algoritmo 6. Nas linhas 6 e 20 são feitas avaliações da função objetivo do problema. Portanto quando o algoritmo ABC for utilizado em comparações de desempenho, é importante ajustar a quantidade de abelhas na colmeia ou o números de iterações, de forma que a quantidade de avaliações da função objetiva seja igual para ambos algoritmos utilizados na comparação.

Algoritmo 6: Pseudo código do algoritmo ABC.

```
1  Iniciar as abelhas em posições aleatórias;
2  enquanto não for iteração final faça
3      para cada abelha da metade da colmeia faça
4          Selecionar duas abelhas diferentes aleatoriamente ;
5          Atualizar posição utilizando a Equação 2.31 ;
6          Avaliar posição utilizando a função objetivo ;
7          se a nova posição é melhor então
8              Atualizar memória da abelha ;
9          senão
10             Incrementa estagnação da abelha ;
11         fim
12         Calcular a qualidade da posição utilizando a Equação 2.30 ;
13     fim
14     para cada abelha da colmeia faça
15         Calcular probabilidade de seleção em roleta utilizando a Equação 2.32;
16     fim
17     para cada abelha da metade da colmeia faça
18         Selecionar duas abelhas diferentes utilizando seleção por roleta;
19         Atualizar posição utilizando a Equação 2.31 ;
20         Avaliar a fonte utilizando a função objetivo ;
21         se a nova posição é melhor então
22             Atualizar memória da abelha ;
23         senão
24             Incrementa estagnação da abelha ;
25         fim
26     fim
27     para cada abelha da colmeia faça
28         se estagnação superou limiar então
29             Gerar nova posição aleatoriamente ;
30             Reiniciar estagnação da abelha ;
31         fim
32     fim
33 fim
34 retorna A melhor memória da colmeia
```

2.9 Discussão sobre as técnicas apresentadas

Este capítulo apresentou o referencial teórico para algumas técnicas de inteligência de enxames que serviram de inspiração para os modelos estudados nesta dissertação.

As técnicas apresentadas neste capítulo foram:

- Algoritmo PSO: É o modelo inicial de otimização por enxames de partículas. É bastante simples e de fácil implementação, também possui poucos parâmetros de configuração. Os parâmetros, quando mal configurados, podem levar o algoritmo a um estado de explosão tornando todas as soluções em infactíveis. A solução adotada para evitar o estado de explosão é limitar o valor máximo do módulo da velocidade a uma percentagem da amplitude do espaço de busca, o que não é muito elegante, visto que esta ação torna o algoritmo dependente do problema.

- Algoritmo PSO Constricted: Surgiu de uma análise de estabilidade e convergência do algoritmo de PSO. Determina de forma analítica os valores dos parâmetros do PSO para os quais a convergência do algoritmo para um ótimo local seja garantida. Além disso, também previne que o estado de explosão ocorra. A análise foi feita considerando o parâmetro de inércia constante igual 1. Também é importante dizer que garantir a convergência não significa garantir um desempenho final superior.

- Algoritmo GCPSO: Foi observado que a melhor partícula do enxame não possui um líder, pois não existe outra partícula melhor do que ela. Considerando esta observação, ocorre que a melhor partícula pode parar de se movimentar rapidamente após encontrar a solução que está em sua memória cognitiva. Sendo assim, é adotada uma equação de velocidade diferente apenas para a melhor partícula, com a finalidade de evitar que a melhor partícula do enxame pare de se movimentar. Um ponto negativo é a necessidade de parâmetros de configuração adicionais.

- Algoritmo APSO: Dado o problema de encontrar a configuração correta do PSO para cada problema, esta abordagem define um sistema de adaptação de parâmetros baseada na distribuição das partículas dentro do espaço de busca. Após avaliada a distribuição das partículas, os parâmetros são configurados de forma apropriada para a situação atual. Também possui um mecanismo para escapar de eventuais ótimos locais.

- Algoritmo FSS: inspirado no comportamento de cardume de peixes, tem como característica diferencial um movimento na direção conjunta dos peixes que tiveram sucesso e também possui um mecanismo que regula a granularidade da busca através da contração e dilatação do cardume. Um ponto deficiente é o grande número de parâmetros para configuração do algoritmo.

- Algoritmo ABC: inspirado no comportamento de abelhas, apresenta um comportamento peculiar onde as localizações mais promissoras são exploradas com mais frequência do que as demais regiões. Possui também um mecanismo de detecção de estagnação e um mecanismo gerador de diversidade.

A ideia do processo de adaptação de parâmetros baseado em estado evolucionário proposta pelo algoritmo de APSO foi seminal para formulação do algoritmo proposto nesta dissertação e que será apresentado no Capítulo 4.

Capítulo 3

Fundamentos de multi-enxames cooperativos

A co-evolução [136] permite que um sistema gere diversidade suficiente para tratar problemas de complexidade bastante elevada. Se refere a situações onde os agentes mais aptos da população são afetados pela presença de outros agentes de forma que o sistema consegue escalar em dificuldade mantendo um bom grau de diversidade. As soluções produzidas por um processo de co-evolução geralmente são robustas, ou seja, são boas independente do contexto onde são utilizadas. Os principais tipos de co-evolução são:

- 1-População em co-evolução competitiva: os agentes da população evoluem através de jogos onde competem uns com os outros.
- 2-População em co-evolução competitiva: duas populações evoluem através de jogos onde as duas populações distintas competem entre elas.
- *n*-População em co-evolução cooperativa: Um problema é dividido em problemas menores, a cada população é atribuído um problema, as populações evoluem cooperando entre si para resolver os problemas.

Nesta dissertação, o conceito de co-evolução cooperativa é fundamental, pois um dos objetivos é estudar o comportamento do processo de adaptação de parâmetros baseado em estado evolucionário em um sistema com várias populações independentes. Os algoritmos apresentados neste capítulo fazem uso dos fundamentados de co-evolução cooperativa com várias populações. São detalhados os seguintes algoritmos: Clã PSO (ClanPSO), Clã PSO Dinâmico, PSO Cooperativo (CPSO) e PSO Cooperativo Multi-enxame (MCPSO).

3.1 Algoritmo ClanPSO

Clãs são grupos de indivíduos ligados entre si por algo mais do que simplesmente a proximidade (*e.g.* possuem um líder em comum). Em alguns clãs é estabelecido um líder, e esse líder guia de forma específica as decisões de cada integrante do clã. Do ponto de vista social, um clã pode ser visto como uma pequena parte de uma sociedade maior. Inspirado na noção natural de clãs e na característica do surgimento de líderes foi proposto o algoritmo de otimização Clã PSO (ClanPSO, *Clan Particle Swarm Optimization*) [19].

A topologia do ClanPSO é composta de vários clãs de partículas, onde cada clã tem uma topologia totalmente conectada. Um exemplo pode ser observado na Figura 3.1.

A cada iteração, cada clã realiza o processo de busca e marca a partícula que obteve maior sucesso dentro do clã. A partícula marcada é dita líder de seu clã. Este processo é chamado de delegação dos líderes. Após todos os clãs definirem seus líderes, é criado um novo enxame virtual apenas com os líderes e então uma outra instância do algoritmo de PSO será executada somente com os líderes dos clãs. Este processo é chamado conferência dos líderes. Os processos utilizados no ClanPSO são detalhados nas próximas subseções. Todos os clãs tem a mesma quantidade de partículas.

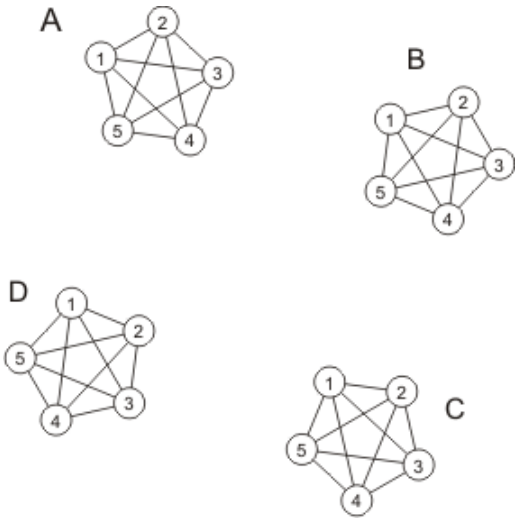


Figura 3.1: Topologia utilizando 4 clãs (A,B,C,D) de 5 partículas.

3.1.1 Delegação dos líderes

A escolha de um líder em um clã é semelhante ao processo de marcação da partícula de maior sucesso. Cada clã utiliza uma topologia de comunicação totalmente conectada, conforme pode ser visto na Figura 3.2, onde os líderes estão selecionados em cada clã.

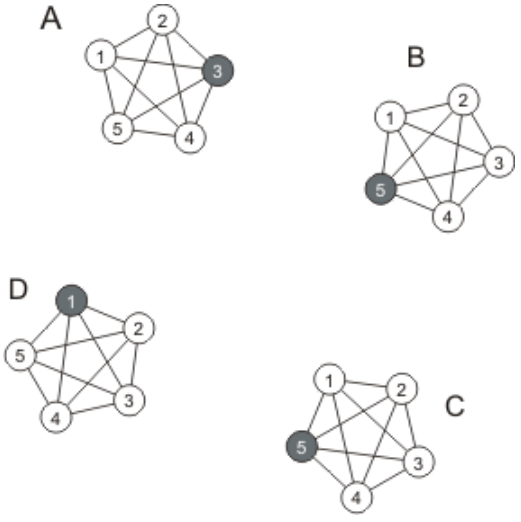


Figura 3.2: Ilustração da marcação dos líderes dos clãs (A,B,C,D).

O processo de delegação do líder utiliza a informação global para escolher o líder diretamente, sem precisar de processamento adicional para realizar a delegação. A

topologia então imita um complexo comportamento social que é formado por vários clãs com vários diferentes líderes. A partir disso, é necessário determinar qual o melhor líder para todos os clãs seguirem.

3.1.2 Conferência dos líderes

Uma solução simples para a questão de qual líder deve ser usado por todos os clãs seria utilizar a partícula que está na melhor posição. Entretanto, se essa solução fosse utilizada, o efeito seria o mesmo de ter um único clã com todas as partículas de todos os clãs. Contudo, isso não é o desejado.

É esperado que os líderes, além de escolherem o melhor dos líderes, também ajustem suas posições. Este processo é realizado tomando os líderes atuais com um enxame e executando um processo de otimização independente utilizando o PSO entre os líderes.

Como o processo de conferência dos líderes faz uso do comportamento natural entre as partículas de um enxame, pode-se utilizar das topologias globais ou anel para troca de informações, veja a ilustração deste processo na figura 3.3.

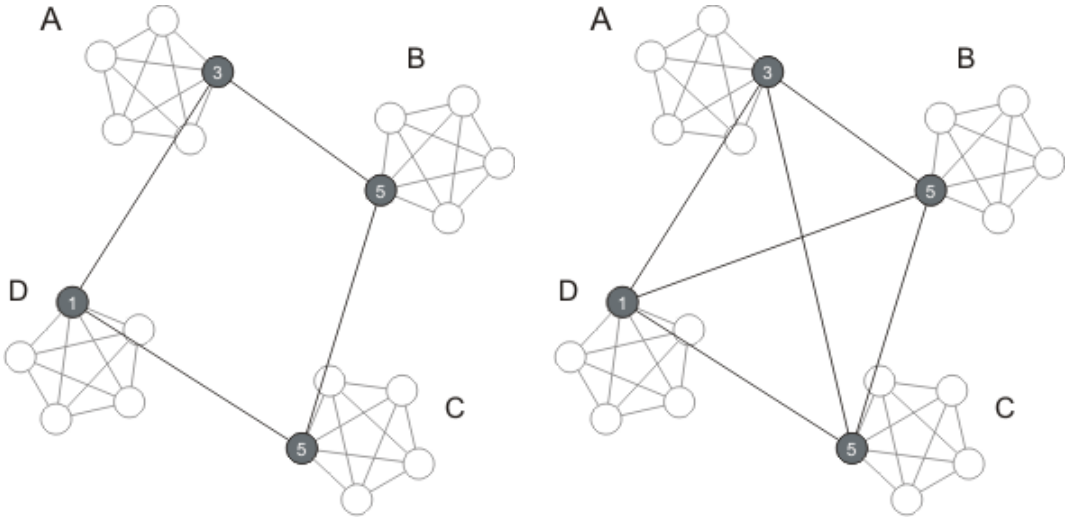


Figura 3.3: Ilustração dos tipos de conferência utilizando no Clã PSO: (a) conferência local e (b) conferência global.

O processo de conferência utiliza apenas os líderes de cada clã para executar uma nova busca com PSO. Se for utilizado um pequeno número de clãs, não será necessário muito processamento adicional em relação ao PSO original.

Quando o processo de conferência é realizado utilizando topologia global, é provido um caminho para que a convergência seja acelerada através do compartilhamento global de informação entre os líderes dos clãs. Isso aumenta a capacidade de busca em profundidade.

Quando o processo de conferência utiliza a topologia em anel, é possível explorar o espaço de busca em amplitude, espalhando as informações dos líderes em conjunto com a topologia global interna de cada clã. Sendo assim, a qualidade das soluções encontradas pelo enxame geralmente é melhorada, mas a velocidade de convergência pode ser diminuída. A topologia a ser usada no processo de conferência depende do tipo de problema a ser resolvido.

3.1.3 Retorno das informações para os clãs

Após o encerramento da conferência dos líderes, cada líder "retorna" a seu clã de origem. As novas informações adquiridas durante a conferência dos líderes são incorporadas ao clã.

Esta ação, indiretamente, guia todas as outras partículas para a melhor posição encontrada na topologia dos clãs como um todo. Como consequência, o enxame inteiro irá convergir naturalmente para o melhor lugar encontrado pelos líderes.

3.1.4 Pseudo código para algoritmo ClanPSO

O passo a passo executado pelo algoritmo de ClanPSO é apresentado no Algoritmo 7. Na linha 5, reiniciar a lista de líderes é apenas remover as partícula da lista de líderes, o conteúdo das partículas não devem ser alteradas. Na linha 18, é configurada a topologia de comunicação utilizada pelo líderes, as opções mais comuns são a topologia global e a topologia em anel (local).

Algoritmo 7: Pseudo código do algoritmo Clã PSO.

```
1  iniciar partículas com velocidades e posições aleatórias;
2  agrupar partículas em clãs (topologia global);
3  iniciar parâmetros utilizando as Equações 2.4 e 2.5 ;
4  enquanto não for iteração final faça
5      Reiniciar lista de líderes ;
6      para cada clã do enxame faça
7          para cada partícula do clã faça
8              Encontrar a melhor partícula da clã (líder) ;
9              Atualizar a velocidade utilizando a Equação 2.6 ;
10             Atualizar a posição utilizando a Equação 2.2 ;
11             Avaliar a posição utilizando função objetivo do problema;
12             se nova posição é melhor então
13                 Atualizar memória cognitiva;
14             fim
15         fim
16         Adicionar líder do clã a lista de líderes;
17     fim
18     Configurar topologia para líderes;
19     para cada partícula da lista de líderes faça
20         Encontrar a melhor partícula;
21         Atualizar a velocidade utilizando a Equação 2.6 ;
22         Atualizar a posição utilizando a Equação 2.2 ;
23         Avaliar a posição utilizando função objetivo do problema;
24         se nova posição é melhor então
25             Atualizar memória cognitiva;
26         fim
27     fim
28 fim
29 retorna A melhor memória cognitiva
```

3.2 Algoritmo ClanPSO dinâmico

No ClanPSO, é necessário definir o número de partículas para cada clã. Cada conjunto de partículas que compõe um clã é mantido no mesmo clã até o fim da execução do algoritmo. Foi observado que o número de partículas em cada clã tem influência no desempenho do algoritmo e o número ótimo de partículas por clã depende do problema. Considerando essas observações, foi proposto o Clã PSO dinâmico (*Dynamic Clan Particle Swarm Optimization*) [137] visando minimizar essas deficiências.

Considerando que algumas partículas não estão espacialmente perto de seu clã, seria interessante que estas partículas fossem influenciadas por outros clãs. É desejado que as partículas sejam livres para entrar e sair dos clãs, desde que certas condições sejam atendidas. Como consequência dessa liberdade, o número de partículas em um clã pode aumentar fortalecendo o processo de busca em profundidade do clã em questão.

Dadas essas motivações, tem-se a criação de um mecanismo que permite às partículas migrar de um clã para outro durante o processo de busca. Este novo processo é chamado de processo de migração e foi incluído no ClanPSO após a conferência dos líderes.

3.2.1 Migração de partículas

Considere que uma partícula P do Clã C_i está no espaço de busca que contém n clãs. Durante o processo de migração, é medida a distância euclidiana entre a partícula P e os líderes de todos os enxames. Se a distância euclidiana da partícula P para o líder de seu clã é maior do que a distância euclidiana para o líder de outro clã, a partícula pode migrar para o clã do líder mais próximo. Isto reorganiza os clãs com partículas que estão espacialmente próximas.

A migração pode ocorrer em iterações predefinidas, chamadas de épocas. Foram propostos dois modos de operação: o modo de exploração em amplitude e o modo de exploração em profundidade.

No modo de exploração em amplitude, as épocas de migração possuem intervalos grandes, tipicamente 1000 iterações. No modo de exploração em profundidade, as épocas possuem intervalos pequenos, tipicamente 100, 200 ou 400 iterações. O modo de exploração em amplitude é usado na maior parte do tempo durante a execução do algoritmo, o modo de exploração em profundidade é usado ao fim do processo de busca. Foi introduzido então um novo parâmetro usado para indicar em que iteração o algoritmo irá mudar do modo de exploração em amplitude para exploração em profundidade.

Um efeito colateral pode ocorrer por causa do processo de migração. Depois de algumas iterações, todas as partículas podem estar próximas umas das outras. Sendo assim, o ClanPSO dinâmico pode se comportar como se fosse um PSO com topologia global, o que não é desejado. Para solucionar esse problema é definido o número mínimo s_{min} de partículas por clã. Um clã não pode mais perder partículas quando o limite s_{min} é atingido. Isto pode evitar o comportamento extremo indesejado.

3.2.2 Pseudo código para algoritmo ClanPSO dinâmico

O pseudo código do algoritmo Clã PSO dinâmico é apresentado no Algoritmo 8. O processo de migração de partículas é realizado nas linhas 28 à 33. A distância euclidiana é calculada através da Equação 3.1.

$$d(\vec{p},\vec{q}) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \tag{3.1}$$

Algoritmo 8: Pseudo código do algoritmo Clã PSO Dinâmico.

```
1  iniciar partículas com velocidades e posições aleatórias;
2  agrupar partículas em clãs (topologia global);
3  iniciar parâmetros utilizando as Equações 2.4 e 2.5 ;
4  enquanto não for iteração final faça
5      Reiniciar lista de líderes ;
6      para cada clã do enxame faça
7          para cada partícula do clã faça
8              Encontrar a melhor partícula da clã (líder) ;
9              Atualizar a velocidade utilizando a Equação 2.6 ;
10             Atualizar a posição utilizando a Equação 2.2 ;
11             Avaliar a posição utilizando função objetivo do problema;
12             se nova posição é melhor então
13                 Atualizar memória cognitiva;
14             fim
15         fim
16         Adicionar líder do clã a lista de líderes;
17     fim
18     Configurar topologia para líderes;
19     para cada partícula da lista de líderes faça
20         Encontrar a melhor partícula;
21         Atualizar a velocidade utilizando a Equação 2.6 ;
22         Atualizar a posição utilizando a Equação 2.2 ;
23         Avaliar a posição utilizando função objetivo do problema;
24         se nova posição é melhor então
25             Atualizar memória cognitiva;
26         fim
27     fim
28     se iteração for múltiplo do limiar então
29         para cada partícula do enxame faça
30             Calcular distância euclidiana para cada um dos líderes;
31             Mover partícula para o clã do líder mais próximo;
32         fim
33     fim
34 fim
35 retorna A melhor memória cognitiva
```

3.3 Algoritmo CPSO

O PSO cooperativo (CPSO, *Cooperative PSO*) [138] [139] [140] [141] faz uso da co-evolução cooperativa para melhorar o desempenho do PSO original. São utilizados

múltiplos enxames para otimizar diferentes variáveis do vetor de solução de forma cooperativa.

Em geral, os algoritmos de otimização estocástica têm seu desempenho deteriorado com o aumento da dimensionalidade do espaço de busca. Considere que um algoritmo de busca estocástica global gera uma amostra de uma distribuição uniforme que cobre todo o espaço de busca, o algoritmo termina quando gera uma solução que pertence à região ótima. A região ótima é um pequeno hipervolume do espaço de busca em torno do ótimo global. A probabilidade de gerar uma amostra dentro da região ótima é simplesmente o hiper volume da região ótima dividido pelo hiper volume do espaço de busca. Essa probabilidade diminui exponencialmente à medida que a dimensionalidade do espaço de busca aumenta. Portanto, é muito mais difícil de encontrar o ótimo global de um problema de alta dimensionalidade do que em um problema de baixa dimensionalidade com topologia similar. Uma maneira de superar esse aumento exponencial da dificuldade é a partição do espaço de busca de alta dimensionalidade em subespaços de baixa dimensionalidade, desde que o algoritmo de otimização garanta que será capaz de pesquisar cada região possível do espaço de busca. Nas próximas seções são apresentas três versões do algoritmo, criadas pelos autores do CPSO, são elas: CPSO-S [140] , CPSO-S_k [140] e CPSO-H_k [140].

3.3.1 Algoritmo CPSO-S

No PSO original, o enxame possui vetores de n-dimensões. Esses vetores podem ser divididos criando enxames com vetores de uma dimensão. Cada enxame otimiza uma única dimensão do vetor solução.

Um problema desta configuração é o fato de a função objetivo requerer um vetor de n-dimensões como entrada. Se o enxame representa apenas uma dimensão desse vetor, não será possível computar o valor da função objetivo. Então um vetor de contexto é necessário para que seja possível calcular o valor da função objetivo utilizando enxames de uma dimensão. O esquema mais simples para construir um vetor de contexto é selecionar a melhor partícula de cada enxame de uma dimensão e concatená-los em um vetor de n-dimensões.

Cada enxame possui informação apenas sobre uma dimensão do vetor solução, o resto do vetor é provido pelos outros $n-1$ enxames. Isto promove a cooperação entre enxames diferentes, pois todos os enxames contribuem na formação do vetor de contexto.

Uma vantagem é que somente uma dimensão é modificada por vez, resultando uma busca fina de alta qualidade. Isto evita efetivamente a modificação de dimensões de forma indesejada. E pode prover uma alta diversidade, visto que o vetor de contexto é formado utilizando diferentes enxames. Um efeito colateral de dividir o vetor de solução é que não se pode garantir a convergência para um ótimo local.

3.3.2 Pseudo código para algoritmo CPSO-S

O pseudo código do algoritmo CPSO-S é apresentado no Algoritmo 9.

Algoritmo 9: Pseudo código do algoritmo CPSO-S.

```
Criar enxames de partículas de uma dimensão;
Iniciar parâmetros utilizando as Equações 2.4 e 2.5 ;
enquanto não for iteração final faça
    para cada enxame da coleção de enxames faça
        Selecionar melhor partícula do enxame;
    fim
    Montar vetor de contexto global;
    para cada enxame da coleção de enxames faça
        para cada partícula de enxame faça
            Encontrar a melhor partícula do enxame;
            Atualizar a velocidade utilizando a Equação 2.6 ;
            Atualizar a posição utilizando a Equação 2.2 ;
            Montar vetor de contexto para partícula;
            Avaliar o vetor de contexto utilizando função objetivo do problema;
            se vetor de contexto é melhor então
                Atualizar memória cognitiva da partícula;
            fim
        fim
    fim
fim
para cada enxame da coleção de enxames faça
    Selecionar melhor partícula do enxame;
fim
Montar vetor de contexto global;
retorna Vetor de contexto global
```

3.3.3 Algoritmo CPSO-S_k

O algoritmo CPSO-S_k é a generalização do CPSO-S para tratar problemas em que algumas dimensões podem estar correlacionadas. Em alguns problemas, as dimensões do vetor de solução podem estar altamente correlacionadas e estas devem ser agrupadas em um mesmo enxame, caso contrário mudanças independentes feitas por outros enxames podem causar um efeito indesejado nas dimensões com alta correlação.

Infelizmente nem sempre as dimensões que são correlacionadas são conhecidas a priori. Uma solução para isso é escolher c-dimensões aleatoriamente e assumi-las como correlacionadas. Alguns enxames tem uma dimensão e outros são de c-dimensões ($c < n$). Embora seja menos frequente a convergência para uma solução sub-ótima, o CPSO-S_k continua sem garantia de convergência para um ótimo local.

3.3.4 Pseudo código para algoritmo CPSO-S_k

O pseudo código do algoritmo CPSO-S_k é apresentado no Algoritmo 10.

Algoritmo 10: Pseudo código do algoritmo CPSO-S_k.

```
1  Criar enxames de partículas de K dimensões aleatórias;
2  Iniciar parâmetros utilizando as Equações 2.4 e 2.5 ;
3  enquanto não for iteração final faça
4      para cada enxame da coleção de enxames faça
5          Selecionar melhor partícula do enxame;
6      fim
7      Montar vetor de contexto global;
8      para cada enxame da coleção de enxames faça
9          para cada partícula de enxame faça
10             Encontrar a melhor partícula do enxame;
11             Atualizar a velocidade utilizando a Equação 2.6 ;
12             Atualizar a posição utilizando a Equação 2.2 ;
13             Montar vetor de contexto para partícula;
14             Avaliar o vetor de contexto utilizando função objetivo do problema;
15             se vetor de contexto é melhor então
16                 Atualizar memória cognitiva da partícula;
17             fim
18         fim
19     fim
20 fim
21 para cada enxame da coleção de enxames faça
22     Selecionar melhor partícula do enxame;
23 fim
24 Montar vetor de contexto global;
25 retorna Vetor de contexto global
```

3.3.5 Algoritmo CPSO-H_k

O CPSO-H_k é uma hibridização do CPSO-S_k que garante a convergência para um ótimo local. Dada a garantia de convergência para um ótimo local do PSO e a rápida convergência do CPSO-S_k em certos tipos de funções, seria ideal um algoritmo que explorasse ambas as propriedades. Isso é feito através do uso intercalado dos dois algoritmos, então o CPSO-S_k é executado por um ciclo, seguido de um ciclo do PSO. Algoritmos co-evolutivos podem ser construídos utilizando a troca de informações sobre as melhores soluções descobertas até o momento pelas populações de entidades. Esta troca de informações é uma forma de cooperação entre os algoritmos CPSO-S_k e PSO.

É necessário um mecanismo para realizar a troca de informações. Neste caso, algumas das partículas de um dos algoritmos são trocadas com a melhor solução descoberta até o momento pelo outro algoritmo. Depois de uma iteração do CPSO-S_k, o vetor de contexto é usado para substituir uma partícula escolhida aleatoriamente no enxame do PSO com exceção da melhor. Então, é executada uma iteração do PSO. A melhor partícula do enxame do PSO é dividida em vetores de uma dimensão, que vão substituir partículas escolhidas aleatoriamente nos enxames do CPSO-S_k, excluindo as melhores de cada enxame.

Durante a troca de informação entre os algoritmos, as melhores soluções de cada um são protegidas, pois modificações indesejadas nessas soluções podem levar a uma

perda de desempenho. Empiricamente é demonstrado que o excesso de troca de forma de informações pode impedir o progresso do algoritmo. A seleção de partículas feita por uma distribuição aleatória uniforme pode causar a substituição de todas as partículas exceto, da melhor. Isto pode ocorrer especialmente no caso em que um enxame esteja em defasagem em relação a outro, principalmente nas primeiras iterações. Portanto, o excesso de troca de informações pode levar a uma diminuição significativa na diversidade das partículas.

Um mecanismo para prevenir a perda na diversidade das partículas pode ser implementado limitando o número máximo de partículas que podem participar ativamente na troca de informações. Se apenas metade das partículas são possíveis alvos para serem substituídos, então no máximo a metade da diversidade do enxame pode ser comprometida. Isso não afeta significativamente a influência positiva do processo de troca de informações.

3.3.6 Pseudo código para algoritmo CPSO-H_k

O pseudo código do algoritmo CPSO-H_k é apresentado no Algoritmo 11.

Algoritmo 11: Pseudo código do algoritmo CPSO-H_k.

```
1  Criar enxames de partículas de K dimensões aleatórias;
2  Criar um enxame de partículas com todas as dimensões do problema (N);
3  Iniciar parâmetros utilizando as Equações 2.4 e 2.5 ;
4  enquanto não for iteração final faça
5      Executar algoritmo CPSO-S_k no enxame K;
6      Montar vetor de contexto (K);
7      Substituir uma partícula do enxame N pelo vetor de contexto (K);
8      Executar algoritmo PSO no enxame N;
9      Dividir a melhor partícula do enxame N em sub-partículas K;
10     Substituir partículas nos enxames K pelas sub-partículas K;
11 fim
12 retorna A melhor solução encontrada
```

3.4 Algoritmo MCPSO

PSO cooperativo multi-enxame (MCPSO, *Multi-Swarm Cooperative PSO*) [142] é uma abordagem que tem como objetivo balancear a capacidade de busca em amplitude e busca em profundidade do PSO original através de um esquema multi-enxame cooperativo.

Nos sistemas naturais muitas espécies desenvolvem interações cooperativas com outras espécies com a finalidade de aumentar sua chance de sobrevivência. Essa co-evolução cooperativa é chamada de simbiose [143]. O fenômeno da simbiose pode ser encontrado em muitas formas de vida [144].

As relações de simbiose podem ser classificadas como:

- Mutualismo: ambas as espécie se beneficiam da relação;
- Comensalismo: uma das espécies se beneficia enquanto a outra não é afetada;

- Parasitismo: uma das espécies se beneficia enquanto a outra é afetada;

O MCPSO incorpora a ideia de comensalismo. O modelo consiste de um enxame mestre e vários enxames escravos. Os enxames escravos fornecem novas partículas promissoras para o enxame mestre durante o processo de busca. O enxame mestre atualiza suas partículas utilizando todas suas partículas e as melhores partículas de todos os enxames escravos. As interações entre o enxame mestre e os escravos influenciam na capacidade de busca em amplitude e busca em profundidade, e mantêm a diversidade da população de forma a reduzir o risco de convergência para um ótimo local.

Foram criados modelos que dependem do tipo de relação co-evolucionária entre os enxames mestre e os escravos:

- MCPSO Competitivo (COM-MCPSO): é baseado em um cenário antagônico, no qual o enxame mestre melhora suas partículas por meio de competições com os enxames escravos.

- MCPSO Colaborativo (COL-MCPSO): é baseado em um cenário sinergético, no qual o enxame mestre melhora suas partículas por meio de colaborações com os enxames escravos.

Cada enxame escravo executa o algoritmo de PSO ou variação do algoritmo de PSO. A cada ciclo de busca do PSO os enxames escravos enviam a melhor partícula para o enxame mestre. O enxame mestre seleciona a melhor de todas as partículas enviadas pelos enxames escravos e atualiza as velocidades de suas partículas.

3.4.1 MCPSO Competitivo

O modelo COM-MCPSO é caracterizado pela oportunidade que a melhor partícula entre todos os enxames (mestre ou escravo) tem de guiar o enxame mestre. Para isso é utilizada a Equação (3.2) na atualização de velocidade das partículas do enxame mestre.

$$\vec{v}(t+1) = \omega\vec{v}(t) + r_1 c_1 [\vec{x}_p(t) - \vec{x}(t)] + \Phi r_2 c_2 [\vec{x}_m(t) - \vec{x}(t)] + (1-\Phi) r_3 c_3 [\vec{x}_s(t) - \vec{x}(t)], \quad (3.2)$$

em que r_1, r_2, r_3 são números aleatório obtidos de um distribuição uniforme entre 0 e 1, $\vec{x}_p(t)$ é a memória cognitiva da partícula, $\vec{x}_m(t)$ é a memória cognitiva da melhor partícula do enxame mestre, $\vec{x}_s(t)$ é a memória cognitiva da melhor partícula enviada pelos enxames escravos e Φ é o fator de migração. Para um problema de minimização, o valor de fator de migração é dado conforme Equação (3.3).

$$\Phi = \begin{cases} 0, & se f(\vec{x}_s) < f(\vec{x}_m), \\ 0,5, & se f(\vec{x}_s) = f(\vec{x}_m), \\ 1, & se f(\vec{x}_s) > f(\vec{x}_m). \end{cases} \quad (3.3)$$

3.4.2 Pseudo código para algoritmo COM-MCPSO

O pseudo código do algoritmo COM-MCPSO é apresentado no Algoritmo 12.

Algoritmo 12: Pseudo código do algoritmo COM-MCPSO.

```
1  Criar enxames;
2  enquanto não for iteração final faça
3      Reiniciar lista das melhores partículas;
4      para cada enxame da coleção de enxames faça
5          Executar algoritmo PSO no enxame;
6          Selecionar a melhor partícula na do enxame;
7          Adicionar a melhor partícula na lista das melhores;
8      fim
9      para cada partícula da lista de melhores partículas faça
10         Atualizar velocidade utilizando a Equação 3.2 ;
11         Atualizar a posição utilizando a Equação 2.2 ;
12         Avaliar posição utilizando a função objetivo do problema ;
13         se posição atual melhor que memória cognitiva então
14             Atualizar a memória cognitiva ;
15         fim
16     fim
17 fim
18 retorna A melhor solução encontrada
```

3.4.3 MCPSO Colaborativo

O modelo COL-MCPSO é caracterizado pela habilidade de colaboração entre o enxame mestre e os enxames escravos. Para isso é utilizada a Equação (3.4) na atualização de velocidade das partículas do enxame mestre.

$$\vec{v}(t+1) = \omega\vec{v}(t) + r_1 c_1 [\vec{x}_p(t) - \vec{x}(t)] + r_2 c_2 [\vec{x}_m(t) - \vec{x}(t)] + r_3 c_3 [\vec{x}_s(t) - \vec{x}(t)], \quad (3.4)$$

em que r_1, r_2, r_3 são números aleatório obtidos de um distribuição uniforme então 0 e 1, $\vec{x}_p(t)$ é a memória cognitiva da partícula, $\vec{x}_m(t)$ é a memória cognitiva da melhor partícula do enxame mestre, $\vec{x}_s(t)$ é a memória cognitiva da melhor partícula enviada pelos enxames escravos.

3.4.4 Pseudo código para algoritmo COL-MCPSO

O pseudo código do algoritmo COL-MCPSO é apresentado no Algoritmo 13.

Algoritmo 13: Pseudo código do algoritmo COL-MCPSO.

```
1  Criar enxames;
2  enquanto não for iteração final faça
3      Reiniciar lista das melhores partículas;
4      para cada enxame da coleção de enxames faça
5          Executar algoritmo PSO no enxame;
6          Selecionar a melhor partícula na do enxame;
7          Adicionar a melhor partícula na lista das melhores;
8      fim
9      para cada partícula da lista de melhores partículas faça
10         Atualizar velocidade utilizando a Equação 3.4 ;
11         Atualizar a posição utilizando a Equação 2.2 ;
12         Avaliar posição utilizando a função objetivo do problema ;
13         se posição atual melhor que memória cognitiva então
14             Atualizar a memória cognitiva ;
15         fim
16     fim
17 fim
18 retorna A melhor solução encontrada
```

3.5 Discussão sobre as técnicas apresentadas

Neste capítulo foram apresentadas as técnicas de:

- Algoritmo Clã PSO: uma nova topologia dinâmica para PSO que agrupa as partículas em clãs que se reúnem em conferência para troca de informações. A troca de informações de maneira cooperativa entre os clãs pode oferecer melhoria no desempenho do algoritmo. No entanto, a configuração ótima do número de clãs e partículas por clã pode depender do problema.

- Algoritmo Clã PSO dinâmico: propõe uma solução para tornar o Clã PSO menos dependente da configuração manual do número de partículas por clã, através de um mecanismo de migração, onde as partículas mais próximas espacialmente de um clã são absorvidas. Este algoritmo não será abordado nos resultados desta dissertação, pois não demonstrou melhoria expressiva em relação ao Clã PSO.

- Algoritmo CPSO: divide o vetor solução multi-dimensional em vetores de uma dimensão e cria um vetor de contexto reunindo as melhores soluções de cada enxame uni-dimensional. No entanto, existem alguns tipos específicos de problemas (rotacionados) que não podem ser resolvidos por algoritmos de otimização uni-dimensionais. A variação de CPSO utilizada mais adiante será o CPSO-*S*.

- Algoritmo MCPSO: utiliza múltiplos enxames escravos para fornecer melhores partículas para o enxame mestre. Através do processo cooperativo de troca de informações, o enxame mestre pode obter resultados melhores. Uma deficiência é que o número de enxames escravos depende do problema. A abordagem colaborativa será utilizada para comparação nesta dissertação.

Capítulo 4

Otimização por Enxames de Partículas utilizando Clãs com Comportamento Adaptativo

O algoritmo de Otimização por Enxames de Partículas utilizando Clãs com Comportamento Adaptativo (ClanAPSO, *Clan Adaptative Particle Swarm Optimization*) propõe incorporar um processo de adaptação sistemática de parâmetros a um estratégia de busca multi enxames. Essa proposta é inspirada no conceito de clãs presente no algoritmo ClanPSO e, nos conceitos de adaptação de parâmetros e estratégia elitista presentes no algoritmo APSO.

O processo de adaptação de parâmetros, presente no algoritmo APSO, demonstrou melhora significativa de desempenho, em um grande número de problemas de teste, em relação a versões não adaptativas do algoritmo PSO. Portanto, a adaptação de parâmetros demonstra ser um recurso interessante para ser incorporado a novas versões do algoritmo PSO. Porém, quando aplicado a um único enxame, o processo de adaptação pode realizar apenas um tipo de busca por iteração do algoritmo.

A proposta do ClanAPSO identificou a oportunidade de utilizar um processo de adaptação em cada enxame de um sistema multi enxame, isto permite que alguns enxame possam realizar diferentes tipos de busca por iteração do algoritmo, ou seja, existe a possibilidade de executar busca em amplitude e busca em profundidade de maneira simultânea.

O conceito de clãs (co-evolução cooperativa), presente no algoritmo ClanPSO, é interessante porque permite identificar os líderes locais de um grupo de partículas e, realizar troca de informações entre clãs de maneira indireta. A comunicação indireta entre clãs promove uma melhoria na segurança do processo de busca, pois trata-se de um processo distribuído de propagação de informações. A proposta do ClanAPSO identificou que a utilização de clãs demonstrou melhores resultados do que a execução intercalada de vários enxames.

A estratégia elitista, presente no algoritmo APSO, funciona de modo a substituir a melhor ou pior partícula de um enxame, dependendo da qualidade da solução obtida pela mutação gaussiana. Quando executada em um único enxame, demonstra resultados melhores do que versões sem estratégia elitista. No caso da proposta do ClanAPSO, quando a estratégia elitista é executada na conferência dos líderes e, ocorre a substituição da pior partícula, que no caso é líder de algum clã tem-se um comportamento semelhante ao obtido pelo *Dynamic* ClanPSO. Ao contrario do que ocorre no *Dynamic* ClanPSO onde

as partícula mais próxima de um líder são migradas de enxame para serem influenciadas por um novo líder, no ClanAPSO o líder do clã mais “fraco” é substituído pela mutação do líder do clã mais “forte”, que passa a influenciar o clã.

4.1 Pseudo código para algoritmo ClanAPSO

O pseudo código do algoritmo de ClanAPSO é apresentado no Algoritmo 14.

Algoritmo 14: Pseudo código do algoritmo Clã PSO Adaptativo.

```
1  iniciar partículas com velocidades e posições aleatórias;
2  agrupar partículas em clãs (topologia global);
3  enquanto não for iteração final faça
4      Reiniciar lista de líderes ;
5      para cada clã do enxame faça
6          Executar adaptação de parâmetros ;
7          Executar estratégia elitista ;
8          para cada partícula do clã faça
9              Encontrar a melhor partícula da clã (líder) ;
10             Atualizar a velocidade utilizando a Equação 2.1 ;
11             Atualizar a posição utilizando a Equação 2.2 ;
12             Avaliar a posição utilizando função objetivo do problema;
13             se nova posição é melhor então
14                 Atualizar memória cognitiva;
15             fim
16         fim
17         Adicionar líder do clã a lista de líderes;
18     fim
19     Configurar topologia para líderes;
20     Executar adaptação de parâmetros ;
21     Executar estratégia elitista ;
22     para cada partícula da lista de líderes faça
23         Encontrar a melhor partícula;
24         Atualizar a velocidade utilizando a Equação 2.1 ;
25         Atualizar a posição utilizando a Equação 2.2 ;
26         Avaliar a posição utilizando função objetivo do problema;
27         se nova posição é melhor então
28             Atualizar memória cognitiva;
29         fim
30     fim
31 fim
32 retorna A melhor memória cognitiva
```

Nas linhas 6, 7, 20 e 21 encontra-se os pontos onde foram realizadas as modificações no algoritmo ClanPSO para incorporar o processo adaptativo de parâmetros baseado em estado evolucionário (Algoritmo 15) e a estratégia elitista (Algoritmo 16) existentes do algoritmo APSO. Nas linhas 10, 11, 24 e 25 encontra-se outra diferença, as equações de

velocidade e posição utilizadas no ClanAPSO são as mesmas do PSO original, invés das equações de velocidade e posição do PSO Constricted que são utilizadas no ClanPSO.

No Algoritmo 15 é apresentado o processo de adaptação de parâmetros. Este processo foi desenvolvido pelo algoritmo de APSO, o objetivo deste processo é torna o algoritmo mais independe do problema, eliminando a necessidade de configuração manual dos parâmetros de aceleração e inércia.

Algoritmo 15: Pseudo código da adaptação de parâmetros.

```
para cada partícula do clã faça
    Calcular distância média utilizando a Equação 2.9 ;
fim
Calcular fator evolucionário utilizando a Equação 2.10 ;
Calcular pertinências utilizando as Equações 2.11,2.12,2.13 e 2.14 ;
Classificar clã em um estado evolucionário ;
Adaptar os coeficientes de aceleração conforme Tabela 2.1 ;
Normalizar os coeficientes de aceleração utilizando a Equação 2.15 ;
Atualizar parâmetro de inércia utilizando Equação 2.18 ;
```

No Algoritmo 16 é apresentada a estratégia elitista. O objetivo desta estratégia é a geração de novos estados de escape, através da perturbação da posição obtida pela melhor partícula de um clã. Quando a perturbação é bem sucedida, um novo estado de escape é criado, caso contrário a pior partícula do clã é substituída pela posição obtida após a perturbação da melhor partícula do clã.

Algoritmo 16: Pseudo código da estratégia elitista.

```
se clã classificado em estado de convergência então
    Gerar uma nova posição utilizando as Equações 2.16 e 2.17 ;
    se nova a posição é melhor que a melhor memória cognitiva então
        Atualizar a posição e memória cognitiva da melhor partícula do clã;
    senão
        Atualizar a posição e memória cognitiva da pior partícula do clã;
    fim
fim
```

4.2 Custo computacional

A divisão do enxames em clãs faz com que processo de adaptação de parâmetros, tenha o custo computacional reduzido. Quando todas as partículas pertence ao mesmo enxame, como é o caso do APSO, é necessário calcular a distância euclidiana de uma partícula para todas as outras, o que resulta em $n-1$ cálculos da distância euclidiana para um enxame de n partículas, portanto são necessários $n(n-1)$ cálculos de distância euclidiana entre partículas. A Equação 4.1 apresenta total de cálculos de distância euclidiana entre partículas para o algoritmo APSO.

$$n^2 - n. \tag{4.1}$$

No caso do ClanAPSO, as n partículas são dividas em clãs c de $\frac{n}{c}$ partículas, portanto para cada partícula de um clã são necessários $\frac{n}{c}-1$ cálculos de distância euclidiana. Logo, para cada clã são necessários $\frac{n}{c}(\frac{n}{c}-1)$ cálculos de distância euclidiana. Para a conferência de c líderes, tem-se um total de $c(c-1)$ cálculos de distância euclidiana. No total, para todos os clãs, incluindo a conferência líderes, são necessários $c[\frac{n}{c}(\frac{n}{c}-1)]+c(c-1)$. A Equação 4.2 apresenta total de cálculos de distância euclidiana entre partículas para o algoritmo ClanAPSO.

$$\frac{n^2}{c}-n+c^2-c. \tag{4.2}$$

Para o algoritmo ClanAPSO funcionar corretamente é necessário pelo menos 3 clãs de 3 partículas, porque para calcular a fator evolucionário é preciso de pelo menos duas distância diferentes para evitar divisão por zero. O mínimo de 3 clãs é porque o enxame virtual de líderes também necessita de duas distância diferentes para evitar divisão por zero. Portanto o mínimo de partículas utilizadas são nove. A Tabela 4.1 apresenta a quantidade necessária de cálculos de distância euclidiana para os algoritmos APSO e ClanAPSO.

Tabela 4.1: Quantidade de cálculos de distância euclidiana para APSO e ClanAPSO.

Total de partículas	APSO (partículas)	ClanAPSO (clãs x partículas)
9	(9)=72	(3x3)=24
12	(12)=132	(3x4)=42 (4x3)=36
30	(30)=870	(3x10)=276 (5x6)=170 (6x5)=150 (10x3)=150

Capítulo 5

Arranjo experimental e resultados

Este capítulo descreve: os problemas de otimização que serão utilizados nos experimentos, a metodologia experimental e, diversas análises sobre os dados obtidos das simulações.

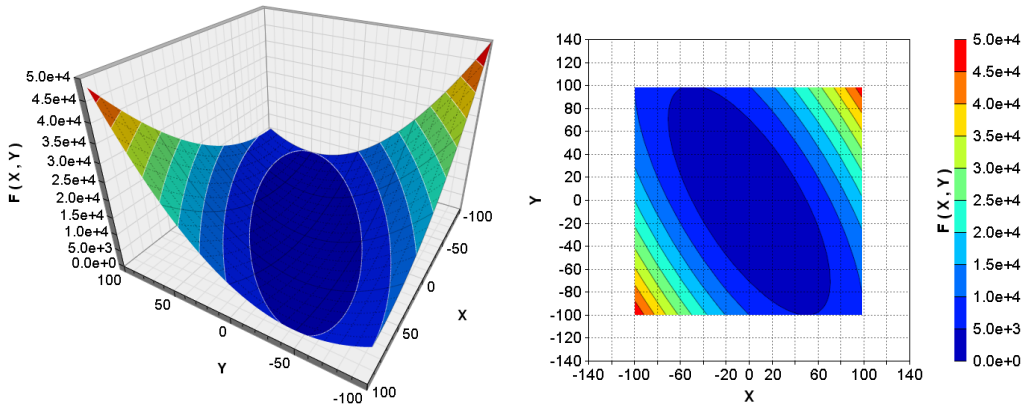
5.1 Problemas de teste

Problemas de teste são empregadas para medir o desempenho de algoritmos. Na literatura é possível encontrar conjuntos de problemas de teste que exploram diversos aspectos específicos de problemas de otimização [145] [140] [26] [20] [56]. Nesta dissertação, o critério de escolha para o conjunto de problemas de teste utilizado foi o número de citações do conjunto de teste na literatura, porque facilita a comparação de resultados com outros autores.

5.1.1 Problema Schwefel 1.2

Trata-se de um problema unimodal com vários níveis de platores. Descrito na Equação (5.1). Ilustrado na Figura 5.1.

$$f(\vec{x}) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2, \quad n \geq 1. \tag{5.1}$$



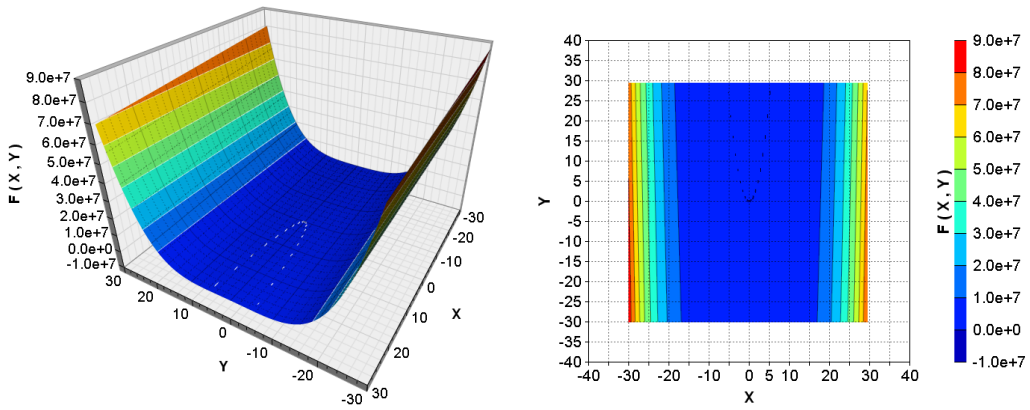
(a) Superfície (b) Contorno

Figura 5.1: Ilustração da função Schwefel 1.2 em 2D.

5.1.2 Problema Rosenbrock

Trata-se de um problema unimodal com platores nas duas últimas dimensões. Descrito na Equação (5.2). Ilustrado na Figura 5.2.

$$f(\vec{x}) = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right], \quad n \geq 2. \tag{5.2}$$



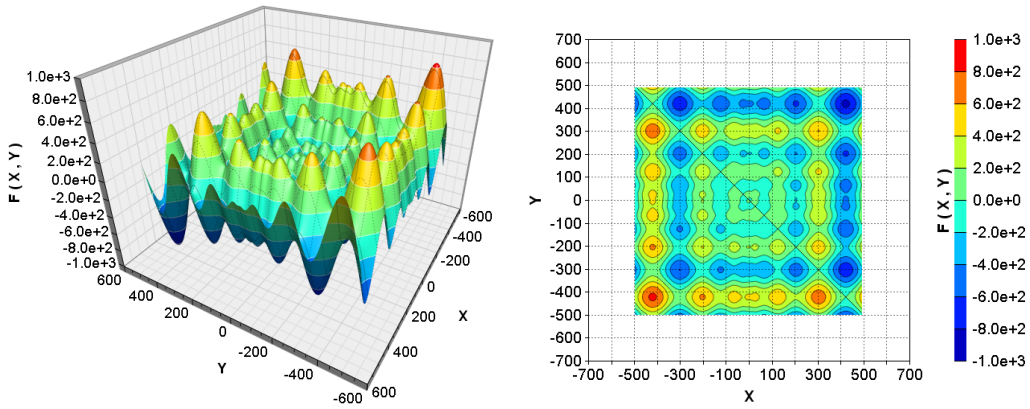
(a) Superfície (b) Contorno

Figura 5.2: Ilustração da função Rosenbrock em 2D.

5.1.3 Problema Schewefel 2.26

Trata-se de um problema multimodal onde os mínimos e máximos estão dispostos simetricamente. Descrito na Equação (5.3). Ilustrado na Figura 5.3.

$$f(\vec{x}) = -\sum_{i=1}^{n} [x_i \cdot \sin(\sqrt{|x_i|})], \quad n \geq 1. \tag{5.3}$$



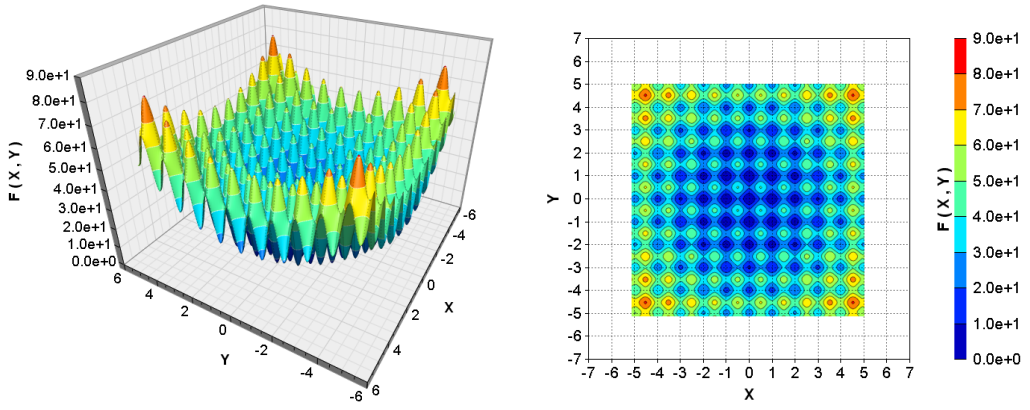
(a) Superfície (b) Contorno

Figura 5.3: Ilustração da função Schwefel 2.26 em 2D.

5.1.4 Problema Rastrigin

Trata-se de um problema multimodal. Descrito na Equação (5.4). Ilustrado na Figura 5.4.

$$f(\vec{x}) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right], \quad n \geq 1. \tag{5.4}$$



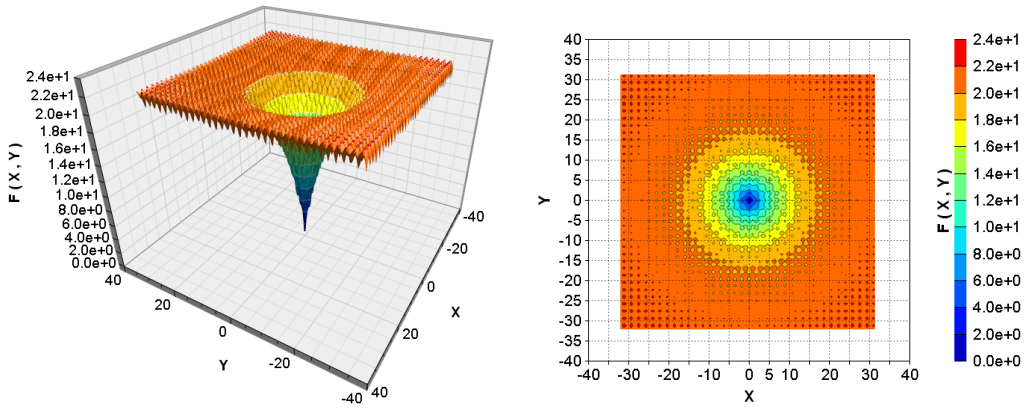
(a) Superfície (b) Contorno

Figura 5.4: Ilustração da função Rastrigin em 2D.

5.1.5 Problema Ackley

Trata-se de um problema multimodal. Descrito na Equação (5.5). Ilustrado na Figura 5.5.

$$f(\vec{x}) = -20\exp\left(-0,2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e, \quad n \geq 1. \tag{5.5}$$



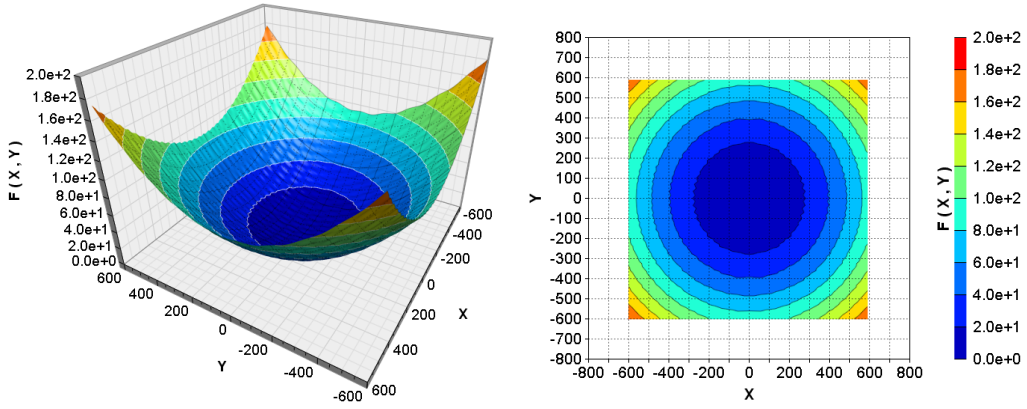
(a) Superfície (b) Contorno

Figura 5.5: Ilustração da função Ackley em 2D.

5.1.6 Problema Griewank

Trata-se de um problema multimodal. Descrito na Equação (5.6); Ilustrado na Figura 5.6.

$$f(\vec{x}) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right), \quad n \geq 1. \tag{5.6}$$



(a) Superfície (b) Contorno

Figura 5.6: Ilustração da função Griewank em 2D.

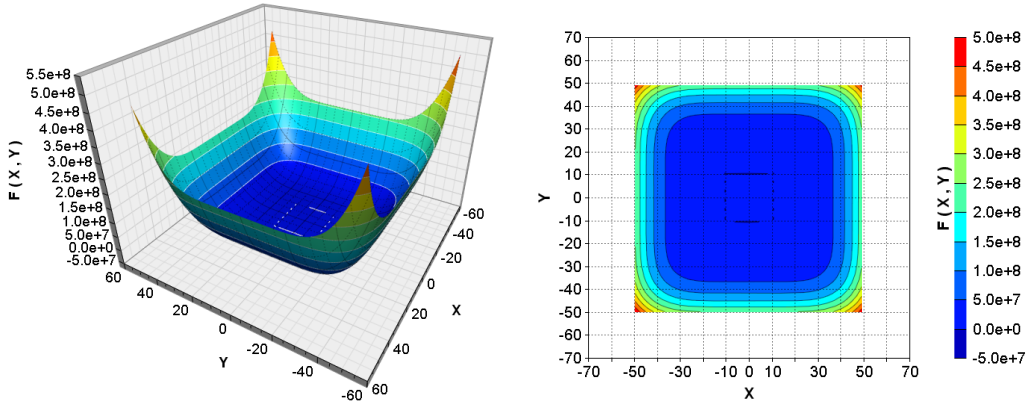
5.1.7 Problema Penalized P8

Trata-se de um problema multimodal. Descrito nas Equações (5.7), (5.8) e (5.9). Ilustrado na Figura 5.7.

$$f(\vec{x}) = \frac{\pi}{n}\left\{10\sin^2(\pi y_1) + \sum_{i=1}^{n-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})] + (y_n-1)^2\right\} + \sum_{i=1}^{n} u(x_i,10,100,4), \quad n \geq 2, \tag{5.7}$$

$$u(x_i,a,k,m) = \begin{cases} k(x_i-a)^m & x_i > a \\ 0 & -a \leq x_i \leq a \\ k(-x_i-a)^m & x_i < -a \end{cases}, \tag{5.8}$$

$$y_i = 1 + \frac{1}{4}(x_i+1). \tag{5.9}$$



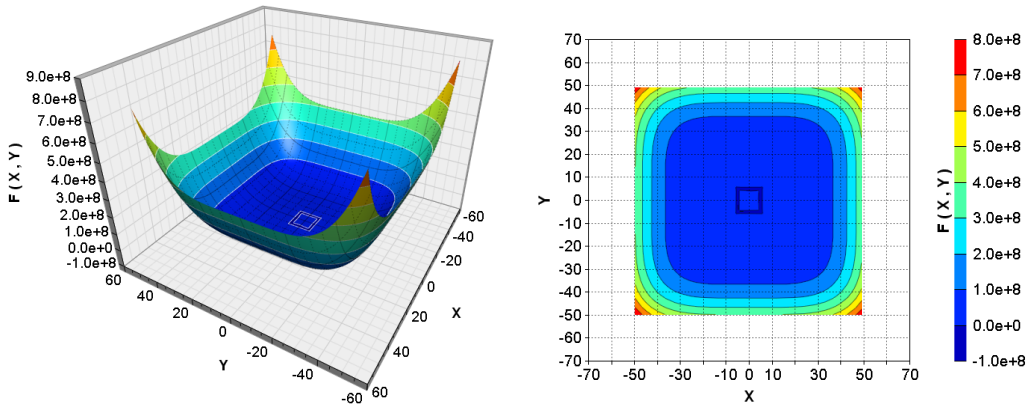
(a) Superfície (b) Contorno

Figura 5.7: Ilustração da função P8 em 2D.

5.1.8 Problema Penalized P16

Trata-se de um problema multimodal. Descrito nas Equações (5.10),(5.8) e (5.9). Ilustrado na Figura 5.8.

$$f(\vec{x}) = 0,1\left\{\sin^2(\pi 3x_1) + \sum_{i=1}^{n-1}(x_i-1)^2[1-\sin^2(3\pi x_{i+1})] + (x_n-1)^2[1+\sin^2(2\pi x_n)]\right\} + \sum_{i=1}^{n} u(x_i,5,100,4), \quad n \geq 2 \tag{5.10}$$



(a) Superfície (b) Contorno

Figura 5.8: Ilustração da função P16 em 2D.

5.1.9 Definição de espaços e ponto ótimo dos problemas de teste

A Tabela 5.1, especifica o espaço de busca, o espaço de inicialização e o ponto ótimo para cada um dos problemas de teste. O objetivo é encontrar o ponto ótimo que minimiza a função objetivo do problema.

Tabela 5.1: Limites do espaço de busca e ponto ótimo para problemas de minimização.

Função	Espaço de busca	Espaço de inicialização	Ponto ótimo
Schwefel 1.2	$-100 \leq x_i \leq 100$	$50 \leq x_i \leq 100$	$f(0;...;0) = 0$
Rosenbrock	$-30 \leq x_i \leq 30$	$15 \leq x_i \leq 30$	$f(1;...;1) = 0$
Schewefel 2.26	$-500 \leq x_i \leq 500$	$-500 \leq x_i \leq -250$	$f(420,9;...;420,9) \approx -nx_i$
Rastrigin	$-5,12 \leq x_i \leq 5,12$	$2,50 \leq x_i \leq 5,12$	$f(0;...;0) = 0$
Ackley	$-32 \leq x_i \leq 32$	$16 \leq x_i \leq 32$	$f(0;...;0) = 0$
Griewank	$-600 \leq x_i \leq 600$	$300 \leq x_i \leq 600$	$f(0;...;0) = 0$
Penalized P8	$-50 \leq x_i \leq 50$	$25 \leq x_i \leq 50$	$f(1;...;1) = 0$
Penalized P16	$-50 \leq x_i \leq 50$	$25 \leq x_i \leq 50$	$f(1;...;1) = 0$

5.2 Metologia experimental

Esta seção descreve como foram realizados os procedimentos de experimentação dos algoritmos. São tratados os seguintes tópicos: a geração das solução iniciais, o tratamento das solução infactíveis pelos algoritmos, as medidas de avaliação dos algoritmos, a nomenclatura dos algoritmos e, os parâmetros dos algoritmos.

5.2.1 Geração das solução iniciais

As soluções inicias para todos os algoritmos experimentados, foram geradas aleatoriamente no espaço de inicialização de cada problema, conforme especificado na Tabela 5.1.

O espaço de inicialização não contém o ponto ótimo da problema, os algoritmos precisam ser capazes de se moverem no espaço de busca, escapando dos ótimos locais e outras eventuais dificuldades, na tentativa de encontrar o ponto ótimo da problema.

5.2.2 Tratamento das soluções infactíveis pelos algoritmos

Os algoritmos podem gerar soluções infactíveis ao longo do processo de busca. A avaliação de uma solução foram das especificações do espaço de busca representa um custo desnecessário, por esta razão as soluções infactíveis serão ignoradas pelos algoritmos.

Ignorar as soluções infactíveis, pode aumentar o número de iterações necessárias por um algoritmo para obter um resultado satisfatório. Caso o número de soluções infactíveis geradas seja "grande", pode ocorre um aumento do tempo de execução do algoritmo. Para mitigar esse problema, é preciso configurar corretamente os parâmetros dos algoritmos, para minimizar o número de soluções infactíveis geradas.

5.2.3 Medidas de avaliação dos algoritmos

Cada experimento é repetido pelo menos 30 vezes, porque devido a utilização de números aleatórios nos algoritmos os resultados podem variar. Sobre o conjunto de resultados obtidos nas varias repetições de cada experimento podem ser calculadas as medidas estatísticas de média e desvio padrão, bem como testes de significância estatística.

São observados os seguintes aspectos dos algoritmos:

1. O valor da função objetivo para melhor solução encontrada;
2. A dispersão das soluções geradas no espaço busca durante a execução do algoritmo;
3. A velocidade do convergência do algoritmo;
4. O desempenho em relação ao aumento de dificuldade (aumento de dimensões);
5. O tempo de execução do algoritmo.

5.2.4 Nomenclatura dos algoritmos

A nomenclatura que é utilizada nas próximas seções segue o seguinte padrão: <algoritmo>-<número de clãs>x<número de partículas por clã>-<topologia utilizada na conferência dos líderes>-<dimensões do problema>D.

Para <algoritmo> algumas opções são: "ClanPSO" para o algoritmo de Clã PSO, "ClanAPSO" para o algoritmo de Clã PSO Adaptativo. Para a topologia utilizada é utilizado: "L" para topologia local e "G" para topologia global. Por exemplo: ClanAPSO-6x5-L se refere ao algoritmo de ClanAPSO com 6 clãs de 5 partículas utilizando topologia local na conferência do líderes.

5.2.5 Parâmetros dos algoritmos

O número de dimensões das soluções varia 2 até 1000 dimensões dependendo da análise. Todos os algoritmos serão configurados para executar 10 mil iterações (300 mil chamadas à função objetivo).

O número de iterações é utilizado como critério de parada dos algoritmos. Este critério foi escolhido porque limita a quantidade de rotinas a serem chamadas de forma determinística e, permite a comparação justa entre os algoritmos, visto que todos os algoritmos puderam avaliar a mesma quantidade de soluções geradas.

O algoritmo *Mersenne Twister* [146] é utilizado para geração de números aleatórios, a cada experimento é realizado utilizando uma semente diferente. A escolha do algoritmo de geração de número aleatórios deve levar em consideração a qualidade dos números gerados. Um possível método de avaliação é *Diehard Battery of Tests of Randomness* [147], o algoritmo *Mersenne Twister* passa em todos os testes da bateria.

Os demais parâmetros específicos de cada algoritmo seguem os valores recomendados por seus autores, conforme os Capítulos 2, 3 e 4.

5.3 Análise dos resultados

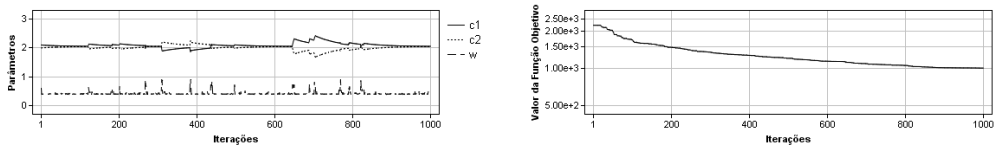
Nesta seção são analisados os resultados obtidos após a execução dos algoritmos apresentados nos Capítulos 2, 3 e 4, para todos problemas de teste apresentados neste capítulo.

São feitas as seguintes análises: Comportamento dos parâmetros dos algoritmos; Diversidade das soluções geradas; Influência da quantidade de clãs; Velocidade de convergência dos algoritmos; Desempenho dos algoritmos em função da dimensionalidade do problema; Comparação com outras técnicas baseadas em otimização por enxames de partículas colaborativas; Comparação com outras técnicas de inteligência de enxames; Análise de custo computacional.

5.3.1 Comportamento dos parâmetros dos algoritmos

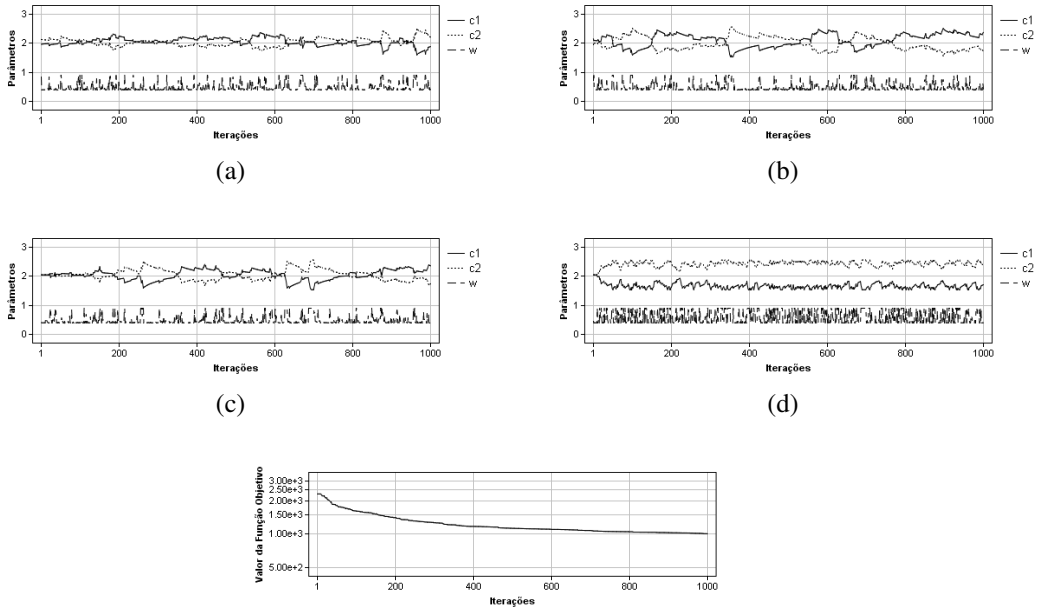
Os valores dos parâmetros c_1, c_2 e ω são ajustados a cada iteração dos algoritmos adaptativos. Portanto, é interessante observar o comportamento obtido por estes ajustes para identificar possíveis problemas do processo de adaptação. Por exemplo, verificar se ocorre estagnação dos parâmetros.

A Figura 5.9 apresenta o comportamento dos parâmetros obtido pelo APSO-30-G na função Rastrigin. A Figura 5.10 apresenta o comportamento dos parâmetros obtido pelo ClanAPSO-3x10-G na função Rastrigin. Nota-se, que a utilização de clãs ocasionou uma maior variação dos parâmetros c_1, c_2 e ω em relação ao algoritmo que não utilizas clãs. Isto sugere que a utilização de clãs pode acelerar o processo de adaptação de adaptação de parâmetros. Este comportamento também pode ser observado para outras funções de teste.



(a) (b)

Figura 5.9: Evolução dos parâmetros do APSO-30-G para função Rastrigin. (a) Comportamento dos parâmetros ao longo das iterações e (b) Comportamento do valor da função objetivo ao longo das iterações.



(a) (b)

(c) (d)

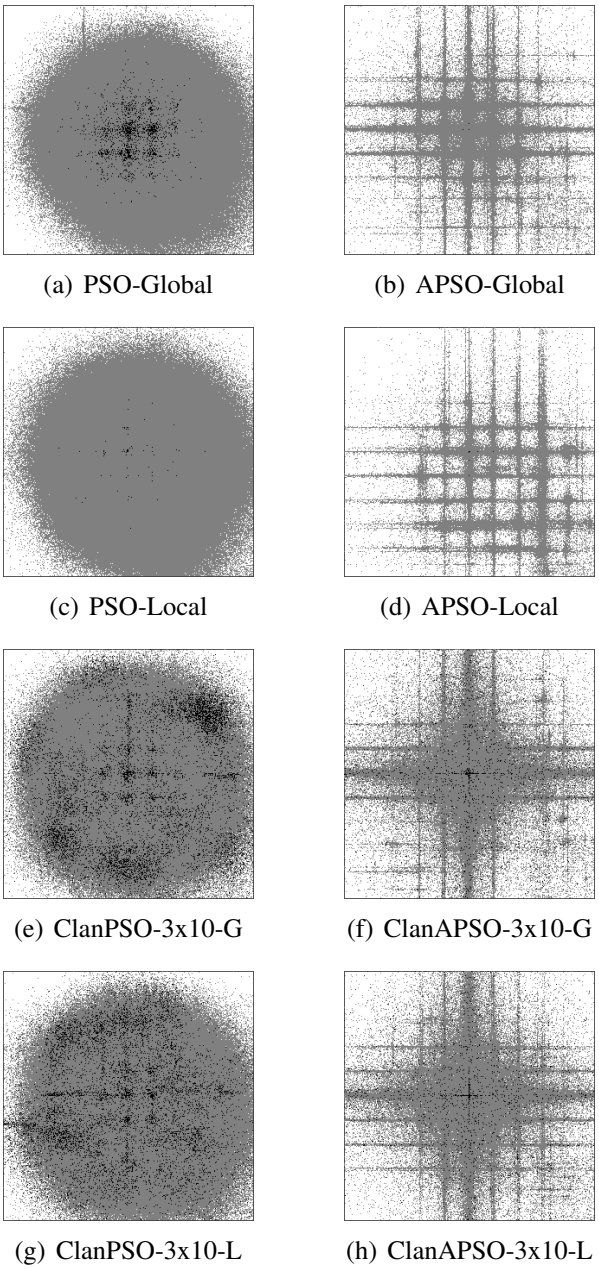
(e)

Figura 5.10: Evolução dos parâmetros do ClanAPSO-3x10-G para função Rastrigin. (a) Comportamento dos parâmetros ao longo das iterações no Clã 1, (b) Comportamento dos parâmetros ao longo das iterações no Clã 2, (c) Comportamento dos parâmetros ao longo das iterações no Clã 3, (d) Comportamento dos parâmetros ao longo das iterações nos líderes, e (e) Comportamento do valor da função objetivo ao longo das iterações.

5.3.2 Diversidade das soluções geradas

O mapa de soluções consiste em um gráfico onde são exibidas todas as soluções que são geradas por um determinado algoritmo. É uma ferramenta visual que pode ajudar a entender o comportamento do algoritmo. Também é uma maneira rápida para observação da capacidade de geração de diversidade do algoritmo. Os pontos na cor cinza representam o histórico da posição de cada partícula ao longo do processo de busca. Os pontos na cor preta representam o histórico da posição na memória cognitiva de cada partícula. O que se espera obter no mapa de soluções é que a nuvem de pontos se organize de forma semelhante ao contorno da função objetivo.

A Figura 5.11 apresenta o mapa de soluções geradas para a função Rastrigin em duas dimensões e 1000 iterações. Pode-se notar que nos algoritmos que não possuem adaptação de parâmetros, as soluções são geradas de forma espalhada no espaço de



(a) PSO-Global (b) APSO-Global

(c) PSO-Local (d) APSO-Local

(e) ClanPSO-3x10-G (f) ClanAPSO-3x10-G

(g) ClanPSO-3x10-L (h) ClanAPSO-3x10-L

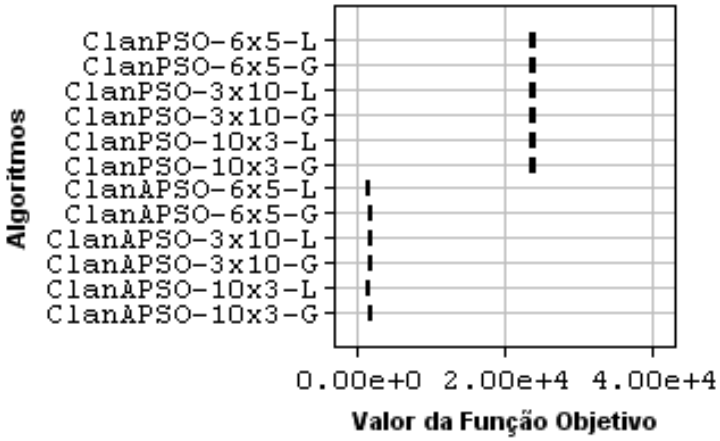
Figura 5.11: Mapa das soluções geradas para função Rastrigin em duas dimensões.

busca. Enquanto que as soluções encontradas pelas versões adaptativas se concentram nos ótimos da função. Para o algoritmo de ClanAPSO, a concentração das soluções na região central do mapa de soluções é maior que a concentração obtida pelo algoritmo de APSO. Isto é um ponto positivo, pois é onde se localiza o ponto ótimo para função Rastrigin. Além disso, o gráfico obtido pelo algoritmo ClanAPSO apresenta maior similaridade com o gráfico da função Rastrigin do que os outros algoritmos. Portanto, pode se deduzir que, para este problema, o algoritmo de ClanAPSO apresentou uma maior capacidade de exploração em profundidade em vários mínimos, devido ao processo de adaptação individual dos clãs. Isto gerou um aumento na diversidade de comportamentos das partículas e, consequentemente, o ClanAPSO obteve um número maior de regiões promissoras encontradas quando comparado com os outros algoritmos. Este comportamento também pode ser observado para outras funções de teste.

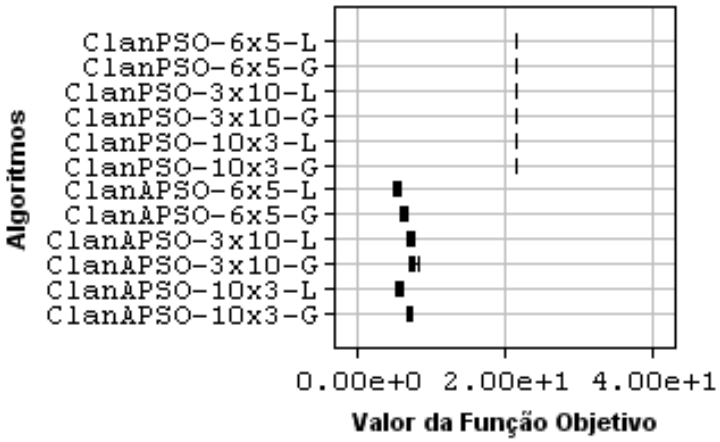
5.3.3 Influência da quantidade de clãs

A quantidade de clãs e partículas por clã pode influenciar no desempenho do algoritmo de ClanAPSO. Portanto, é interessante analisar como se comporta o desempenho do algoritmo em diferentes configurações. Visando manter o limite total de partículas, foram adotadas as configurações de 3 clãs de 10 partículas, 6 clãs de 5 partículas e 10 clãs de 3 partículas. Como referência, para esta análise também será executado o ClanPSO. Cada configuração dos algoritmos foi experimentada 30 vezes, a fim de obter dados suficientes para confecção dos gráficos de *boxplot*.

A Figura 5.12 apresenta o desempenho dos algoritmos ClanPSO e ClanAPSO para as funções de teste em 1000 dimensões. Nota-se que o algoritmo de ClanAPSO não se



(a)



(b)

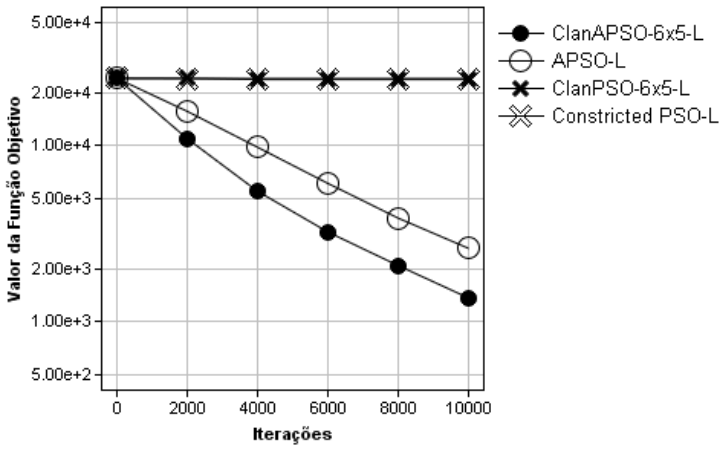
Figura 5.12: Boxplot dos resultados obtidos para os algoritmos ClanPSO e ClanAPSO 1000 dimensões e 10000 iterações. (a) Função Rastrigin. (b) Função Ackley.

demonstrou muito sensível a quantidade de clãs. Este comportamento também pode ser observado para outras funções de teste.

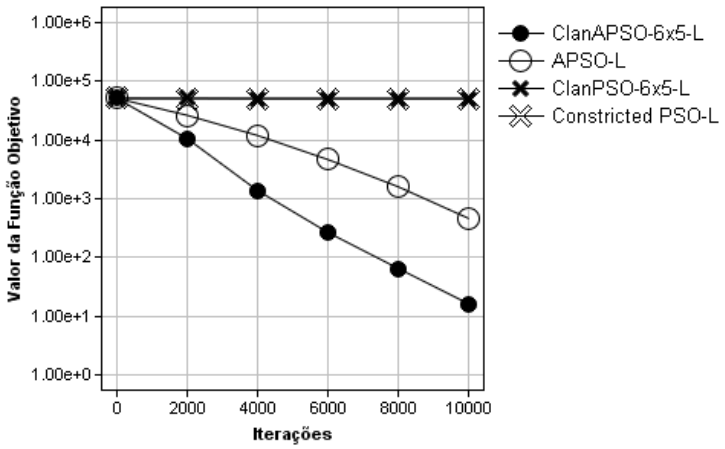
5.3.4 Velocidade de convergência dos algoritmos

Para analisar a taxa de sucesso com que cada algoritmo obtém melhores soluções ao longo das iterações, é interessante apresentar gráficos comparativos, onde alguns algoritmos tentam resolver o mesmo problema de otimização. Todos os experimentos utilizaram um total de 30 partículas e 10000 iterações. As funções objetivo foram configuradas com 1000 dimensões. Cada ponto nos gráficos representa o valor médio calculado após 30 repetições.

A Figura 5.13 apresenta a velocidade de convergência dos algoritmos para os problemas de teste. Nota-se que o algoritmo ClanAPSO-6x5-L convergiu mais



(a)



(b)

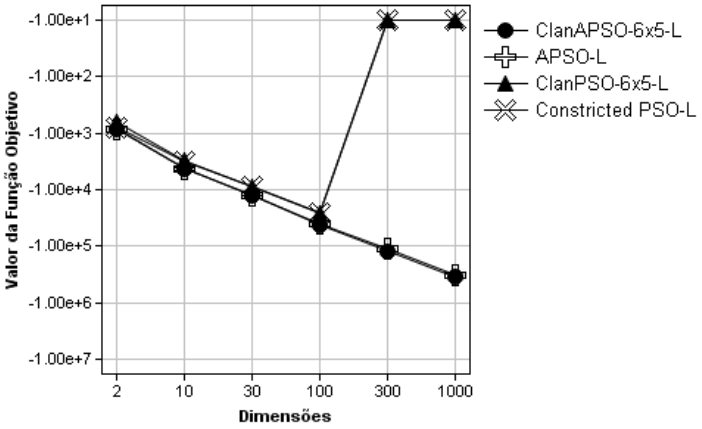
Figura 5.13: Gráfico comparativo da velocidade de convergência dos algoritmos em problemas de 1000 dimensões. (a) Função Rastrigin. (b) Função Griewank.

rapidamente que as demais variações de PSO. Este comportamento também pode ser observado para outras funções de teste.

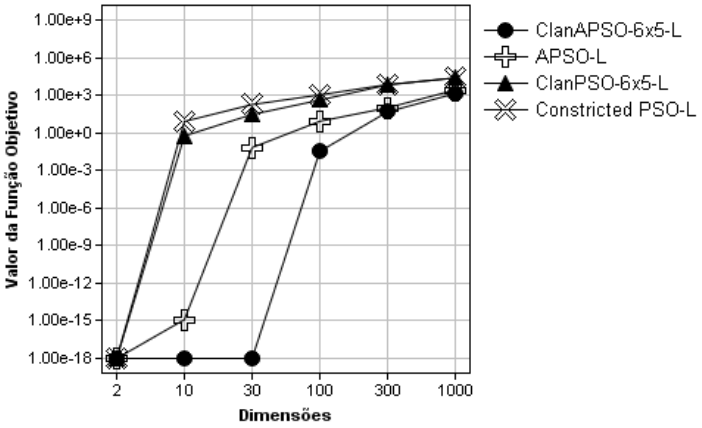
5.3.5 Desempenho dos algoritmos em função da dimensionalidade do problema

Para analisar a escabilidade dos algoritmos em relação a dificuldade da busca, é interessante apresentar gráficos do desempenho médio dos algoritmos em uma escala crescente de dimensionalidade.

Na Figura 5.14 são apresentados os resultados dos algoritmos para os problemas de teste. Nota-se, que na função Schwefel 2.26, o desempenho dos algoritmos não



(a)



(b)

Figura 5.14: Gráfico comparativo do desempenho médio dos algoritmos após 10 mil iterações em ordem crescente de dimensionalidade do problema. (a) Função Schewefel 2.26. (b) Função Rastrigin.

adaptativos sofre um grande impacto no desempenho após 100 dimensões, provavelmente a capacidade de busca desses algoritmos foi saturada. É importante salientar que, o valor da função Schwefel 2.26 varia com o número dimensões, conforme a Tabela 5.1 o valor aproximado do ótimo global é -420 vezes o número de dimensões. O algoritmo ClanAPSO-6x5-L obteve bons resultados nos demais problemas apresentados

neste capítulo.

A significância estatística (nível p) é calculada utilizando o método *Two-Sample t-test* [148][11]. É assumido $\alpha = 0,05$, ou seja, uma margem de confiança de 95%, para classificar estatisticamente os resultados como significativos ou não, quanto ao aspecto da reprodutibilidade dos resultados de cada algoritmo. A maior parte dos experimentos se mostrou estatisticamente significantes, as tabelas com os resultados para todas as dimensões experimentadas encontra-se no Apêndice B.

5.3.6 Comparação com outras técnicas baseadas em otimização por enxames de partículas colaborativas

Nesta seção é realizada uma análise comparativa do algoritmo de ClanAPSO com outras técnicas colaborativas baseadas em otimização por enxame de partículas. Nesta análise, o ClanAPSO é comparado com os algoritmos MCPSOCOL e CPSOS, porque estes algoritmos também fazem uso de multi-enxames colaborativos em seus processos internos.

Nos gráficos a seguir são apresentados os valores médios calculados após 30 repetições da melhor solução encontrada por cada algoritmo ao longo das iterações. Foram feitas simulações para 100 e 300 dimensões, no intuito de verificar a escalabilidade do algoritmo com o aumento da dimensionalidade das funções de teste.

As Figuras 5.15 e 5.16 apresentam o desempenho dos algoritmos para os problemas de teste. O algoritmo MCPSOCOL foi com configurado com 6 enxames de 5 partículas e com os parâmetros $c_1 = c_2 = c_3 = 2$ e $\omega = 0,6$. O número de enxames do algoritmo CPSOS varia com o número de dimensões, os experimentos foram realizados utilizando 3 partículas por enxame em cada dimensão. Portanto, para 100 dimensões, o CPSOS possui 300 partículas e para 300 dimensões o CPSOS possui 900 partículas. Para uma competição justa seria necessário reconfigurar o algoritmo ClanAPSO para executar com o mesmo número de partículas. No entanto, nota-se, que considerando os resultados obtidos, o algoritmo de ClanAPSO, mesmo em desvantagem em relação ao número de partículas, é superior aos demais algoritmos nos problemas simulados nesta seção.

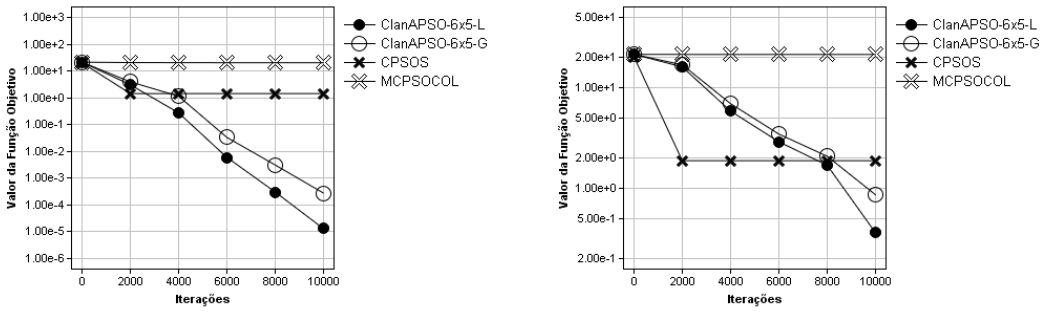
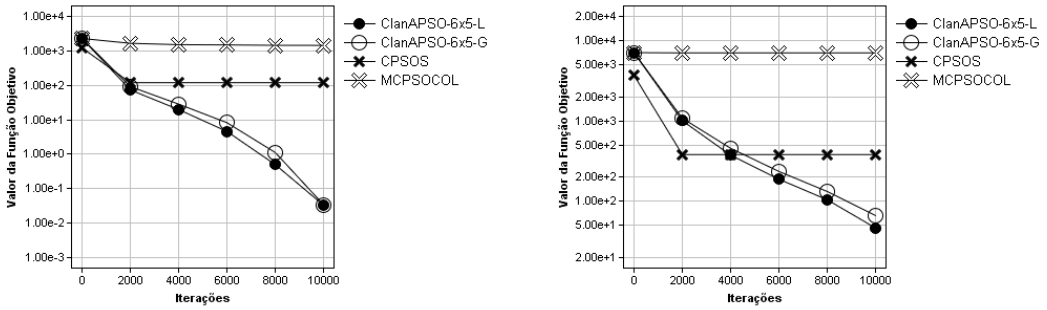


Figura 5.15: Gráfico comparativo entre ClanAPSO, MCPSOCOL e CPSOS na função Ackley. (a) 100 dimensões. (b) 300 dimensões.



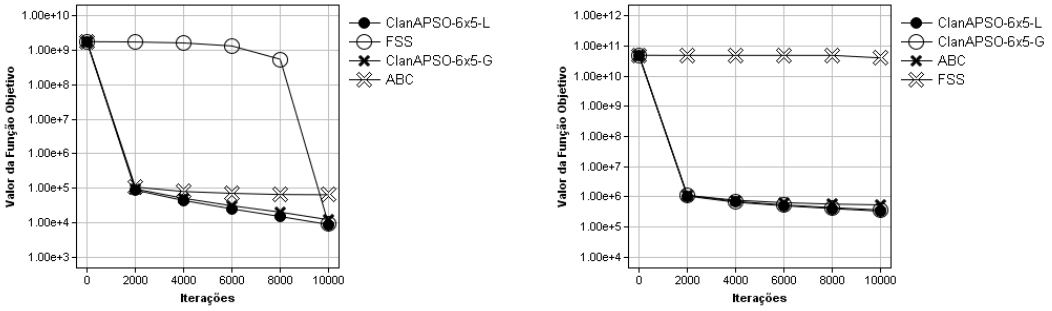
(a) (b)

Figura 5.16: Gráfico comparativo entre ClanAPSO, MCPSOCOL e CPSOS na função Rastrigin. (a) 100 dimensões. (b) 300 dimensões.

5.3.7 Comparação com outras técnicas de inteligência de enxames

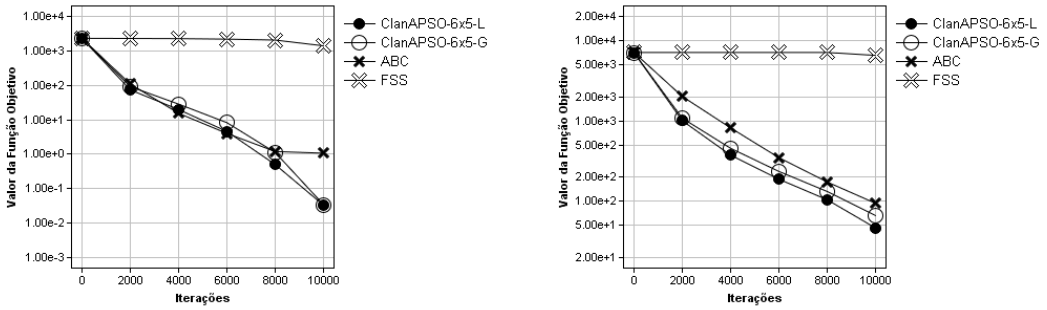
Nesta seção é apresentada uma análise comparativa do algoritmo de ClanAPSO com outras técnicas de inteligência de enxames. A comparação com outras técnicas é importante porque ajuda a identificar cenários onde é melhor se aplicar uma determinada técnica em detrimento de outra. No intuito de verificar a velocidade de convergência do algoritmo ClanAPSO, foram escolhidos os algoritmos de FSS e ABC. Estes algoritmos foram escolhidos porque possuem inspirações diferentes do modelo de partículas.

Nas Figuras 5.17 e 5.18 são apresentados os valores médios calculados após 30 repetições da melhor solução encontrada por cada algoritmo ao longo das iterações. Nota-se, que considerando os resultados obtidos, o algoritmo de ClanAPSO é superior aos demais algoritmos nos problemas simulados nesta seção.



(a) (b)

Figura 5.17: Gráfico comparativo entre ClanAPSO, FSS e ABC na função Schwefel-1.2. (a) 100 dimensões. (b) 300 dimensões.



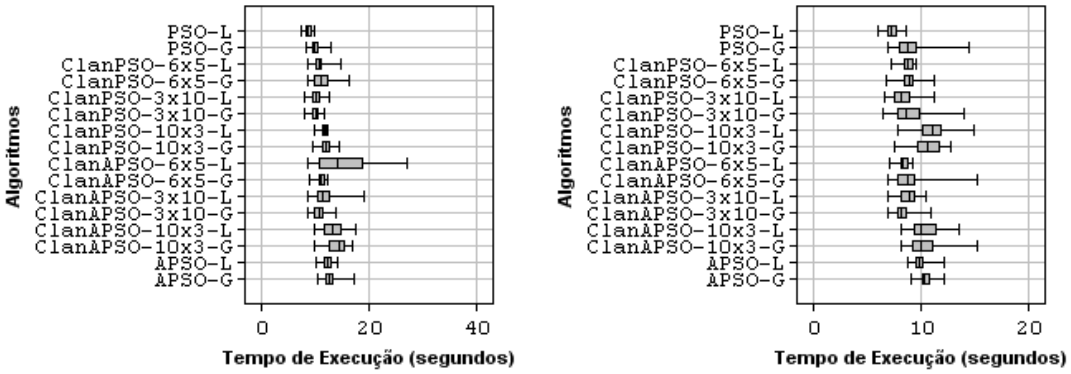
(a) (b)

Figura 5.18: Gráfico comparativo entre ClanAPSO, FSS e ABC na função Rastrigin. (a) 100 dimensões. (b) 300 dimensões.

5.3.8 Análise de custo computacional

A análise de custo computacional realizada nesta seção tem como objetivo observar o tempo médio de execução dos algoritmos. Foram utilizadas 10 mil iterações para cada algoritmo. Cada algoritmo foi executado 30 vezes utilizando 30 e 100 dimensões.

A Figura 5.19 apresenta os tempos de execução dos algoritmos os problemas de teste. O algoritmo proposto nesta dissertação, apresentou tempo de execução menor do que o algoritmo APSO em todos os experimentos realizados, isso se deve a menor quantidade requerida de cálculos de distância euclidiana no ClanAPSO, conforme detalhado no Capítulo 4.



(a) (b)

Figura 5.19: Gráfico comparativo do tempo de execução dos algoritmos. (a) Boxplot do tempo de execução dos algoritmos para função Rastrigin em 100 dimensões. (b) Boxplot do tempo de execução dos algoritmos para Rosenbrock em 100 dimensões.

Capítulo 6

Considerações Finais

Este capítulo apresenta as conclusões obtidas dos resultados experimentais do algoritmo Otimização por Enxames de Partículas utilizando Clãs com Comportamento Adaptativo, as discussões sobre resultados e comparações realizadas e uma lista de possíveis trabalhos futuros para dar continuidade às ideias deste dissertação.

6.1 Conclusões

O algoritmo de Otimização por Enxames de Partículas utilizando Clãs com Comportamento Adaptativo (ClanAPSO, *Clan Adaptive Particle Swarm Optimization*), proposto nesta dissertação, se baseia na ideia de dotar clãs de partículas com um processo de adaptação de parâmetros. O processo de adaptação de parâmetros tenta resolver questões sobre a velocidade de convergência e capacidade de escapar de ótimos locais. A abordagem colaborativa de clãs oferece um meio para que populações independentes resolvam um mesmo problema de forma distribuída. Quando ambos os conceitos são combinados e incorporados em um algoritmo, o resultado é um novo algoritmo que apresenta maior diversidade de soluções, rápida convergência e maior capacidade de escapar de ótimos locais.

Durante a análise de adaptação de parâmetros, notou-se que diferentes clãs podem estar operando em diferentes estados evolucionários durante a execução do ClanAPSO. Esse é um ponto positivo, pois promove a diversidade de comportamentos das partículas (soluções), evitando, assim, que o algoritmo fique estagnado em uma região sub-ótima.

Na análise de soluções geradas, pode ser notado que a inclusão da estratégia adaptativa baseada em estado evolucionário permitiu ao ClanAPSO obter uma melhor capacidade de exploração em amplitude em áreas promissoras. Enquanto os algoritmos não adaptativos apresentaram um comportamento esparso sem priorizar as melhores regiões. Nota-se também que adaptação de parâmetros nos clãs promoveu uma maior capacidade de busca em profundidade nas regiões próximas ao ponto ótimo das funções. Pode-se perceber que a variação dos parâmetros foi maior no ClanAPSO do que no APSO.

No estudo da influência da quantidade de clãs foi observado que o ClanAPSO obteve desempenho médio superior ao ClanPSO em todos os casos experimentados. Nota-se também que os algoritmos ClanPSO e ClanAPSO não se demonstram muito sensíveis às variações de número de clãs.

A análise comparativa com outras variantes de PSO o algoritmo de ClanAPSO demonstrou desempenho superior em todos os problemas experimentados. Em especial nos problemas de teste com mais alta dimensionalidade. Quando comparado a outras

técnicas colaborativas baseadas em otimização por enxames de partículas e outras técnicas de inteligência de enxame como busca baseada em cardumes e otimização por colmeias artificias, o ClanAPSO também apresentou resultados superiores.

6.2 Discussões

O algoritmo ClanAPSO apresentou resultados melhores na maioria do experimentos realizados no Capítulo 5 em comparação com os algoritmos abordados. Durante o desenvolvimento desta dissertação, foram realizados experimentos utilizando intercalação de algumas das técnicas abordadas, mas os resultados foram inferiores aos obtidos pelo algoritmo proposto no Capítulo 4.

Vale ressaltar, que o algoritmo de FSS apresentava resultados melhores do que as variações mais básicas PSO para funções multimodais [9], mas com o surgimento do ClanAPSO o algoritmo FSS foi superado.

6.3 Trabalhos Futuros

Muito trabalho ainda pode ser realizado numa possível extensão desta pesquisa, como, por exemplo:

- Investigar a utilização de clãs dinâmicos com adaptação baseada em estado evolucionário, ou seja, verificar a influência da migração de partículas entre enxames em um dado estado evolucionário;
- Um segundo nível de hibridização do algoritmo através da inclusão de enxames heterogêneos, onde partículas, abelhas e peixes possam interagir em um ecossistema artificial de otimização e busca;
- Realizar testes com problemas de visão computacional, como, por exemplo: detecção e rastreamento de objetos, realidade aumentada, controle de tráfego de automóveis e outros;
- Realizar um estudo comparativo de implementações em CPUs e GPUs, utilizando tecnologia *CUDA* da Nvidia, *Stream* da AMD e *OpenCL* da Khronos Group.

Referências

[1] Payman Arabshahi. *A huge swarm of red-billed queleas returns to the communal roost at dusk, Okavango Delta, Botswana.* Disponível em: http://staff.washington.edu/paymana/swarm/, Acessado em: 20/03/2010.

[2] Carl Zimmer. *A school of bigeye trevally in Malaysia.* Disponível em: http://www.nytimes.com/2007/11/13/science/13traff.html?pagewanted=all, Acessado em: 20/03/2010.

[3] Ken Thompson. *Swarm intelligence and business process optimization*. Disponível em: http://www.bioteams.com/2006/09/04/swarm_intelligence_and.html, Acessado em: 20/03/2010.

[4] X.S. Yang. Introduction to mathematical optimization: from linear programming to metaheuristics. 2008.

[5] G. Beni and J. Wang. Swarm intelligence in cellular robotic systems. *NATO ASI SERIES F COMPUTER AND SYSTEMS SCIENCES*, 102:703–703, 1989.

[6] S. Wolfram and M. Gad-el Hak. A new kind of science. *Applied Mechanics Reviews*, 56:B18, 2003.

[7] R. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. In *Micro Machine and Human Science, 1995. MHS'95. Proceedings of the Sixth International Symposium on*, pages 39–43, 1995.

[8] J. Kennedy and R. C. Eberhart. Particle swarm optimization. In *Neural Networks, 1995. Proceedings., IEEE International Conference on*, volume 4, 1995.

[9] Carmelo J. A. Bastos Filho, Fernando Buarque de Lima Neto, Anthony J. C. C. Lins, Antonio I. S. Nascimento, and Marilia P. Lima. A Novel Search Algorithm based on Fish School Behavior. In *IEEE International Conference on Systems, Man, and Cybernetics, 2008. IEEE SMC2008.*, pages 2646–2651, 2008.

[10] D. Karaboga. An idea based on honey bee swarm for numerical optimization. *Techn. Rep. TR06, Erciyes Univ. Press, Erciyes*, 2005.

[11] W.H. Press, W.T. Vetterling, S.A. Teukolsky, and B.P. Flannery. *Numerical recipes in C++: the art of scientific computing*. Cambridge University Press New York, NY, USA, 2001.

[12] R.E. Bellman and Rand Corporation. *Dynamic programming*. Rand Corporation research study. Princeton University Press, 1957.

[13] R.E. Bellman. *Dynamic Programming*. Dover Books on Mathematics. Dover Publications, 2003.

[14] R.E. Bellman. *Adaptive control processes: a guided tour*. A Rand Corporation Research Study Series. Princeton University Press, 1961.

[15] G. Hughes. On the mean accuracy of statistical pattern recognizers. *Information Theory, IEEE Transactions on*, 14(1):55–63, 2002.

[16] D. H. Wolpert, W. G. Macready, I. B. M. A. R. Center, and C. A. San Jose. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997.

[17] D.H. Wolpert, W.G. Macready, N.A.R. Center, and CA Moffett Field. Coevolutionary free lunches. *IEEE Transactions on Evolutionary Computation*, 9(6):721–735, 2005.

[18] G. Evans, J.M. Blackledge, and P. Yardley. *Numerical methods for partial differential equations*. Springer London, 2000.

[19] D. F. Carvalho and C. J. A. Bastos Filho. Clan Particle Swarm Optimization. In *IEEE Congress on Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence)*, pages 3044–3051, 2008.

[20] Z. H. Zhan, J. Zhang, Y. Li, and H. S. H. Chung. Adaptive particle swarm optimization. *IEEE Transactions on Systems Man, and Cybernetics-Part B: Cybernetics*, 39(6):1362–1381, 2009.

[21] J. Kennedy. The particle swarm: social adaptation of knowledge. In *IEEE International Conference on Evolutionary Computation, 1997.*, pages 303–308, 1997.

[22] Y. Shi and R. C. Eberhart. A modified particle swarm optimizer. In *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, pages 69–73, 1998.

[23] J. Kennedy and R. C. Eberhart. *Swarm intelligence*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2001.

[24] J. Kennedy and R. Mendes. Population structure and particle swarm performance. *Proceedings of the Evolutionary Computation on*, 1:1671–1676, 2002.

[25] J. Kennedy and R. Mendes. Neighborhood topologies in fully informed and best-of-neighborhood particle swarms. *IEEE Transcations on Systems Man and Cybernetics Part C Applications and Reviews*, 36(4):515, 2006.

[26] D. Bratton and J. Kennedy. Defining a Standard for Particle Swarm Optimization. In *Swarm Intelligence Symposium, 2007. SIS 2007. IEEE*, pages 120–127, 2007.

[27] Y. Shi, R. C. Eberhart, and I. N. Kokomo. Fuzzy adaptive particle swarm optimization. In *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, volume 1, 2001.

[28] Y. Shi, R. C. Eberhart, et al. Empirical study of particle swarm optimization. In *Proceedings of the 1999 Congress on Evolutionary Computation*, volume 3, pages 1948–1950. Piscataway, NJ, USA: IEEE, 1999.

[29] R. C. Eberhart and Y. Shi. Particle swarm optimization: developments, applications and resources. In *Proceedings of the 2001 congress on evolutionary computation*, volume 1, pages 81–86. Piscataway, NJ, USA: IEEE, 2001.

[30] A. P. Engelbrecht. *Fundamentals of computational swarm intelligence*. John Wiley & Sons, 2006.

[31] M. Clerc. The swarm and the queen: towards a deterministic and adaptive particle swarm optimization. In *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, volume 3, 1999.

[32] M. Clerc and J. Kennedy. The particle swarm-explosion, stability, and convergence in amultidimensional complex space. *IEEE Transactions on Evolutionary Computation*, 6(1):58–73, 2002.

[33] F. Van Den Bergh and A. P. Engelbrecht. A new locally convergent particle swarm optimizer. In *Proceedings of the IEEE Conference on Systems, Man, and Cybernetics, Hammamet, Tunisia*, 2002.

[34] E. S. Peer, F. Van Den Bergh, and A. P. Engelbrecht. Using neighbourhoods with the guaranteed convergence PSO. In *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE*, pages 235–242, 2003.

[35] J. Kennedy and R. C. Eberhart. A discrete binary version of the particle swarm algorithm. In *1997 IEEE International Conference on Systems, Man, and Cybernetics, 1997.'Computational Cybernetics and Simulation'.*, volume 5, 1997.

[36] R. C. Eberhart and Y. Shi. Tracking and optimizing dynamic systems with particle swarms. In *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, volume 1, 2001.

[37] K. E. Parsopoulos and M. N. Vrahatis. Particle swarm optimization method in multiobjective problems. In *Proceedings of the 2002 ACM symposium on Applied Computing*, pages 603–607. ACM New York, NY, USA, 2002.

[38] C. A. C. Coello, G. T. Pulido, and M. S. Lechuga. Handling multiple objectives with particle swarm optimization. *IEEE Transactions Evolutionary Computation*, 8(3):256–279, 2004.

[39] S. Mostaghim and J. Teich. Covering Pareto-optimal fronts by subswarms in multi-objective particle swarm optimization. In *2004 Congress on Evolutionary Computation (CEC'2004)*, volume 2, pages 1404–1411. Citeseer, 2004.

[40] K. E. Parsopoulos, D. K. Tasoulis, and M. N. Vrahatis. Multiobjective optimization using parallel vector evaluated particle swarm optimization. In *Proceedings of the IASTED International Conference on Artificial Intelligence and Applications (AIA 2004*, pages 823–828. Citeseer, 2004.

[41] S. Ho, S. Yang, G. Ni, E. Lo, and H. C. Wong. A particle swarm optimization-based method for multiobjective design optimizations. *IEEE Transactions on magnetics*, 41(5):1756–1759, 2005.

[42] M. Reyes-Sierra and C. A. C. Coello. Multi-Objective Particle Swarm Optimizers: A Survey of the State-of-the-Art. *International Journal of Computational Intelligence Research*, 2(3):287–308, 2006.

[43] R. A. Santana, M. R. Pontes, and C. J. A. Bastos Filho. A Multiple Objective Particle Swarm Optimization Approach Using Crowding Distance and Roulette Wheel. In *2009 Ninth International Conference on Intelligent Systems Design and Applications*, pages 237–242. IEEE, 2009.

[44] P. J. Angeline, N. S. Inc, and N. Y. Vestal. Using selection to improve particle swarm optimization. In *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence. The 1998 IEEE International Conference on*, pages 84–89, 1998.

[45] E. Ozcan and C. K. Mohan. Analysis of a simple particle swarm optimization system. *Intelligent Engineering Systems Through Artificial Neural Networks*, 8:253–258, 1998.

[46] J. Kennedy. Stereotyping: improving particle swarm performance with clusteranalysis. In *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on*, volume 2, 2000.

[47] R. C. Eberhart and Y. Shi. Comparing inertia weights and constriction factors in particle swarm optimization. In *Proceedings of the 2000 Congress on Evolutionary Computation*, volume 1, pages 84–88. Piscataway, NJ, USA: IEEE, 2000.

[48] R. Mendes, J. Kennedy, and J. Neves. Watch thy neighbor or how the swarm can learn from its environment. In *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE*, pages 88–94, 2003.

[49] R. Mendes, J. Kennedy, and J. Neves. The fully informed particle swarm: Simpler, maybe better. *IEEE Transactions on Evolutionary Computation*, 8(3):204–210, 2004.

[50] C. J. A. Bastos-Filho, M. P. Caraciolo, P. B. C. Miranda, and D. F. Carvalho. Multi-ring Particle Swarm Optimization. In *Proceedings of the 2008 10th Brazilian Symposium on Neural Networks-Volume 00*, pages 111–116. IEEE Computer Society Washington, DC, USA, 2008.

[51] J. Sun, W. Xu, and B. Feng. A global search strategy of quantum-behaved particle swarm optimization. In *2004 IEEE Conference on Cybernetics and Intelligent Systems*, volume 1, 2004.

[52] S. Yang, M. Wang, and L. Jiao. A quantum particle swarm optimization. In *Proceeding of the 2004 IEEE Congress on Evolutionary Computation*, volume 1, pages 320–324, 2004.

[53] J. Sun, B. Feng, and W. Xu. Particle swarm optimization with particles having quantum behavior. In *Evolutionary Computation, 2004. CEC2004. Congress on*, volume 1, 2004.

[54] J. Sun, W. Xu, and B. Feng. Adaptive parameter control for quantum-behaved particle swarm optimization on individual level. In *2005 IEEE International Conference on Systems, Man and Cybernetics*, volume 4, 2005.

[55] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. *IEEE Transactions on Evolutionary Computation*, 8(3):240–255, 2004.

[56] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *IEEE Transactions on Evolutionary Computation*, 10(3):281–295, 2006.

[57] N. P. Suraweera and D. N. Ranasinghe. A Natural Algorithmic Approach to the Structural Optimisation of Neural Networks. In *Information and Automation for Sustainability, 2008. ICIAFS 2008. 4th International Conference on*, pages 150–156, 2008.

[58] S. Kiranyaz, T. Ince, A. Yildirim, and M. Gabbouj. Evolutionary artificial neural networks by multi-dimensional particle swarm optimization. *Neural Networks*, 1:1, 2009.

[59] M. Carvalho and T. B. Ludermir. Particle swarm optimization of feed-forward neural networks with weight decay. In *Hybrid Intelligent Systems, 2006. HIS'06. Sixth International Conference on*, pages 5–5, 2006.

[60] L. Zhao and Y. Yang. PSO-based single multiplicative neuron model for time series prediction. *Expert Systems with Applications*, 36(2P2):2805–2812, 2009.

[61] A. Chatterjee, K. Pulasinghe, K. Watanabe, and K. Izumi. A particle-swarm-optimized fuzzy-neural network for voice-controlled robot systems. *IEEE Transactions on Industrial Electronics*, 52(6):1478–1489, 2005.

[62] C. Zhang, H. Shao, and Y. Li. Particle swarm optimisation for evolving artificial neural network. In *2000 IEEE International Conference on Systems, Man, and Cybernetics*, volume 4, 2000.

[63] W. Z. Lu, H. Y. Fan, A. Y. T. Leung, and J. C. K. Wong. Analysis of pollutant levels in central Hong Kong applying neural network method with particle swarm optimization. *Environmental monitoring and assessment*, 79(3):217–230, 2002.

[64] I. N. Kassabalidis, M. A. El-Sharkawi, R. J. Marks, L. S. Moulin, and A. P. Alves da Silva. Dynamic security border identification using enhanced particle swarm optimization. *IEEE Transactions on Power Systems*, 17(3):723–729, 2002.

[65] V. Tandon, H. El-Mounayri, and H. Kishawy. NC end milling optimization using evolutionary computation. *International Journal of Machine Tools and Manufacture*, 42(5):595–605, 2002.

[66] R. Mendes, P. Cortez, M. Rocha, and J. Neves. Particle swarms for feedforward neural network training. *learning*, 6:1, 2002.

[67] W. Z. Lu, H. Y. Fan, and S. M. Lo. Application of evolutionary neural network method in predicting pollutant levels in downtown area of Hong Kong. *Neurocomputing*, 51(1):387–400, 2003.

[68] C. F. Juang. A hybrid of genetic algorithm and particle swarm optimization for recurrent network design. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 34(2):997–1006, 2004.

[69] L. Messerschmidt and A. P. Engelbrecht. Learning to play games using a PSO-based competitive learning approach. *IEEE Transactions on Evolutionary Computation*, 8(3):280–288, 2004.

[70] Y. Da and G. Xiurun. An improved PSO-based ANN with simulated annealing technique. *Neurocomputing*, 63:527–533, 2005.

[71] K. W. Chau. Particle swarm optimization training algorithm for ANNs in stage prediction of Shing Mun River. *Journal of hydrology*, 329(3-4):363–367, 2006.

[72] J. R. Zhang, J. Zhang, T. M. Lok, and M. R. Lyu. A hybrid particle swarm optimization–back-propagation algorithm for feedforward neural network training. *Applied Mathematics and Computation*, 185(2):1026–1037, 2007.

[73] H. Yoshida, K. Kawata, Y. Fukuyama, S. Takayama, and Y. Nakanishi. A particle swarm optimization for reactive power and voltagecontrol considering voltage security assessment. *IEEE Transactions on Power Systems*, 15(4):1232–1239, 2000.

[74] Y. Fukuyama and H. Yoshida. A particle swarm optimization for reactive power and voltage control in electric power systems. In *Proceedings of the 2001 Congress on Evolutionary Computation (CEC-2001)*, pages 87–93, 2001.

[75] M. A. Abido. Optimal power flow using particle swarm optimization. *International Journal of Electrical Power and Energy Systems*, 24(7):563–571, 2002.

[76] M. A. Abido. Optimal design of power-system stabilizers using particle swarm optimization. *IEEE Transactions on Energy Conversion*, 17(3):406–413, 2002.

[77] Z. L. Gaing. A particle swarm optimization approach for optimum design of PID controller in AVR system. *IEEE Transaction on Energy Conversion*, 19(2):384–391, 2004.

[78] M. R. Alrashidi and M. E. El-Hawary. A survey of particle swarm optimization applications in electric power systems. *IEEE Transactions on Evolutionary Computation*, 1:1, 2006.

[79] J. Chuanwen and E. Bompard. A hybrid method of chaotic particle swarm optimization and linear interior for reactive power optimisation. *Mathematics and Computers in Simulation*, 68(1):57–65, 2005.

[80] A. A. A. Esmin, G. Lambert-Torres, and A. C. Zambroni de Souza. A hybrid particle swarm optimization applied to loss power minimization. *IEEE Transactions on Power Systems*, 20(2):859–866, 2005.

[81] Z. Bo and C. A. O. Yi-jia. A multi-agent particle swarm optimization algorithm for reactive power optimization. *Proceedings of the CSEE*, 5:1, 2005.

[82] J. G. Vlachogiannis and K. Y. Lee. A comparative study on particle swarm optimization for optimal steady-state performance of power systems. *IEEE Transactions on Power Systems*, 21(4):1718–1728, 2006.

[83] Y. del Valle, G. K. Venayagamoorthy, S. Mohagheghi, J. Hernandez, and R. G. Harley. Particle swarm optimization: basic concepts, variants and applications in power systems. *IEEE Transactions on Evolutionary Computation*, 12(2):171, 2008.

[84] S. Naka, T. Genji, T. Yura, and Y. Fukuyama. Practical distribution state estimation using hybrid particle swarm optimization. In *Proceedings of the IEEE Power Engineering Society Transmission and Distribution Conference*, volume 2, pages 815–820, 2001.

[85] B. Brandstatter and U. Baumgartner. Particle swarm optimization-mass-spring system analogon. *IEEE Transactions on magnetics*, 38(2 Part 1):997–1000, 2002.

[86] S. Naka, T. Genji, T. Yura, and Y. Fukuyama. A hybrid particle swarm optimization for distribution state estimation. *IEEE Transactions on Power Systems*, 18(1):60–68, 2003.

[87] Z. L. Gaing. Particle swarm optimization to solving the economic dispatch considering the generator constraints. *IEEE Transactions on Power Systems*, 18(3):1187–1195, 2003.

[88] X. Yu, X. Xiong, and Y. Wu. A PSO-based approach to optimal capacitor placement with harmonic distortion consideration. *Electric Power Systems Research*, 71(1):27–33, 2004.

[89] S. P. Ghoshal. Optimizations of PID gains by particle swarm optimizations in fuzzy based automatic generation control. *Electric Power Systems Research*, 72(3):203–212, 2004.

[90] J. B. Park, K. S. Lee, J. R. Shin, and K. Y. Lee. A particle swarm optimization for economic dispatch with nonsmooth cost functions. *IEEE Transactions on Power Systems*, 20(1):34–42, 2005.

[91] C. M. Huang, C. J. Huang, and M. L. Wang. A particle swarm optimization to identifying the ARMAX model for short-term load forecasting. *IEEE Transactions on Power Systems*, 20(2):1126–1133, 2005.

[92] J. Chuanwen and E. Bompard. A self-adaptive chaotic particle swarm algorithm for short term hydroelectric system scheduling in deregulated environment. *Energy Conversion and Management*, 46(17):2689–2696, 2005.

[93] S. Cui and D.S. Weile. Application of a parallel particle swarm optimization scheme to the design of electromagnetic absorbers. *IEEE Transactions on Antennas and Propagation*, 53(11):3616–3624, 2005.

[94] M. Donelli and A. Massa. Computational approach based on a particle swarm optimizer for microwave imaging of two-dimensional dielectric scatterers. *IEEE Transactions on Microwave Theory and Techniques*, 53(5):1761–1776, 2005.

[95] H. Zhang, X. Li, H. Li, and F. Huang. Particle swarm optimization-based schemes for resource-constrained project scheduling. *Automation in Construction*, 14(3):393–404, 2005.

[96] T. A. A. Victoire and A. E. Jeyakumar. Reserve constrained dynamic dispatch of units with valve-point effects. *IEEE Transactions on Power Systems*, 20(3):1273–1282, 2005.

[97] C. F. Juang and C. H. Hsu. Temperature control by chip-implemented adaptive recurrent fuzzy controller designed by evolutionary algorithm. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 52(11):2376–2384, 2005.

[98] T. O. Ting, M. V. C. Rao, and C. K. Loo. A novel approach for unit commitment problem via an effective hybrid particle swarm optimization. *IEEE Transactions on Power Systems*, 21(1):411–418, 2006.

[99] S. Ho, S. Yang, G. Ni, and H. C. Wong. A particle swarm optimization method with enhanced global search ability for design optimizations of electromagnetic devices. *IEEE Transactions on magnetics*, 42(4):1107–1110, 2006.

[100] D. N. Jeyakumar, T. Jayabarathi, and T. Raghunathan. Particle swarm optimization for various types of economic dispatch problems. *International Journal of Electrical Power and Energy Systems*, 28(1):36–42, 2006.

[101] A. I. Selvakumar and K. Thanushkodi. A new particle swarm optimization solution to nonconvex economic dispatch problems. *IEEE Transactions on Power Systems*, 22(1):42–51, 2007.

[102] Q. He and L. Wang. An effective co-evolutionary particle swarm optimization for constrained engineering design problems. *Engineering Applications of Artificial Intelligence*, 20(1):89–99, 2007.

[103] J. S. Heo, K. Y. Lee, and R. Garduno-Ramirez. Multiobjective control of power plants using particle swarm optimization techniques. *IEEE Transactions on Energy Conversion*, 21(2):552–561, 2006.

[104] W. Xia and Z. Wu. An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems. *Computers & Industrial Engineering*, 48(2):409–425, 2005.

[105] G. Venter and J. Sobieszczanski-Sobieski. Multidisciplinary optimization of a transport aircraft wing using particle swarm optimization. *Structural and Multidisciplinary Optimization*, 26(1):121–131, 2004.

[106] J. Robinson, S. Sinton, and Y. Rahmat-Samii. Particle swarm, genetic algorithm, and their hybrids: optimization of a profiled corrugated horn antenna. In *IEEE antennas and propagation society international symposium*, volume 1, pages 314–317. IEEE; 1999, 2002.

[107] D. Gies and Y. Rahmat-Samii. Particle swarm optimization for reconfigurable phase-differentiated array design. *Microwave and Optical Technology Letters*, 38(3):168–175, 2003.

[108] D. W. Boeringer and D. H. Werner. Particle swarm optimization versus genetic algorithms for phased array synthesis. *IEEE Transactions on Antennas and Propagation*, 52(3):771–779, 2004.

[109] M.M. Khodier and C.G. Christodoulou. Linear array geometry synthesis with minimum sidelobe level and null control using particle swarm optimization. *IEEE Transactions on Antennas and Propagation*, 53(8 Part 2):2674–2679, 2005.

[110] N. Jin and Y. Rahmat-Samii. Advances in particle swarm optimization for antenna designs: Real-number, binary, single-objective and multiobjective implementations. *IEEE Transactions on Antennas and Propagation*, 55(3 Part 1):556–567, 2007.

[111] P. Y. Yin. A discrete particle swarm algorithm for optimal polygonal approximation of digital curves. *Journal of visual communication and image representation*, 15(2):241–260, 2004.

[112] D.K. Agrafiotis and W. Cedeno. Feature Selection for Structure- Activity Correlation Using Binary Particle Swarms. *J. Med. Chem*, 45(5):1098–1107, 2002.

[113] X. Wang, J. Yang, X. Teng, W. Xia, and R. Jensen. Feature selection based on rough sets and particle swarm optimization. *Pattern Recognition Letters*, 28(4):459–471, 2007.

[114] C. Elegbede. Structural reliability assessment based on particles swarm optimization. *Structural safety*, 27(2):171–186, 2005.

[115] T. Sousa, A. Silva, and A. Neves. Particle swarm based data mining algorithms for classification tasks. *Parallel Computing*, 30(5-6):767–783, 2004.

[116] P. C. Fourie and A. A. Groenwold. The particle swarm optimization algorithm in size and shape optimization. *Structural and Multidisciplinary Optimization*, 23(4):259–267, 2002.

[117] A. Salman, I. Ahmad, and S. Al-Madani. Particle swarm optimization for task assignment problem. *Microprocessors and Microsystems*, 26(8):363–371, 2002.

[118] S. Kannan, S. M. R. Slochanal, P. Subbaraj, and N. P. Padhy. Application of particle swarm optimization technique and its variants to generation expansion planning problem. *Electric Power Systems Research*, 70(3):203–210, 2004.

[119] X. Zhang, Y. Zheng, Y. Shen, J. Zhang, and B. Yang. Particle swarm optimization used as a control algorithm for adaptive PMD compensation. *IEEE Photonics Technology Letters*, 17(1):85–87, 2005.

[120] A. Allahverdi and F. S. Al-Anzi. A PSO and a Tabu search heuristics for the assembly scheduling problem of the two-stage distributed database application. *Computers and Operations Research*, 33(4):1056–1080, 2006.

[121] Z. Lian, B. Jiao, and X. Gu. A similar particle swarm optimization algorithm for job-shop scheduling to minimize makespan. *Applied Mathematics and Computation*, 183(2):1008–1017, 2006.

[122] C. J. Liao, C. T. Tseng, and P. Luarn. A discrete version of particle swarm optimization for flowshop scheduling problems. *Computers and Operations Research*, 34(10):3099–3111, 2007.

[123] Y. Dong, J. Tang, B. Xu, and D. Wang. An application of swarm optimization to nonlinear programming. *Computers and Mathematics with Applications*, 49(11-12):1655–1668, 2005.

[124] Y. Chen, B. Yang, and J. Dong. Time-series prediction using a local linear wavelet neural network. *Neurocomputing*, 69(4-6):449–465, 2006.

[125] M. P. Wachowiak, R. Smolíková, Y. Zheng, J. M. Zurada, and A. S. Elmaghraby. An approach to multimodal biomedical image registration utilizing particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, 8(3):289–301, 2004.

[126] E. Zahara, S. K. S. Fan, and D. M. Tsai. Optimal multi-thresholding using a hybrid optimization approach. *Pattern Recognition Letters*, 26(8):1082–1095, 2005.

[127] M. Omran, A. P. Engelbrecht, and A. Salman. Particle swarm optimization method for image clustering. *International Journal of Pattern Recognition and Artificial Intelligence*, 19(3):297–322, 2005.

[128] Y. Yuan, Z. He, and M. Chen. Virtual MIMO-based cross-layer design for wireless sensor networks. *IEEE Transactions on Vehicular Technology*, 55(3):856, 2006.

[129] C. J. A. Bastos Filho, F. B. de Lima Neto, A. J. C. C. Lins, A. I. S. Nascimento, and M. P. Lima. Fish School Search. *Nature-Inspired Algorithms for Optimisation*, 1:261, 2009.

[130] C. J. A. Bastos Filho, F. B. L. Neto, M. F. C. Sousa, M. R. Pontes, and S. S. Madeiro. On the Influence of the Swimming Operators in the Fish School Search Algorithm. In *CEC2009*, pages 111–116. CEC2009, 2009.

[131] S. S. Madeiro. Buscas multimodais por cardumes baseados em densidade. Master's thesis, University of Pernambuco, 2010.

[132] D. Karaboga and B. Basturk. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal of Global Optimization*, 39(3):459–471, 2007.

[133] R. S. Rao, S. V. L. Narasimham, and M. Ramalingaraju. Optimization of Distribution Network Configuration for Loss Reduction Using Artificial Bee Colony Algorithm. *Int J Electr Power Energy Syst Eng*, 1(2):116–122, 2008.

[134] L. Bao and J. Zeng. Comparison and Analysis of the Selection Mechanism in the Artificial Bee Colony Algorithm. In *2009 Ninth International Conference on Hybrid Intelligent Systems*, pages 411–416. IEEE, 2009.

[135] D. Karaboga and B. Basturk. On the performance of artificial bee colony (ABC) algorithm. *Applied Soft Computing*, 8(1):687–697, 2008.

[136] Sean Luke. *Essentials of Metaheuristics*. Disponível em: http://cs.gmu.edu/∼sean/book/metaheuristics/, Acessado em: 20/03/2010.

[137] C. J. A. Bastos Filho, D. F. Carvalho, E. M. N. Figueiredo, and P. B. C. de Miranda. Dynamic Clan Particle Swarm Optimization. In *2009 Ninth International Conference on Intelligent Systems Design and Applications*, pages 249–254. IEEE, 2009.

[138] F. Van Den Bergh and A. P. Engelbrecht. Cooperative learning in neural networks using particle swarm optimizers. *South African Computer Journal*, 26:84–90, 2000.

[139] F. Van Den Bergh and A. P. Engelbrecht. Training product unit networks using cooperative particle swarm optimizers. In *Proceedings of IJCNN*, pages 126–131. Citeseer, 2001.

[140] F. Van Den Bergh and A. P. Engelbrecht. A cooperative approach to particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, 8(3):225–239, 2004.

[141] F. Van Den Bergh. *An analysis of particle swarm optimizers*. PhD thesis, University of Pretoria, Pretoria, South Africa, South Africa, 2002. Supervisor-Engelbrecht, A. P.

[142] B. Niu, Y. Zhu, X. He, and H. Wu. MCPSO: A multi-swarm cooperative particle swarm optimizer. *Applied Mathematics and Computation*, 185(2):1050–1062, 2007.

[143] S. Paracer and V. Ahmadjian. *Symbiosis: an introduction to biological associations*. Oxford University Press, USA, 2000.

[144] J. Sapp. The dynamics of symbiosis: an historical overview. *Botany*, 82(8):1046–1056, 2004.

[145] X. Yao, Y. Liu, and G. Lin. Evolutionary programming made faster. *Evolutionary Computation*, 3(2):82–102, 1999.

[146] M. Matsumoto and T. Nishimura. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 8(1):3–30, 1998.

[147] G. Marsaglia. DIEHARD, a battery of tests for random number generators. *Department of Statistics and Supercomputer Computations Research Institute, Florida State University (http://stat. fsu. edu/ geo/diehard. html)*, 1995.

[148] H.T. Lawless and H. Heymann. *Sensory evaluation of food: principles and practices*. Aspen Publishers, 1999.

Apêndice A

Publicações durante o mestrado.

Título: Adaptive Clan Particle Swarm Optimization.
Autores: Murilo R. Pontes, Fernando B. Lima Neto, Carmelo J. A. Bastos-Filho.
Submetido para: 2011 IEEE Symposium Series on Computational Intelligence (IEEE SSCI 2011), 2011.
Abstract: Particle Swarm Optimization has been widely used to solve real world problems, mainly when there are too many variables to be optimized and these variables are continuous. In nature one can observe many examples of cooperative behaviors that lead to complex problem solving. Recently, some Particle Swarm Optimization variations gracefully incorporate such cooperative features with consequent beneficial new abilities. In this paper we put forward the incorporation of auto-adaptation capability in a cooperative Particle Swarm Optimization algorithm, called Clan Particle Swarm Optimization. Next, we present a deep analysis on the adaptation process for one multimodal function and evaluate the performance of our proposal in some well known benchmark problems. The results revealed that our proposal achieved better performance than other approaches, specially in tough multimodal problems.

Título: A Multiple Objective Particle Swarm Optimization Approach using Crowding Distance and Roulette Wheel.
Autores: SANTANA, R. A. ; BASTOS FILHO, C. J. A. ; PONTES, M. R.
Publicado em: Intelligent Systems Design and Applications, 2009. ISDA '09. Ninth International Conference on, 2009, Pisa, Itália. 9th International Conference on Intelligent Systems Design and Applications. Los Alamitos, CA, EUA : IEEE Press, 2009. v. 1. p. 237-242.
Abstract: This paper presents a multiobjective optimization algorithm based on Particle Swarm Optimization (MOPSO-CDR) that uses a diversity mechanism called crowding distance to select the social leaders and the cognitive leader. We also use the same mechanism to delete solutions of the external archive. The performance of our proposal was evaluated in five well known benchmark functions using four metrics previously presented in the literature. Our proposal was compared to other four multi objective optimization algorithms based on Particle Swarm Optimization, called m-DNPSO, CSS-MOPSO, MOPSO and MOPSO-CDLS. The results showed that the proposed approach is competitive when compared to the other approaches and outperforms the other algorithms in many cases.

Título: On the Influence of the Swimming Operators in the Fish School Search Algorithm.

Autores: BASTOS FILHO, C. J. A. ; LIMA NETO, F. B. ; SOUSA, M. F. C. ; PONTES, M. R. ; MADEIRO, S. S.

Publicado em: 2009 IEEE International Conference on Systems, Man, and Cybernetics, 2009, San Antonio, Texas. 2009 IEEE International Conference on Systems, Man, and Cybernetics, 2009. v. 1. p. 512-517.

Abstract: Real world engineer tasks and times series prediction are sometimes high dimensional spaces problems that are hard to compute. A common approach to tackle these challenges is to apply swarm based or evolutionary algorithms. Fish School Search (FSS) is an example of such technique that excels on difficult search problems such as these. As FSS is a novel technique only output results where investigate so far. This paper analyzes the influence of the FSS operators on the performance of the algorithm in six benchmark functions. We compared FSS results with some PSO variations and show that all operators are important and complementary. We assessed the influence of each swimming operator separately. The volitive mechanism is the operator that provides most exploration abilities to the search process. The carried out assessment has shown that, in average, the best results are obtained only when all the operators are active.

Apêndice B

Tabelas de resultados dos algoritmos.

Tabela 6.1: Resultados e significância estatística ($\alpha = 0,05$) para Schwefel-1.2.

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ABC	2	6,93e-19 (2,90e-37)	30	3,13e-07	sim
APSO-G	2	0,00e+00 (0,00e+00)	30	-	-
APSO-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-10x3-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-10x3-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-3x10-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-3x10-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-6x5-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-6x5-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-10x3-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-10x3-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-3x10-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-3x10-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-6x5-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-6x5-L	2	0,00e+00 (0,00e+00)	30	-	-
Constricted PSO-G	2	0,00e+00 (0,00e+00)	30	-	-
Constricted PSO-L	2	0,00e+00 (0,00e+00)	30	-	-
CPSOS	2	2,27e-32 (1,54e-62)	30	3,34e-01	não
FSS	2	2,68e-07 (4,79e-14)	30	1,09e-04	sim
GCPSO-G	2	0,00e+00 (0,00e+00)	30	-	-
GCPSO-L	2	0,00e+00 (0,00e+00)	30	-	-
MCPSOCOL	2	0,00e+00 (0,00e+00)	30	-	-
PSO-G	2	0,00e+00 (0,00e+00)	30	-	-
PSO-L	2	0,00e+00 (0,00e+00)	30	-	-
ABC	10	3,56e-01 (1,16e-01)	30	7,21e-06	sim
APSO-G	10	3,08e-48 (1,07e-94)	30	1,16e-01	não
APSO-L	10	1,96e-204 (0,00e+00)	30	-	-
ClanAPSO-10x3-G	10	4,27e-50 (2,87e-98)	30	1,82e-01	não
ClanAPSO-10x3-L	10	4,66e-93 (5,26e-184)	30	-	-
ClanAPSO-3x10-G	10	1,79e-64 (6,88e-127)	30	2,54e-01	não
ClanAPSO-3x10-L	10	2,05e-65 (5,99e-129)	30	1,62e-01	não
ClanAPSO-6x5-G	10	4,49e-62 (5,50e-122)	30	3,12e-01	não
ClanAPSO-6x5-L	10	1,09e-85 (2,83e-169)	30	-	-

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ClanPSO-10x3-G	10	1,24e-21 (1,57e-41)	30	1,02e-01	não
ClanPSO-10x3-L	10	3,46e-18 (2,97e-35)	30	1,06e-03	sim
ClanPSO-3x10-G	10	4,54e-19 (2,31e-36)	30	1,14e-01	não
ClanPSO-3x10-L	10	2,06e-19 (1,99e-37)	30	1,40e-02	sim
ClanPSO-6x5-G	10	2,12e-21 (5,55e-41)	30	1,32e-01	não
ClanPSO-6x5-L	10	2,49e-19 (1,33e-36)	30	2,54e-01	não
Constricted PSO-G	10	1,13e-63 (1,35e-125)	30	1,04e-01	não
Constricted PSO-L	10	3,02e-219 (0,00e+00)	30	-	-
CPSOS	10	1,20e+01 (7,27e+02)	30	2,20e-02	sim
FSS	10	5,43e-03 (1,91e-05)	30	3,14e-04	sim
GCPSO-G	10	4,76e-60 (3,42e-118)	30	1,74e-01	não
GCPSO-L	10	7,54e-232 (0,00e+00)	30	-	-
MCPSOCOL	10	7,16e+01 (1,71e+04)	30	9,98e-03	sim
PSO-G	10	1,62e-15 (1,74e-29)	30	4,14e-02	sim
PSO-L	10	3,36e-59 (2,49e-116)	30	2,60e-01	não
ABC	30	1,56e+03 (3,06e+05)	30	1,94e-08	sim
APSO-G	30	5,55e-01 (2,97e-01)	30	4,59e-04	sim
APSO-L	30	7,19e-12 (1,64e-22)	30	6,85e-03	sim
ClanAPSO-10x3-G	30	3,99e-01 (1,43e-01)	30	1,07e-03	sim
ClanAPSO-10x3-L	30	1,58e-02 (1,57e-04)	30	2,27e-04	sim
ClanAPSO-3x10-G	30	4,64e-02 (1,72e-03)	30	1,07e-05	sim
ClanAPSO-3x10-L	30	2,93e-02 (6,49e-04)	30	3,27e-05	sim
ClanAPSO-6x5-G	30	3,50e-02 (6,48e-04)	30	2,45e-06	sim
ClanAPSO-6x5-L	30	8,56e-03 (3,22e-05)	30	5,68e-06	sim
ClanPSO-10x3-G	30	1,98e+02 (4,60e+03)	30	1,23e-07	sim
ClanPSO-10x3-L	30	2,26e+02 (8,49e+03)	30	1,40e-05	sim
ClanPSO-3x10-G	30	4,06e+02 (8,48e+04)	30	4,09e-07	sim
ClanPSO-3x10-L	30	3,79e+02 (7,73e+04)	30	1,12e-06	sim
ClanPSO-6x5-G	30	2,19e+02 (1,58e+04)	30	1,12e-07	sim
ClanPSO-6x5-L	30	2,57e+02 (3,73e+04)	30	1,23e-05	sim
Constricted PSO-G	30	1,79e-01 (3,01e-01)	30	9,52e-02	não
Constricted PSO-L	30	2,96e-20 (2,41e-39)	30	2,01e-03	sim
CPSOS	30	1,01e+03 (4,42e+05)	30	7,42e-07	sim
FSS	30	7,03e+00 (9,94e+00)	30	6,13e-06	sim
GCPSO-G	30	1,74e+01 (2,83e+03)	30	8,39e-02	não
GCPSO-L	30	1,21e-20 (2,49e-39)	30	2,01e-01	não
MCPSOCOL	30	3,75e+04 (1,11e+08)	30	9,62e-08	sim
PSO-G	30	8,41e+02 (1,45e+05)	30	6,72e-08	sim
PSO-L	30	1,06e+00 (9,71e-01)	30	2,00e-04	sim
ABC	100	6,51e+04 (5,33e+07)	30	8,83e-08	sim
APSO-G	100	2,65e+04 (1,22e+07)	30	4,56e-08	sim
APSO-L	100	3,62e+03 (8,93e+05)	30	2,56e-07	sim
ClanAPSO-10x3-G	100	1,79e+04 (1,19e+07)	30	9,08e-07	sim
ClanAPSO-10x3-L	100	7,94e+03 (1,96e+06)	30	1,24e-09	sim
ClanAPSO-3x10-G	100	1,47e+04 (8,58e+06)	30	2,49e-06	sim
ClanAPSO-3x10-L	100	1,35e+04 (5,46e+06)	30	3,77e-06	sim
ClanAPSO-6x5-G	100	1,24e+04 (6,57e+06)	30	6,64e-08	sim

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ClanAPSO-6x5-L	100	8,96e+03 (2,33e+06)	30	2,80e-10	sim
ClanPSO-10x3-G	100	5,56e+04 (4,51e+07)	30	8,33e-07	sim
ClanPSO-10x3-L	100	4,93e+04 (5,30e+07)	30	1,23e-07	sim
ClanPSO-3x10-G	100	6,00e+04 (1,14e+08)	30	1,25e-07	sim
ClanPSO-3x10-L	100	5,88e+04 (1,12e+08)	30	5,88e-07	sim
ClanPSO-6x5-G	100	5,53e+04 (7,28e+07)	30	4,24e-08	sim
ClanPSO-6x5-L	100	5,50e+04 (8,19e+07)	30	3,48e-07	sim
Constricted PSO-G	100	1,88e+07 (3,02e+15)	30	7,30e-02	não
Constricted PSO-L	100	3,34e+08 (3,45e+16)	30	1,87e-07	sim
CPSOS	100	2,04e+04 (2,09e+07)	30	5,46e-08	sim
FSS	100	9,25e+03 (5,13e+06)	30	2,07e-06	sim
GCPSO-G	100	2,52e+07 (3,00e+15)	30	1,57e-02	sim
GCPSO-L	100	3,41e+08 (3,12e+16)	30	4,80e-08	sim
MCPSOCOL	100	4,44e+08 (2,57e+16)	30	1,18e-07	sim
PSO-G	100	6,29e+04 (5,74e+07)	30	1,44e-08	sim
PSO-L	100	3,86e+04 (4,37e+07)	30	2,65e-08	sim
ABC	300	5,50e+05 (1,65e+09)	30	1,45e-06	sim
APSO-G	300	5,45e+05 (2,58e+09)	30	1,17e-05	sim
APSO-L	300	3,52e+05 (4,80e+08)	30	1,21e-05	sim
ClanAPSO-10x3-G	300	4,03e+05 (7,64e+08)	30	4,20e-07	sim
ClanAPSO-10x3-L	300	3,34e+05 (4,37e+08)	30	7,63e-06	sim
ClanAPSO-3x10-G	300	4,04e+05 (1,06e+09)	30	1,57e-06	sim
ClanAPSO-3x10-L	300	3,82e+05 (7,08e+08)	30	8,11e-07	sim
ClanAPSO-6x5-G	300	3,71e+05 (5,78e+08)	30	1,04e-07	sim
ClanAPSO-6x5-L	300	3,39e+05 (5,96e+08)	30	4,36e-08	sim
ClanPSO-10x3-G	300	4,37e+10 (2,97e+18)	30	3,88e-07	sim
ClanPSO-10x3-L	300	4,34e+10 (3,01e+18)	30	5,04e-08	sim
ClanPSO-3x10-G	300	4,39e+10 (3,31e+18)	30	5,74e-06	sim
ClanPSO-3x10-L	300	4,38e+10 (3,48e+18)	30	3,38e-06	sim
ClanPSO-6x5-G	300	4,43e+10 (3,00e+18)	30	5,75e-08	sim
ClanPSO-6x5-L	300	4,41e+10 (2,84e+18)	30	5,60e-07	sim
Constricted PSO-G	300	4,76e+10 (7,46e+17)	30	4,24e-06	sim
Constricted PSO-L	300	4,76e+10 (7,46e+17)	30	4,24e-06	sim
CPSOS	300	2,18e+05 (1,41e+09)	30	3,05e-08	sim
FSS	300	3,99e+10 (4,35e+18)	30	6,17e-07	sim
GCPSO-G	300	4,82e+10 (3,37e+17)	30	1,01e-07	sim
GCPSO-L	300	4,82e+10 (3,37e+17)	30	1,01e-07	sim
MCPSOCOL	300	4,75e+10 (5,62e+17)	30	1,31e-06	sim
PSO-G	300	4,46e+10 (1,48e+18)	30	1,08e-05	sim
PSO-L	300	4,46e+10 (1,48e+18)	30	1,08e-05	sim
ABC	1000	5,91e+06 (3,27e+11)	30	1,46e-07	sim
APSO-G	1000	7,41e+06 (4,69e+11)	30	3,45e-08	sim
APSO-L	1000	5,22e+06 (1,28e+11)	30	2,36e-08	sim
ClanAPSO-10x3-G	1000	4,71e+06 (1,16e+11)	30	1,33e-06	sim
ClanAPSO-10x3-L	1000	4,17e+06 (6,09e+10)	30	9,75e-07	sim
ClanAPSO-3x10-G	1000	4,83e+06 (1,28e+11)	30	8,99e-06	sim
ClanAPSO-3x10-L	1000	4,83e+06 (1,02e+11)	30	1,80e-09	sim

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ClanAPSO-6x5-G	1000	4,51e+06 (1,80e+11)	30	1,53e-05	sim
ClanAPSO-6x5-L	1000	4,18e+06 (6,00e+10)	30	5,63e-07	sim
ClanPSO-10x3-G	1000	1,79e+12 (2,45e+20)	30	2,84e-06	sim
ClanPSO-10x3-L	1000	1,79e+12 (2,11e+20)	30	3,93e-06	sim
ClanPSO-3x10-G	1000	1,79e+12 (3,61e+20)	30	1,24e-08	sim
ClanPSO-3x10-L	1000	1,79e+12 (2,97e+20)	30	2,33e-07	sim
ClanPSO-6x5-G	1000	1,79e+12 (3,34e+20)	30	1,02e-08	sim
ClanPSO-6x5-L	1000	1,79e+12 (2,81e+20)	30	8,48e-08	sim
Constricted PSO-G	1000	1,82e+12 (1,73e+20)	30	3,51e-06	sim
Constricted PSO-L	1000	1,82e+12 (1,73e+20)	30	3,51e-06	sim
CPSOS	1000	-	0	-	-
FSS	1000	1,82e+12 (1,55e+20)	30	3,25e-06	sim
GCPSO-G	1000	1,83e+12 (1,43e+20)	30	2,19e-06	sim
GCPSO-L	1000	1,83e+12 (1,43e+20)	30	2,19e-06	sim
MCPSOCOL	1000	-	0	-	-
PSO-G	1000	1,80e+12 (2,35e+20)	30	7,18e-08	sim
PSO-L	1000	1,80e+12 (2,35e+20)	30	7,18e-08	sim

Tabela 6.2: Resultados e significância estatística ($\alpha = 0,05$) para Rosenbrock.

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ABC	2	5,12e-03 (1,26e-04)	30	3,61e-02	sim
APSO-G	2	0,00e+00 (0,00e+00)	30	-	-
APSO-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-10x3-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-10x3-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-3x10-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-3x10-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-6x5-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-6x5-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-10x3-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-10x3-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-3x10-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-3x10-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-6x5-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-6x5-L	2	0,00e+00 (0,00e+00)	30	-	-
Constricted PSO-G	2	0,00e+00 (0,00e+00)	30	-	-
Constricted PSO-L	2	0,00e+00 (0,00e+00)	30	-	-
CPSOS	2	2,05e+01 (1,02e+02)	30	3,35e-06	sim
FSS	2	1,56e-04 (3,47e-08)	30	3,02e-04	sim
GCPSO-G	2	0,00e+00 (0,00e+00)	30	-	-
GCPSO-L	2	0,00e+00 (0,00e+00)	30	-	-
MCPSOCOL	2	1,90e-06 (1,09e-10)	30	3,34e-01	não
PSO-G	2	0,00e+00 (0,00e+00)	30	-	-
PSO-L	2	0,00e+00 (0,00e+00)	30	-	-
ABC	10	1,26e-01 (1,71e-02)	30	2,11e-04	sim

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
APSO-G	10	2,31e-01 (4,74e-01)	30	9,43e-02	não
APSO-L	10	2,09e-01 (1,76e-01)	30	9,31e-03	sim
ClanAPSO-10x3-G	10	1,11e-01 (5,02e-02)	30	9,42e-03	sim
ClanAPSO-10x3-L	10	1,47e-02 (1,49e-03)	30	5,94e-02	não
ClanAPSO-3x10-G	10	3,39e-02 (1,11e-02)	30	9,78e-02	não
ClanAPSO-3x10-L	10	6,22e-02 (5,83e-02)	30	1,80e-01	não
ClanAPSO-6x5-G	10	2,71e-02 (9,36e-03)	30	1,49e-01	não
ClanAPSO-6x5-L	10	2,14e-02 (2,04e-03)	30	1,31e-02	sim
ClanPSO-10x3-G	10	1,12e+00 (6,74e-01)	30	8,31e-07	sim
ClanPSO-10x3-L	10	1,15e+00 (9,11e-01)	30	9,20e-09	sim
ClanPSO-3x10-G	10	8,40e-01 (2,70e-01)	30	1,58e-04	sim
ClanPSO-3x10-L	10	1,03e+00 (4,64e-01)	30	3,36e-08	sim
ClanPSO-6x5-G	10	9,92e-01 (4,98e-01)	30	5,42e-07	sim
ClanPSO-6x5-L	10	7,01e-01 (2,25e-01)	30	6,26e-09	sim
Constricted PSO-G	10	6,98e-02 (9,87e-02)	30	2,53e-01	não
Constricted PSO-L	10	1,06e+00 (3,22e+00)	30	1,29e-03	sim
CPSOS	10	3,13e+02 (5,21e+05)	30	2,81e-02	sim
FSS	10	8,06e+00 (1,86e+02)	30	1,30e-01	não
GCPSO-G	10	3,25e-02 (5,14e-03)	30	4,55e-02	sim
GCPSO-L	10	1,06e+00 (3,22e+00)	30	1,32e-03	sim
MCPSOCOL	10	4,17e+01 (3,23e+03)	30	1,01e-03	sim
PSO-G	10	1,55e+00 (5,18e-01)	30	2,08e-04	sim
PSO-L	10	1,74e+00 (1,77e+00)	30	5,19e-07	sim
ABC	30	4,36e-01 (3,82e-01)	30	2,34e-03	sim
APSO-G	30	1,52e+01 (4,24e+02)	30	2,33e-03	sim
APSO-L	30	1,16e+01 (4,86e+02)	30	8,45e-03	sim
ClanAPSO-10x3-G	30	2,53e+01 (1,03e+03)	30	2,01e-04	sim
ClanAPSO-10x3-L	30	1,69e+01 (8,27e+02)	30	2,69e-03	sim
ClanAPSO-3x10-G	30	1,80e+01 (6,55e+02)	30	1,40e-03	sim
ClanAPSO-3x10-L	30	1,35e+01 (4,73e+02)	30	3,45e-03	sim
ClanAPSO-6x5-G	30	1,45e+01 (6,07e+02)	30	3,36e-03	sim
ClanAPSO-6x5-L	30	8,71e+00 (2,55e+02)	30	1,96e-02	sim
ClanPSO-10x3-G	30	5,01e+01 (1,01e+03)	30	1,58e-06	sim
ClanPSO-10x3-L	30	5,26e+01 (7,18e+02)	30	1,11e-10	sim
ClanPSO-3x10-G	30	5,10e+01 (1,73e+03)	30	1,73e-06	sim
ClanPSO-3x10-L	30	5,19e+01 (2,25e+03)	30	3,19e-05	sim
ClanPSO-6x5-G	30	4,77e+01 (1,26e+03)	30	1,64e-05	sim
ClanPSO-6x5-L	30	4,15e+01 (7,96e+02)	30	3,18e-06	sim
Constricted PSO-G	30	2,87e+01 (6,27e+02)	30	2,83e-04	sim
Constricted PSO-L	30	4,42e+00 (1,70e+02)	30	7,26e-02	não
CPSOS	30	6,43e+03 (1,18e+09)	30	3,27e-01	não
FSS	30	5,60e+01 (3,46e+03)	30	8,37e-03	sim
GCPSO-G	30	2,42e+01 (5,69e+02)	30	3,49e-04	sim
GCPSO-L	30	6,14e+00 (2,02e+02)	30	2,40e-02	sim
MCPSOCOL	30	1,03e+07 (6,94e+13)	30	2,84e-05	sim
PSO-G	30	5,84e+01 (9,96e+02)	30	4,08e-10	sim
PSO-L	30	3,27e+01 (8,44e+02)	30	6,01e-05	sim

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ABC	100	1,27e+01 (3,73e+01)	30	1,07e-08	sim
APSO-G	100	2,28e+02 (8,63e+03)	30	2,11e-07	sim
APSO-L	100	1,42e+02 (6,19e+03)	30	1,43e-06	sim
ClanAPSO-10x3-G	100	2,13e+02 (3,84e+03)	30	1,12e-07	sim
ClanAPSO-10x3-L	100	1,72e+02 (3,98e+03)	30	1,62e-06	sim
ClanAPSO-3x10-G	100	1,97e+02 (5,74e+03)	30	2,16e-04	sim
ClanAPSO-3x10-L	100	2,05e+02 (4,08e+03)	30	1,69e-06	sim
ClanAPSO-6x5-G	100	1,93e+02 (5,83e+03)	30	6,40e-07	sim
ClanAPSO-6x5-L	100	1,73e+02 (3,57e+03)	30	1,28e-05	sim
ClanPSO-10x3-G	100	4,29e+04 (4,91e+08)	30	1,60e-07	sim
ClanPSO-10x3-L	100	2,73e+04 (6,13e+08)	30	9,81e-03	sim
ClanPSO-3x10-G	100	3,64e+04 (2,93e+09)	30	2,21e-03	sim
ClanPSO-3x10-L	100	2,65e+04 (1,80e+09)	30	1,01e-02	sim
ClanPSO-6x5-G	100	1,94e+04 (2,99e+08)	30	5,92e-05	sim
ClanPSO-6x5-L	100	1,29e+04 (1,23e+08)	30	4,36e-05	sim
Constricted PSO-G	100	2,72e+02 (9,67e+03)	30	7,40e-05	sim
Constricted PSO-L	100	2,57e+08 (2,55e+16)	30	2,00e-08	sim
CPSOS	100	5,40e+03 (3,65e+08)	30	1,48e-01	não
FSS	100	4,23e+02 (1,96e+05)	30	3,24e-03	sim
GCPSO-G	100	2,82e+02 (1,01e+04)	30	3,54e-06	sim
GCPSO-L	100	1,89e+08 (1,46e+16)	30	2,24e-08	sim
MCPSOCOL	100	6,44e+08 (3,76e+16)	30	2,03e-06	sim
PSO-G	100	1,91e+05 (9,55e+10)	30	4,43e-03	sim
PSO-L	100	4,21e+02 (4,41e+04)	30	8,97e-06	sim
ABC	300	8,38e+01 (2,13e+03)	30	3,35e-05	sim
APSO-G	300	2,26e+04 (3,89e+07)	30	4,31e-06	sim
APSO-L	300	1,51e+03 (2,83e+05)	30	4,52e-03	sim
ClanAPSO-10x3-G	300	2,37e+03 (1,33e+05)	30	4,39e-05	sim
ClanAPSO-10x3-L	300	1,64e+03 (2,66e+05)	30	2,16e-03	sim
ClanAPSO-3x10-G	300	2,08e+03 (3,23e+05)	30	1,37e-03	sim
ClanAPSO-3x10-L	300	1,80e+03 (1,65e+05)	30	1,88e-03	sim
ClanAPSO-6x5-G	300	1,95e+03 (3,32e+05)	30	6,19e-04	sim
ClanAPSO-6x5-L	300	1,52e+03 (1,94e+05)	30	2,04e-03	sim
ClanPSO-10x3-G	300	7,32e+09 (6,10e+16)	30	2,74e-06	sim
ClanPSO-10x3-L	300	7,29e+09 (5,49e+16)	30	8,45e-07	sim
ClanPSO-3x10-G	300	7,45e+09 (4,73e+16)	30	9,11e-07	sim
ClanPSO-3x10-L	300	7,45e+09 (4,64e+16)	30	9,52e-07	sim
ClanPSO-6x5-G	300	7,37e+09 (5,57e+16)	30	6,46e-06	sim
ClanPSO-6x5-L	300	7,37e+09 (6,17e+16)	30	2,48e-06	sim
Constricted PSO-G	300	7,85e+09 (3,10e+16)	30	9,24e-06	sim
Constricted PSO-L	300	7,85e+09 (3,10e+16)	30	9,24e-06	sim
CPSOS	300	7,59e+04 (1,36e+11)	30	2,80e-01	não
FSS	300	5,24e+09 (6,23e+17)	30	5,67e-07	sim
GCPSO-G	300	7,92e+09 (2,57e+16)	30	9,09e-07	sim
GCPSO-L	300	7,92e+09 (2,57e+16)	30	9,09e-07	sim
MCPSOCOL	300	7,88e+09 (2,92e+16)	30	1,40e-07	sim
PSO-G	300	7,45e+09 (4,42e+16)	30	3,38e-08	sim

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
PSO-L	300	7,43e+09 (5,42e+16)	30	1,67e-07	sim
ABC	1000	6,90e+05 (3,03e+11)	30	5,80e-05	sim
APSO-G	1000	4,11e+08 (1,47e+16)	30	9,52e-07	sim
APSO-L	1000	2,27e+08 (1,97e+16)	30	1,10e-05	sim
ClanAPSO-10x3-G	1000	8,48e+05 (1,06e+10)	30	1,57e-06	sim
ClanAPSO-10x3-L	1000	3,34e+05 (2,45e+09)	30	4,15e-07	sim
ClanAPSO-3x10-G	1000	9,81e+05 (1,53e+10)	30	1,20e-05	sim
ClanAPSO-3x10-L	1000	7,24e+05 (1,76e+10)	30	9,38e-05	sim
ClanAPSO-6x5-G	1000	5,94e+05 (4,29e+09)	30	8,17e-08	sim
ClanAPSO-6x5-L	1000	2,62e+05 (7,01e+08)	30	3,50e-08	sim
ClanPSO-10x3-G	1000	2,73e+10 (5,66e+16)	30	1,72e-07	sim
ClanPSO-10x3-L	1000	2,72e+10 (5,66e+16)	30	2,34e-07	sim
ClanPSO-3x10-G	1000	2,73e+10 (6,70e+16)	30	7,85e-07	sim
ClanPSO-3x10-L	1000	2,73e+10 (6,40e+16)	30	3,31e-07	sim
ClanPSO-6x5-G	1000	2,73e+10 (6,67e+16)	30	4,14e-07	sim
ClanPSO-6x5-L	1000	2,73e+10 (6,57e+16)	30	2,11e-07	sim
Constricted PSO-G	1000	2,76e+10 (5,52e+16)	30	4,95e-07	sim
Constricted PSO-L	1000	2,76e+10 (5,52e+16)	30	4,95e-07	sim
CPSOS	1000	-	0	-	-
FSS	1000	2,78e+10 (1,03e+17)	30	8,42e-08	sim
GCPSO-G	1000	2,77e+10 (5,27e+16)	30	2,09e-06	sim
GCPSO-L	1000	2,77e+10 (5,27e+16)	30	2,09e-06	sim
MCPSOCOL	1000	-	0	-	-
PSO-G	1000	2,73e+10 (5,79e+16)	30	9,89e-06	sim
PSO-L	1000	2,73e+10 (5,79e+16)	30	9,89e-06	sim

Tabela 6.3: Resultados e significância estatística ($\alpha = 0,05$) para Schwefel-2.26.

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ABC	2	-8,38e+02 (0,00e+00)	30	-	-
APSO-G	2	-8,38e+02 (0,00e+00)	30	-	-
APSO-L	2	-8,38e+02 (0,00e+00)	30	-	-
ClanAPSO-10x3-G	2	-8,38e+02 (0,00e+00)	30	-	-
ClanAPSO-10x3-L	2	-8,38e+02 (0,00e+00)	30	-	-
ClanAPSO-3x10-G	2	-8,38e+02 (0,00e+00)	30	-	-
ClanAPSO-3x10-L	2	-8,38e+02 (0,00e+00)	30	-	-
ClanAPSO-6x5-G	2	-8,38e+02 (0,00e+00)	30	-	-
ClanAPSO-6x5-L	2	-8,38e+02 (0,00e+00)	30	-	-
ClanPSO-10x3-G	2	-6,25e+02 (2,32e+03)	30	8,56e-03	sim
ClanPSO-10x3-L	2	-6,25e+02 (3,29e+03)	30	2,81e-02	sim
ClanPSO-3x10-G	2	-6,25e+02 (2,32e+03)	30	8,56e-03	sim
ClanPSO-3x10-L	2	-6,25e+02 (2,32e+03)	30	8,56e-03	sim
ClanPSO-6x5-G	2	-6,29e+02 (2,60e+03)	30	3,54e-03	sim
ClanPSO-6x5-L	2	-6,37e+02 (3,05e+03)	30	4,26e-04	sim
Constricted PSO-G	2	-8,38e+02 (0,00e+00)	30	-	-
Constricted PSO-L	2	-7,83e+02 (4,58e+03)	30	1,59e-06	sim

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
CPSOS	2	-6,02e+02 (1,80e+03)	30	4,80e-02	sim
FSS	2	-8,06e+02 (3,82e+03)	30	5,65e-03	sim
GCPSO-G	2	-8,38e+02 (0,00e+00)	30	-	-
GCPSO-L	2	-8,14e+02 (2,32e+03)	30	8,56e-03	sim
MCPSOCOL	2	-6,68e+02 (4,13e+03)	30	1,05e-11	sim
PSO-G	2	-6,09e+02 (9,03e+02)	30	1,64e-01	não
PSO-L	2	-6,01e+02 (0,00e+00)	30	-	-
ABC	10	-4,19e+03 (0,00e+00)	30	-	-
APSO-G	10	-4,19e+03 (0,00e+00)	30	-	-
APSO-L	10	-4,19e+03 (0,00e+00)	30	-	-
ClanAPSO-10x3-G	10	-4,19e+03 (0,00e+00)	30	-	-
ClanAPSO-10x3-L	10	-4,19e+03 (0,00e+00)	30	-	-
ClanAPSO-3x10-G	10	-4,19e+03 (0,00e+00)	30	-	-
ClanAPSO-3x10-L	10	-4,19e+03 (0,00e+00)	30	-	-
ClanAPSO-6x5-G	10	-4,19e+03 (0,00e+00)	30	-	-
ClanAPSO-6x5-L	10	-4,19e+03 (0,00e+00)	30	-	-
ClanPSO-10x3-G	10	-3,10e+03 (4,80e+03)	30	2,16e-04	sim
ClanPSO-10x3-L	10	-3,11e+03 (8,45e+03)	30	3,35e-03	sim
ClanPSO-3x10-G	10	-3,03e+03 (2,60e+03)	30	3,54e-03	sim
ClanPSO-3x10-L	10	-3,05e+03 (4,45e+03)	30	7,46e-05	sim
ClanPSO-6x5-G	10	-3,08e+03 (6,38e+03)	30	2,19e-08	sim
ClanPSO-6x5-L	10	-3,08e+03 (5,50e+03)	30	5,43e-10	sim
Constricted PSO-G	10	-3,74e+03 (2,00e+04)	30	1,05e-06	sim
Constricted PSO-L	10	-3,16e+03 (2,79e+04)	30	1,25e-04	sim
CPSOS	10	-3,11e+03 (3,29e+04)	30	1,43e-06	sim
FSS	10	-3,67e+03 (3,27e+04)	30	4,50e-10	sim
GCPSO-G	10	-3,71e+03 (1,98e+04)	30	5,57e-08	sim
GCPSO-L	10	-3,12e+03 (1,34e+04)	30	4,59e-06	sim
MCPSOCOL	10	-3,07e+03 (1,46e+04)	30	6,23e-08	sim
PSO-G	10	-3,01e+03 (0,00e+00)	30	-	-
PSO-L	10	-3,03e+03 (3,56e+03)	30	1,35e-02	sim
ABC	30	-1,26e+04 (3,20e-15)	30	3,20e-01	não
APSO-G	30	-1,26e+04 (1,68e+03)	30	4,06e-02	sim
APSO-L	30	-1,26e+04 (4,68e+02)	30	3,34e-01	não
ClanAPSO-10x3-G	30	-1,26e+04 (0,00e+00)	30	-	-
ClanAPSO-10x3-L	30	-1,26e+04 (0,00e+00)	30	-	-
ClanAPSO-3x10-G	30	-1,26e+04 (0,00e+00)	30	-	-
ClanAPSO-3x10-L	30	-1,26e+04 (0,00e+00)	30	-	-
ClanAPSO-6x5-G	30	-1,26e+04 (0,00e+00)	30	-	-
ClanAPSO-6x5-L	30	-1,26e+04 (0,00e+00)	30	-	-
ClanPSO-10x3-G	30	-8,92e+03 (5,56e+04)	30	2,23e-07	sim
ClanPSO-10x3-L	30	-8,88e+03 (4,29e+04)	30	1,77e-06	sim
ClanPSO-3x10-G	30	-9,00e+03 (2,21e+04)	30	1,41e-04	sim
ClanPSO-3x10-L	30	-8,99e+03 (2,61e+04)	30	1,50e-06	sim
ClanPSO-6x5-G	30	-8,93e+03 (5,00e+04)	30	2,82e-05	sim
ClanPSO-6x5-L	30	-8,86e+03 (2,70e+04)	30	7,47e-09	sim
Constricted PSO-G	30	-9,31e+03 (2,04e+05)	30	6,19e-04	sim

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
Constricted PSO-L	30	-8,77e+03 (7,91e+04)	30	2,81e-08	sim
CPSOS	30	-9,38e+03 (1,02e+05)	30	3,01e-08	sim
FSS	30	-9,44e+03 (3,14e+05)	30	2,40e-09	sim
GCPSO-G	30	-9,23e+03 (1,25e+05)	30	7,69e-07	sim
GCPSO-L	30	-8,67e+03 (9,52e+04)	30	1,47e-08	sim
MCPSOCOL	30	-8,02e+03 (7,72e+04)	30	8,76e-08	sim
PSO-G	30	-9,00e+03 (4,08e+03)	30	8,25e-03	sim
PSO-L	30	-8,86e+03 (6,18e+04)	30	4,03e-06	sim
ABC	100	-4,12e+04 (4,93e+04)	30	3,85e-09	sim
APSO-G	100	-4,07e+04 (1,15e+05)	30	1,71e-06	sim
APSO-L	100	-4,05e+04 (9,55e+04)	30	3,26e-06	sim
ClanAPSO-10x3-G	100	-4,19e+04 (1,74e-04)	30	6,22e-03	sim
ClanAPSO-10x3-L	100	-4,19e+04 (3,03e-11)	30	2,69e-05	sim
ClanAPSO-3x10-G	100	-4,18e+04 (6,53e+03)	30	3,76e-05	sim
ClanAPSO-3x10-L	100	-4,18e+04 (8,21e+03)	30	9,36e-07	sim
ClanAPSO-6x5-G	100	-4,19e+04 (2,05e-06)	30	2,07e-01	não
ClanAPSO-6x5-L	100	-4,19e+04 (9,03e+02)	30	1,64e-01	não
ClanPSO-10x3-G	100	-2,53e+04 (8,07e+05)	30	2,93e-08	sim
ClanPSO-10x3-L	100	-2,52e+04 (8,70e+05)	30	2,87e-06	sim
ClanPSO-3x10-G	100	-2,58e+04 (1,40e+06)	30	4,20e-09	sim
ClanPSO-3x10-L	100	-2,56e+04 (1,48e+06)	30	1,36e-07	sim
ClanPSO-6x5-G	100	-2,55e+04 (1,35e+06)	30	4,92e-06	sim
ClanPSO-6x5-L	100	-2,60e+04 (7,18e+05)	30	3,11e-07	sim
Constricted PSO-G	100	-1,86e+04 (9,31e+07)	30	7,65e-07	sim
Constricted PSO-L	100	-2,56e+04 (5,98e+06)	30	1,06e-04	sim
CPSOS	100	-3,21e+04 (7,28e+05)	30	6,19e-07	sim
FSS	100	-5,01e+03 (1,08e+06)	30	1,03e-07	sim
GCPSO-G	100	-1,54e+04 (1,12e+08)	30	5,18e-15	sim
GCPSO-L	100	-2,57e+04 (4,77e+06)	30	2,61e-06	sim
MCPSOCOL	100	-1,56e+04 (5,93e+06)	30	2,01e-08	sim
PSO-G	100	-2,49e+04 (9,77e+06)	30	2,00e-04	sim
PSO-L	100	-2,66e+04 (1,04e+06)	30	2,73e-04	sim
ABC	300	-9,04e+04 (1,21e+04)	30	3,18e-08	sim
APSO-G	300	-1,11e+05 (7,40e+05)	30	1,25e-07	sim
APSO-L	300	-1,11e+05 (1,49e+06)	30	2,67e-05	sim
ClanAPSO-10x3-G	300	-1,20e+05 (5,79e+05)	30	9,07e-09	sim
ClanAPSO-10x3-L	300	-1,22e+05 (4,86e+05)	30	3,71e-07	sim
ClanAPSO-3x10-G	300	-1,19e+05 (7,24e+05)	30	6,76e-08	sim
ClanAPSO-3x10-L	300	-1,19e+05 (7,74e+05)	30	4,11e-08	sim
ClanAPSO-6x5-G	300	-1,21e+05 (6,00e+05)	30	6,77e-07	sim
ClanAPSO-6x5-L	300	-1,21e+05 (2,47e+05)	30	7,36e-08	sim
ClanPSO-10x3-G	300	-6,36e+00 (7,65e+06)	30	2,36e-08	sim
ClanPSO-10x3-L	300	-4,51e+02 (6,22e+06)	30	4,32e-08	sim
ClanPSO-3x10-G	300	6,97e+00 (7,46e+06)	30	6,67e-09	sim
ClanPSO-3x10-L	300	-1,55e+02 (6,86e+06)	30	7,50e-09	sim
ClanPSO-6x5-G	300	-2,80e+01 (5,46e+06)	30	1,90e-08	sim
ClanPSO-6x5-L	300	1,87e+01 (6,50e+06)	30	8,50e-08	sim

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
Constricted PSO-G	300	4,47e+03 (5,62e+06)	30	1,42e-09	sim
Constricted PSO-L	300	4,47e+03 (5,62e+06)	30	1,42e-09	sim
CPSOS	300	-9,66e+04 (1,23e+06)	30	1,67e-08	sim
FSS	300	2,30e+03 (3,85e+06)	30	7,52e-05	sim
GCPSO-G	300	5,01e+03 (5,05e+06)	30	3,49e-08	sim
GCPSO-L	300	5,01e+03 (5,05e+06)	30	3,49e-08	sim
MCPSOCOL	300	4,35e+03 (3,85e+06)	30	5,88e-09	sim
PSO-G	300	1,33e+03 (6,33e+06)	30	1,33e-07	sim
PSO-L	300	1,33e+03 (6,33e+06)	30	1,33e-07	sim
ABC	1000	-2,96e+05 (6,88e+05)	30	5,41e-08	sim
APSO-G	1000	-2,81e+05 (9,20e+06)	30	4,39e-08	sim
APSO-L	1000	-3,23e+05 (3,15e+06)	30	2,82e-08	sim
ClanAPSO-10x3-G	1000	-3,44e+05 (3,26e+06)	30	6,46e-09	sim
ClanAPSO-10x3-L	1000	-3,54e+05 (3,65e+06)	30	9,92e-06	sim
ClanAPSO-3x10-G	1000	-3,43e+05 (5,56e+06)	30	2,97e-07	sim
ClanAPSO-3x10-L	1000	-3,44e+05 (7,15e+06)	30	5,34e-07	sim
ClanAPSO-6x5-G	1000	-3,49e+05 (3,43e+06)	30	9,84e-09	sim
ClanAPSO-6x5-L	1000	-3,53e+05 (2,86e+06)	30	5,03e-08	sim
ClanPSO-10x3-G	1000	2,63e+04 (1,69e+07)	30	2,30e-05	sim
ClanPSO-10x3-L	1000	2,60e+04 (1,65e+07)	30	4,45e-05	sim
ClanPSO-3x10-G	1000	2,62e+04 (1,41e+07)	30	6,41e-06	sim
ClanPSO-3x10-L	1000	2,61e+04 (1,30e+07)	30	8,49e-06	sim
ClanPSO-6x5-G	1000	2,59e+04 (1,69e+07)	30	3,57e-04	sim
ClanPSO-6x5-L	1000	2,59e+04 (1,57e+07)	30	2,78e-04	sim
Constricted PSO-G	1000	2,98e+04 (1,45e+07)	30	1,36e-05	sim
Constricted PSO-L	1000	2,98e+04 (1,45e+07)	30	1,36e-05	sim
CPSOS	1000	-	0	-	-
FSS	1000	3,04e+04 (2,32e+07)	30	2,04e-07	sim
GCPSO-G	1000	3,06e+04 (1,52e+07)	30	1,15e-05	sim
GCPSO-L	1000	3,06e+04 (1,52e+07)	30	1,15e-05	sim
MCPSOCOL	1000	-	0	-	-
PSO-G	1000	2,72e+04 (1,65e+07)	30	6,49e-07	sim
PSO-L	1000	2,72e+04 (1,65e+07)	30	6,49e-07	sim

Tabela 6.4: Resultados e significância estatística ($\alpha = 0,05$) para Rastrigin.

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ABC	2	0,00e+00 (0,00e+00)	30	-	-
APSO-G	2	0,00e+00 (0,00e+00)	30	-	-
APSO-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-10x3-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-10x3-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-3x10-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-3x10-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-6x5-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-6x5-L	2	0,00e+00 (0,00e+00)	30	-	-

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ClanPSO-10x3-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-10x3-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-3x10-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-3x10-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-6x5-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-6x5-L	2	0,00e+00 (0,00e+00)	30	-	-
Constricted PSO-G	2	0,00e+00 (0,00e+00)	30	-	-
Constricted PSO-L	2	0,00e+00 (0,00e+00)	30	-	-
CPSOS	2	2,53e+00 (1,77e+01)	30	1,11e-03	sim
FSS	2	2,17e-03 (4,76e-06)	30	6,47e-05	sim
GCPSO-G	2	0,00e+00 (0,00e+00)	30	-	-
GCPSO-L	2	0,00e+00 (0,00e+00)	30	-	-
MCPSOCOL	2	6,63e-02 (6,37e-02)	30	1,64e-01	não
PSO-G	2	0,00e+00 (0,00e+00)	30	-	-
PSO-L	2	0,00e+00 (0,00e+00)	30	-	-
ABC	10	0,00e+00 (0,00e+00)	30	-	-
APSO-G	10	0,00e+00 (0,00e+00)	30	-	-
APSO-L	10	1,24e-15 (1,72e-29)	30	1,13e-01	não
ClanAPSO-10x3-G	10	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-10x3-L	10	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-3x10-G	10	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-3x10-L	10	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-6x5-G	10	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-6x5-L	10	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-10x3-G	10	1,33e-01 (1,18e-01)	30	4,06e-02	sim
ClanPSO-10x3-L	10	7,42e-01 (9,55e-01)	30	1,57e-05	sim
ClanPSO-3x10-G	10	2,47e-01 (3,21e-01)	30	2,01e-02	sim
ClanPSO-3x10-L	10	5,85e-01 (6,95e-01)	30	9,36e-05	sim
ClanPSO-6x5-G	10	2,08e-01 (2,69e-01)	30	3,29e-02	sim
ClanPSO-6x5-L	10	5,65e-01 (9,38e-01)	30	1,60e-03	sim
Constricted PSO-G	10	5,09e-01 (8,11e-01)	30	2,32e-03	sim
Constricted PSO-L	10	8,09e+00 (2,39e+01)	30	3,98e-06	sim
CPSOS	10	1,22e+01 (2,07e+02)	30	6,49e-04	sim
FSS	10	1,33e+01 (1,87e+01)	30	5,64e-09	sim
GCPSO-G	10	6,49e-01 (6,79e-01)	30	5,40e-06	sim
GCPSO-L	10	1,17e+01 (4,44e+01)	30	4,28e-06	sim
MCPSOCOL	10	3,19e+01 (1,24e+02)	30	2,87e-07	sim
PSO-G	10	2,00e+00 (6,09e+00)	30	3,79e-05	sim
PSO-L	10	7,96e-01 (6,42e-01)	30	2,31e-06	sim
ABC	30	8,50e-14 (1,09e-26)	30	7,46e-05	sim
APSO-G	30	3,32e-02 (3,30e-02)	30	3,34e-01	não
APSO-L	30	6,63e-02 (6,37e-02)	30	1,64e-01	não
ClanAPSO-10x3-G	30	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-10x3-L	30	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-3x10-G	30	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-3x10-L	30	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-6x5-G	30	0,00e+00 (0,00e+00)	30	-	-

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ClanAPSO-6x5-L	30	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-10x3-G	30	2,69e+01 (3,20e+01)	30	1,58e-07	sim
ClanPSO-10x3-L	30	3,04e+01 (3,91e+01)	30	6,78e-06	sim
ClanPSO-3x10-G	30	3,93e+01 (6,12e+02)	30	1,53e-03	sim
ClanPSO-3x10-L	30	4,03e+01 (5,13e+02)	30	2,68e-04	sim
ClanPSO-6x5-G	30	2,82e+01 (6,81e+01)	30	2,36e-04	sim
ClanPSO-6x5-L	30	2,90e+01 (5,25e+01)	30	1,02e-08	sim
Constricted PSO-G	30	8,40e+01 (1,19e+03)	30	1,76e-08	sim
Constricted PSO-L	30	1,79e+02 (9,55e+02)	30	1,52e-07	sim
CPSOS	30	3,49e+01 (1,77e+02)	30	2,95e-07	sim
FSS	30	7,54e+01 (4,47e+02)	30	1,25e-05	sim
GCPSO-G	30	9,76e+01 (1,43e+03)	30	2,83e-09	sim
GCPSO-L	30	1,93e+02 (2,08e+03)	30	1,35e-06	sim
MCPSOCOL	30	2,65e+02 (1,05e+03)	30	1,45e-05	sim
PSO-G	30	8,18e+01 (1,28e+03)	30	1,40e-09	sim
PSO-L	30	2,69e+01 (6,20e+01)	30	2,73e-07	sim
ABC	100	1,07e+00 (1,09e+00)	30	1,64e-04	sim
APSO-G	100	6,52e+00 (7,52e+00)	30	2,80e-08	sim
APSO-L	100	8,29e+00 (7,48e+00)	30	8,01e-07	sim
ClanAPSO-10x3-G	100	9,09e-03 (5,03e-05)	30	2,01e-05	sim
ClanAPSO-10x3-L	100	6,78e-05 (4,42e-09)	30	3,79e-06	sim
ClanAPSO-3x10-G	100	2,99e-01 (2,15e-01)	30	4,15e-04	sim
ClanAPSO-3x10-L	100	4,32e-01 (2,51e-01)	30	1,48e-07	sim
ClanAPSO-6x5-G	100	3,44e-02 (3,32e-02)	30	3,21e-01	não
ClanAPSO-6x5-L	100	3,32e-02 (3,30e-02)	30	3,34e-01	não
ClanPSO-10x3-G	100	3,93e+02 (2,21e+03)	30	5,37e-08	sim
ClanPSO-10x3-L	100	4,12e+02 (2,19e+03)	30	2,31e-07	sim
ClanPSO-3x10-G	100	4,44e+02 (1,75e+04)	30	8,95e-09	sim
ClanPSO-3x10-L	100	4,19e+02 (1,36e+04)	30	9,02e-08	sim
ClanPSO-6x5-G	100	3,97e+02 (8,44e+03)	30	5,72e-09	sim
ClanPSO-6x5-L	100	4,07e+02 (6,30e+03)	30	3,31e-07	sim
Constricted PSO-G	100	8,97e+02 (3,60e+04)	30	2,00e-08	sim
Constricted PSO-L	100	1,06e+03 (7,03e+03)	30	2,60e-06	sim
CPSOS	100	1,19e+02 (7,66e+02)	30	1,17e-08	sim
FSS	100	1,40e+03 (6,45e+04)	30	2,08e-11	sim
GCPSO-G	100	9,49e+02 (5,17e+04)	30	8,70e-07	sim
GCPSO-L	100	1,07e+03 (5,17e+03)	30	3,91e-07	sim
MCPSOCOL	100	1,44e+03 (4,50e+03)	30	1,11e-08	sim
PSO-G	100	6,46e+02 (2,64e+04)	30	5,57e-08	sim
PSO-L	100	5,15e+02 (1,24e+04)	30	7,08e-08	sim
ABC	300	9,47e+01 (3,36e+02)	30	2,05e-07	sim
APSO-G	300	2,31e+02 (2,95e+02)	30	5,92e-07	sim
APSO-L	300	9,89e+01 (1,68e+02)	30	1,89e-07	sim
ClanAPSO-10x3-G	300	8,42e+01 (9,21e+01)	30	1,46e-06	sim
ClanAPSO-10x3-L	300	4,97e+01 (2,92e+01)	30	2,77e-07	sim
ClanAPSO-3x10-G	300	7,07e+01 (6,62e+01)	30	2,62e-10	sim
ClanAPSO-3x10-L	300	6,96e+01 (3,81e+01)	30	6,07e-10	sim

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ClanAPSO-6x5-G	300	6,67e+01 (6,03e+01)	30	6,28e-10	sim
ClanAPSO-6x5-L	300	4,68e+01 (3,87e+01)	30	1,15e-06	sim
ClanPSO-10x3-G	300	6,70e+03 (1,54e+04)	30	2,29e-07	sim
ClanPSO-10x3-L	300	6,66e+03 (1,17e+04)	30	2,46e-07	sim
ClanPSO-3x10-G	300	6,73e+03 (1,46e+04)	30	2,86e-07	sim
ClanPSO-3x10-L	300	6,71e+03 (2,53e+04)	30	3,18e-05	sim
ClanPSO-6x5-G	300	6,69e+03 (9,86e+03)	30	1,46e-08	sim
ClanPSO-6x5-L	300	6,68e+03 (8,90e+03)	30	7,42e-08	sim
Constricted PSO-G	300	7,02e+03 (8,16e+03)	30	3,16e-06	sim
Constricted PSO-L	300	7,02e+03 (8,16e+03)	30	3,16e-06	sim
CPSOS	300	3,82e+02 (1,77e+03)	30	2,32e-07	sim
FSS	300	6,55e+03 (1,18e+04)	30	4,89e-07	sim
GCPSO-G	300	7,06e+03 (6,13e+03)	30	1,96e-06	sim
GCPSO-L	300	7,06e+03 (6,13e+03)	30	1,96e-06	sim
MCPSOCOL	300	7,04e+03 (7,56e+03)	30	5,10e-06	sim
PSO-G	300	6,77e+03 (9,74e+03)	30	1,09e-07	sim
PSO-L	300	6,77e+03 (9,74e+03)	30	1,09e-07	sim
ABC	1000	4,36e+03 (1,76e+04)	30	3,39e-07	sim
APSO-G	1000	4,63e+03 (2,11e+04)	30	3,14e-08	sim
APSO-L	1000	2,61e+03 (1,44e+04)	30	5,70e-06	sim
ClanAPSO-10x3-G	1000	1,77e+03 (3,87e+03)	30	7,36e-07	sim
ClanAPSO-10x3-L	1000	1,43e+03 (2,77e+03)	30	2,16e-06	sim
ClanAPSO-3x10-G	1000	1,73e+03 (3,89e+03)	30	7,70e-06	sim
ClanAPSO-3x10-L	1000	1,66e+03 (4,10e+03)	30	5,75e-06	sim
ClanAPSO-6x5-G	1000	1,59e+03 (4,42e+03)	30	7,30e-08	sim
ClanAPSO-6x5-L	1000	1,37e+03 (2,58e+03)	30	1,70e-07	sim
ClanPSO-10x3-G	1000	2,38e+04 (2,53e+04)	30	6,55e-06	sim
ClanPSO-10x3-L	1000	2,38e+04 (2,61e+04)	30	4,29e-06	sim
ClanPSO-3x10-G	1000	2,38e+04 (3,45e+04)	30	7,91e-07	sim
ClanPSO-3x10-L	1000	2,38e+04 (3,38e+04)	30	8,73e-07	sim
ClanPSO-6x5-G	1000	2,38e+04 (3,83e+04)	30	2,10e-07	sim
ClanPSO-6x5-L	1000	2,38e+04 (3,73e+04)	30	1,81e-07	sim
Constricted PSO-G	1000	2,41e+04 (2,32e+04)	30	6,58e-07	sim
Constricted PSO-L	1000	2,41e+04 (2,32e+04)	30	6,58e-07	sim
CPSOS	1000	-	0	-	-
FSS	1000	2,41e+04 (2,53e+04)	30	5,70e-08	sim
GCPSO-G	1000	2,41e+04 (2,34e+04)	30	5,48e-07	sim
GCPSO-L	1000	2,41e+04 (2,34e+04)	30	5,48e-07	sim
MCPSOCOL	1000	-	0	-	-
PSO-G	1000	2,39e+04 (3,10e+04)	30	4,16e-08	sim
PSO-L	1000	2,39e+04 (3,10e+04)	30	4,16e-08	sim

Tabela 6.5: Resultados e significância estatística ($\alpha = 0,05$) para Ackley.

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ABC	2	4,44e-16 (0,00e+00)	30	-	-

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
APSO-G	2	4,44e-16 (0,00e+00)	30	-	-
APSO-L	2	4,44e-16 (0,00e+00)	30	-	-
ClanAPSO-10x3-G	2	4,44e-16 (0,00e+00)	30	-	-
ClanAPSO-10x3-L	2	5,63e-16 (4,21e-31)	30	3,34e-01	não
ClanAPSO-3x10-G	2	4,44e-16 (0,00e+00)	30	-	-
ClanAPSO-3x10-L	2	4,44e-16 (0,00e+00)	30	-	-
ClanAPSO-6x5-G	2	4,44e-16 (0,00e+00)	30	-	-
ClanAPSO-6x5-L	2	4,44e-16 (0,00e+00)	30	-	-
ClanPSO-10x3-G	2	4,44e-16 (0,00e+00)	30	-	-
ClanPSO-10x3-L	2	4,44e-16 (0,00e+00)	30	-	-
ClanPSO-3x10-G	2	4,44e-16 (0,00e+00)	30	-	-
ClanPSO-3x10-L	2	4,44e-16 (0,00e+00)	30	-	-
ClanPSO-6x5-G	2	4,44e-16 (0,00e+00)	30	-	-
ClanPSO-6x5-L	2	4,44e-16 (0,00e+00)	30	-	-
Constricted PSO-G	2	4,44e-16 (0,00e+00)	30	-	-
Constricted PSO-L	2	4,44e-16 (0,00e+00)	30	-	-
CPSOS	2	1,87e+00 (3,27e+01)	30	8,29e-02	não
FSS	2	3,36e-04 (3,65e-08)	30	1,03e-09	sim
GCPSO-G	2	4,44e-16 (0,00e+00)	30	-	-
GCPSO-L	2	4,44e-16 (0,00e+00)	30	-	-
MCPSOCOL	2	4,44e-16 (0,00e+00)	30	-	-
PSO-G	2	4,44e-16 (0,00e+00)	30	-	-
PSO-L	2	4,44e-16 (0,00e+00)	30	-	-
ABC	10	7,19e-15 (1,18e-30)	30	8,24e-02	não
APSO-G	10	4,71e-15 (2,09e-30)	30	8,56e-03	sim
APSO-L	10	5,54e-15 (3,21e-30)	30	1,67e-07	sim
ClanAPSO-10x3-G	10	5,77e-15 (3,26e-30)	30	-	-
ClanAPSO-10x3-L	10	5,89e-15 (5,86e-30)	30	6,66e-06	sim
ClanAPSO-3x10-G	10	6,13e-15 (3,13e-30)	30	2,95e-06	sim
ClanAPSO-3x10-L	10	6,01e-15 (3,21e-30)	30	1,67e-07	sim
ClanAPSO-6x5-G	10	5,65e-15 (3,25e-30)	30	1,26e-09	sim
ClanAPSO-6x5-L	10	6,60e-15 (5,16e-30)	30	1,59e-04	sim
ClanPSO-10x3-G	10	4,83e-15 (2,34e-30)	30	3,54e-03	sim
ClanPSO-10x3-L	10	5,30e-15 (3,03e-30)	30	2,30e-05	sim
ClanPSO-3x10-G	10	4,94e-15 (2,55e-30)	30	1,32e-03	sim
ClanPSO-3x10-L	10	4,94e-15 (2,55e-30)	30	1,32e-03	sim
ClanPSO-6x5-G	10	5,06e-15 (2,74e-30)	30	4,26e-04	sim
ClanPSO-6x5-L	10	4,71e-15 (2,09e-30)	30	8,56e-03	sim
Constricted PSO-G	10	4,83e-15 (2,34e-30)	30	3,54e-03	sim
Constricted PSO-L	10	6,45e-01 (1,10e+01)	30	3,04e-01	não
CPSOS	10	4,62e+00 (5,39e+01)	30	5,94e-04	sim
FSS	10	5,10e-03 (2,16e-06)	30	4,35e-06	sim
GCPSO-G	10	4,71e-15 (2,09e-30)	30	8,56e-03	sim
GCPSO-L	10	1,34e+00 (2,29e+01)	30	1,39e-01	não
MCPSOCOL	10	1,08e+01 (4,54e+01)	30	3,60e-13	sim
PSO-G	10	4,94e-15 (2,55e-30)	30	1,32e-03	sim
PSO-L	10	5,42e-15 (3,13e-30)	30	2,95e-06	sim

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ABC	30	5,87e-14 (8,93e-29)	30	2,84e-08	sim
APSO-G	30	2,15e-14 (3,21e-29)	30	7,14e-06	sim
APSO-L	30	5,45e-05 (1,76e-08)	30	3,00e-02	sim
ClanAPSO-10x3-G	30	1,64e-14 (2,15e-29)	30	2,93e-03	sim
ClanAPSO-10x3-L	30	1,77e-14 (4,85e-29)	30	1,93e-04	sim
ClanAPSO-3x10-G	30	1,93e-14 (2,02e-29)	30	1,24e-08	sim
ClanAPSO-3x10-L	30	1,71e-14 (2,89e-29)	30	2,07e-04	sim
ClanAPSO-6x5-G	30	1,55e-14 (1,71e-29)	30	3,57e-04	sim
ClanAPSO-6x5-L	30	1,55e-14 (2,32e-29)	30	2,92e-04	sim
ClanPSO-10x3-G	30	2,21e-06 (3,28e-11)	30	5,87e-02	não
ClanPSO-10x3-L	30	1,38e-06 (6,10e-12)	30	1,54e-02	sim
ClanPSO-3x10-G	30	2,00e-07 (1,74e-13)	30	1,57e-02	sim
ClanPSO-3x10-L	30	1,14e-07 (4,55e-14)	30	8,57e-03	sim
ClanPSO-6x5-G	30	3,06e-07 (6,71e-13)	30	6,18e-02	não
ClanPSO-6x5-L	30	6,72e-08 (9,03e-15)	30	2,08e-03	sim
Constricted PSO-G	30	1,18e-14 (1,25e-29)	30	2,95e-06	sim
Constricted PSO-L	30	1,86e+01 (1,76e+01)	30	1,30e-01	não
CPSOS	30	1,35e+00 (9,39e+00)	30	1,91e-02	sim
FSS	30	2,50e-02 (3,56e-05)	30	1,69e-08	sim
GCPSO-G	30	1,18e-14 (1,60e-29)	30	1,01e-08	sim
GCPSO-L	30	1,94e+01 (2,38e+00)	30	1,74e-01	não
MCPSOCOL	30	1,98e+01 (4,53e-03)	30	7,90e-07	sim
PSO-G	30	3,30e-05 (2,41e-08)	30	2,64e-01	não
PSO-L	30	1,96e-14 (2,58e-29)	30	1,19e-06	sim
ABC	100	6,53e-13 (1,09e-25)	30	3,55e-06	sim
APSO-G	100	1,62e-02 (1,66e-04)	30	6,55e-04	sim
APSO-L	100	5,27e-01 (3,41e-01)	30	1,45e-07	sim
ClanAPSO-10x3-G	100	3,51e-03 (3,13e-06)	30	4,36e-06	sim
ClanAPSO-10x3-L	100	6,41e-05 (5,22e-09)	30	7,88e-03	sim
ClanAPSO-3x10-G	100	1,22e-04 (7,88e-09)	30	9,44e-07	sim
ClanAPSO-3x10-L	100	1,03e-04 (9,63e-09)	30	4,86e-04	sim
ClanAPSO-6x5-G	100	2,78e-04 (3,98e-08)	30	1,93e-05	sim
ClanAPSO-6x5-L	100	1,31e-05 (1,05e-10)	30	5,77e-06	sim
ClanPSO-10x3-G	100	1,62e+01 (4,71e+01)	30	1,06e-03	sim
ClanPSO-10x3-L	100	1,71e+01 (4,09e+01)	30	6,93e-03	sim
ClanPSO-3x10-G	100	9,84e+00 (5,48e+01)	30	1,21e-05	sim
ClanPSO-3x10-L	100	1,03e+01 (5,53e+01)	30	6,83e-06	sim
ClanPSO-6x5-G	100	1,32e+01 (6,09e+01)	30	5,07e-10	sim
ClanPSO-6x5-L	100	1,46e+01 (5,98e+01)	30	1,57e-05	sim
Constricted PSO-G	100	2,11e+01 (1,37e-02)	30	9,85e-08	sim
Constricted PSO-L	100	1,98e+01 (4,00e-04)	30	4,13e-08	sim
CPSOS	100	1,44e+00 (5,36e+00)	30	1,14e-03	sim
FSS	100	2,06e+01 (2,84e-01)	30	1,37e-06	sim
GCPSO-G	100	2,10e+01 (2,89e-02)	30	5,97e-07	sim
GCPSO-L	100	1,98e+01 (5,62e-04)	30	1,69e-08	sim
MCPSOCOL	100	2,04e+01 (7,08e-03)	30	3,00e-07	sim
PSO-G	100	1,12e+01 (4,89e+01)	30	3,43e-06	sim

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
PSO-L	100	1,98e+01 (3,18e-03)	30	1,16e-05	sim
ABC	300	6,51e-01 (1,25e-01)	30	8,95e-10	sim
APSO-G	300	3,86e+00 (6,50e-02)	30	1,47e-07	sim
APSO-L	300	2,46e+00 (7,19e-01)	30	4,81e-02	sim
ClanAPSO-10x3-G	300	1,12e+00 (4,69e-02)	30	2,17e-06	sim
ClanAPSO-10x3-L	300	2,59e-01 (6,28e-03)	30	1,70e-05	sim
ClanAPSO-3x10-G	300	1,17e+00 (4,17e-02)	30	5,18e-07	sim
ClanAPSO-3x10-L	300	1,14e+00 (3,36e-02)	30	9,06e-09	sim
ClanAPSO-6x5-G	300	8,63e-01 (4,30e-02)	30	1,18e-06	sim
ClanAPSO-6x5-L	300	3,61e-01 (2,00e-02)	30	3,54e-07	sim
ClanPSO-10x3-G	300	2,14e+01 (1,67e-03)	30	3,95e-05	sim
ClanPSO-10x3-L	300	2,14e+01 (1,36e-03)	30	8,97e-07	sim
ClanPSO-3x10-G	300	2,14e+01 (1,56e-03)	30	3,33e-06	sim
ClanPSO-3x10-L	300	2,14e+01 (1,56e-03)	30	1,37e-05	sim
ClanPSO-6x5-G	300	2,14e+01 (1,74e-03)	30	9,08e-08	sim
ClanPSO-6x5-L	300	2,14e+01 (1,38e-03)	30	2,72e-07	sim
Constricted PSO-G	300	2,14e+01 (4,08e-04)	30	1,35e-08	sim
Constricted PSO-L	300	2,14e+01 (4,08e-04)	30	1,35e-08	sim
CPSOS	300	1,88e+00 (2,36e+00)	30	1,13e-05	sim
FSS	300	2,13e+01 (7,41e-04)	30	5,96e-07	sim
GCPSO-G	300	2,15e+01 (4,88e-04)	30	6,96e-10	sim
GCPSO-L	300	2,15e+01 (4,88e-04)	30	6,96e-10	sim
MCPSOCOL	300	2,14e+01 (5,56e-04)	30	7,82e-06	sim
PSO-G	300	2,14e+01 (9,50e-04)	30	1,09e-07	sim
PSO-L	300	2,14e+01 (9,50e-04)	30	1,09e-07	sim
ABC	1000	1,96e+01 (6,83e-04)	30	1,14e-05	sim
APSO-G	1000	1,85e+01 (3,51e-02)	30	5,26e-07	sim
APSO-L	1000	1,79e+01 (1,33e-01)	30	2,10e-08	sim
ClanAPSO-10x3-G	1000	7,03e+00 (4,74e-02)	30	2,68e-08	sim
ClanAPSO-10x3-L	1000	5,67e+00 (4,85e-02)	30	1,90e-06	sim
ClanAPSO-3x10-G	1000	7,51e+00 (1,27e-01)	30	6,88e-07	sim
ClanAPSO-3x10-L	1000	7,21e+00 (8,75e-02)	30	3,08e-07	sim
ClanAPSO-6x5-G	1000	6,30e+00 (5,35e-02)	30	4,22e-06	sim
ClanAPSO-6x5-L	1000	5,40e+00 (5,41e-02)	30	3,42e-09	sim
ClanPSO-10x3-G	1000	2,15e+01 (2,41e-04)	30	1,87e-06	sim
ClanPSO-10x3-L	1000	2,15e+01 (2,17e-04)	30	3,21e-07	sim
ClanPSO-3x10-G	1000	2,15e+01 (3,24e-04)	30	2,44e-06	sim
ClanPSO-3x10-L	1000	2,15e+01 (2,98e-04)	30	3,89e-06	sim
ClanPSO-6x5-G	1000	2,15e+01 (1,58e-04)	30	1,39e-07	sim
ClanPSO-6x5-L	1000	2,15e+01 (1,39e-04)	30	5,57e-08	sim
Constricted PSO-G	1000	2,15e+01 (1,46e-04)	30	9,44e-10	sim
Constricted PSO-L	1000	2,15e+01 (1,46e-04)	30	9,44e-10	sim
CPSOS	1000	-	0	-	-
FSS	1000	2,15e+01 (5,01e-04)	30	1,13e-07	sim
GCPSO-G	1000	2,15e+01 (1,49e-04)	30	2,24e-07	sim
GCPSO-L	1000	2,15e+01 (1,49e-04)	30	2,24e-07	sim
MCPSOCOL	1000	-	0	-	-

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
PSO-G	1000	2,15e+01 (1,88e-04)	30	1,34e-06	sim
PSO-L	1000	2,15e+01 (1,88e-04)	30	1,34e-06	sim

Tabela 6.6: Resultados e significância estatística ($\alpha = 0,05$) para Griewank.

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ABC	2	0,00e+00 (0,00e+00)	30	-	-
APSO-G	2	0,00e+00 (0,00e+00)	30	-	-
APSO-L	2	1,23e-03 (7,86e-06)	30	1,92e-02	sim
ClanAPSO-10x3-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-10x3-L	2	2,47e-04 (1,82e-06)	30	3,34e-01	não
ClanAPSO-3x10-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-3x10-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-6x5-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanAPSO-6x5-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-10x3-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-10x3-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-3x10-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-3x10-L	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-6x5-G	2	0,00e+00 (0,00e+00)	30	-	-
ClanPSO-6x5-L	2	0,00e+00 (0,00e+00)	30	-	-
Constricted PSO-G	2	0,00e+00 (0,00e+00)	30	-	-
Constricted PSO-L	2	7,40e-04 (5,09e-06)	30	8,24e-02	não
CPSOS	2	1,74e-02 (1,68e-03)	30	7,82e-02	não
FSS	2	1,68e-03 (2,24e-06)	30	5,79e-06	sim
GCPSO-G	2	0,00e+00 (0,00e+00)	30	-	-
GCPSO-L	2	7,40e-04 (5,09e-06)	30	8,24e-02	não
MCPSOCOL	2	9,86e-04 (6,54e-06)	30	4,06e-02	sim
PSO-G	2	0,00e+00 (0,00e+00)	30	-	-
PSO-L	2	2,47e-04 (1,82e-06)	30	3,34e-01	não
ABC	10	3,83e-12 (2,52e-22)	30	2,01e-01	não
APSO-G	10	2,47e-02 (3,74e-04)	30	3,42e-07	sim
APSO-L	10	5,76e-02 (9,90e-04)	30	8,64e-07	sim
ClanAPSO-10x3-G	10	3,81e-02 (2,99e-04)	30	4,62e-08	sim
ClanAPSO-10x3-L	10	4,84e-02 (6,35e-04)	30	4,16e-08	sim
ClanAPSO-3x10-G	10	3,53e-02 (4,43e-04)	30	2,36e-05	sim
ClanAPSO-3x10-L	10	4,02e-02 (5,94e-04)	30	1,29e-06	sim
ClanAPSO-6x5-G	10	4,14e-02 (3,87e-04)	30	5,91e-06	sim
ClanAPSO-6x5-L	10	4,76e-02 (6,32e-04)	30	7,19e-08	sim
ClanPSO-10x3-G	10	4,55e-02 (1,91e-04)	30	1,28e-07	sim
ClanPSO-10x3-L	10	4,01e-02 (4,87e-04)	30	4,45e-05	sim
ClanPSO-3x10-G	10	6,32e-02 (1,90e-03)	30	1,03e-03	sim
ClanPSO-3x10-L	10	8,90e-02 (3,61e-03)	30	2,17e-05	sim
ClanPSO-6x5-G	10	4,83e-02 (6,19e-04)	30	6,20e-09	sim
ClanPSO-6x5-L	10	5,18e-02 (3,73e-04)	30	2,32e-09	sim
Constricted PSO-G	10	1,39e-02 (1,71e-04)	30	5,55e-07	sim

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
Constricted PSO-L	10	7,90e-02 (2,19e-03)	30	2,09e-05	sim
CPSOS	10	1,17e-01 (1,77e-02)	30	3,62e-03	sim
FSS	10	2,59e-02 (2,15e-04)	30	2,86e-06	sim
GCPSO-G	10	8,68e-03 (8,71e-05)	30	5,27e-06	sim
GCPSO-L	10	7,15e-02 (1,86e-03)	30	5,10e-05	sim
MCPSOCOL	10	1,59e-01 (1,07e-02)	30	1,38e-05	sim
PSO-G	10	1,92e-01 (1,27e-02)	30	4,27e-08	sim
PSO-L	10	5,23e-02 (5,82e-04)	30	1,67e-09	sim
ABC	30	1,10e-13 (1,07e-25)	30	7,66e-02	não
APSO-G	30	1,10e-02 (2,05e-04)	30	8,32e-05	sim
APSO-L	30	2,54e-02 (5,33e-04)	30	2,23e-06	sim
ClanAPSO-10x3-G	30	2,02e-02 (3,42e-04)	30	2,36e-05	sim
ClanAPSO-10x3-L	30	1,43e-02 (2,66e-04)	30	4,00e-06	sim
ClanAPSO-3x10-G	30	1,42e-02 (1,98e-04)	30	1,05e-05	sim
ClanAPSO-3x10-L	30	1,67e-02 (4,83e-04)	30	5,83e-05	sim
ClanAPSO-6x5-G	30	2,13e-02 (4,23e-04)	30	1,40e-06	sim
ClanAPSO-6x5-L	30	1,73e-02 (5,40e-04)	30	8,25e-05	sim
ClanPSO-10x3-G	30	9,65e-03 (1,55e-04)	30	1,31e-05	sim
ClanPSO-10x3-L	30	7,62e-03 (1,37e-04)	30	7,64e-04	sim
ClanPSO-3x10-G	30	6,49e-03 (1,90e-04)	30	1,19e-02	sim
ClanPSO-3x10-L	30	1,92e-02 (9,00e-04)	30	9,94e-04	sim
ClanPSO-6x5-G	30	8,44e-03 (1,26e-04)	30	6,30e-05	sim
ClanPSO-6x5-L	30	5,36e-03 (8,25e-05)	30	1,39e-03	sim
Constricted PSO-G	30	9,86e-04 (9,46e-06)	30	8,96e-02	não
Constricted PSO-L	30	4,41e-02 (1,29e-03)	30	6,27e-10	sim
CPSOS	30	1,30e-01 (2,06e-02)	30	8,08e-04	sim
FSS	30	6,01e-02 (2,96e-04)	30	3,63e-06	sim
GCPSO-G	30	2,55e-03 (2,65e-05)	30	8,23e-03	sim
GCPSO-L	30	7,18e-02 (8,89e-03)	30	5,57e-04	sim
MCPSOCOL	30	1,03e+02 (1,17e+03)	30	1,11e-09	sim
PSO-G	30	4,00e-02 (1,09e-02)	30	5,48e-02	não
PSO-L	30	1,56e-02 (2,86e-04)	30	9,60e-05	sim
ABC	100	1,72e-12 (2,59e-23)	30	7,94e-02	não
APSO-G	100	2,79e-02 (2,67e-03)	30	3,73e-03	sim
APSO-L	100	6,73e-02 (6,25e-03)	30	1,77e-05	sim
ClanAPSO-10x3-G	100	1,71e-02 (5,36e-04)	30	2,52e-04	sim
ClanAPSO-10x3-L	100	6,81e-03 (1,19e-04)	30	6,76e-04	sim
ClanAPSO-3x10-G	100	1,78e-02 (1,05e-03)	30	5,34e-03	sim
ClanAPSO-3x10-L	100	7,63e-03 (1,32e-04)	30	2,66e-04	sim
ClanAPSO-6x5-G	100	1,09e-02 (2,79e-04)	30	3,59e-04	sim
ClanAPSO-6x5-L	100	1,10e-02 (2,04e-04)	30	1,13e-05	sim
ClanPSO-10x3-G	100	2,70e+00 (2,39e+00)	30	1,23e-04	sim
ClanPSO-10x3-L	100	2,03e+00 (5,78e-01)	30	3,72e-05	sim
ClanPSO-3x10-G	100	1,88e+00 (4,09e+00)	30	1,75e-02	sim
ClanPSO-3x10-L	100	1,20e+00 (8,06e-02)	30	2,95e-04	sim
ClanPSO-6x5-G	100	1,55e+00 (4,01e-01)	30	8,46e-04	sim
ClanPSO-6x5-L	100	1,35e+00 (1,57e-01)	30	1,49e-03	sim

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
Constricted PSO-G	100	7,87e-02 (2,54e-02)	30	8,93e-03	sim
Constricted PSO-L	100	1,38e+03 (1,81e+05)	30	4,05e-07	sim
CPSOS	100	2,63e-01 (3,23e-01)	30	2,55e-02	sim
FSS	100	6,50e-01 (3,14e-03)	30	4,60e-07	sim
GCPSO-G	100	9,42e-02 (3,91e-02)	30	1,35e-02	sim
GCPSO-L	100	1,29e+03 (1,97e+05)	30	6,77e-07	sim
MCPSOCOL	100	2,09e+03 (1,04e+05)	30	4,29e-10	sim
PSO-G	100	4,93e+00 (9,58e+01)	30	4,88e-02	sim
PSO-L	100	1,14e-02 (3,32e-04)	30	8,36e-04	sim
ABC	300	4,33e-06 (4,74e-10)	30	2,98e-01	não
APSO-G	300	1,70e+00 (3,14e-02)	30	9,60e-06	sim
APSO-L	300	4,53e-01 (1,12e-01)	30	1,60e-07	sim
ClanAPSO-10x3-G	300	9,23e-01 (1,63e-02)	30	1,56e-07	sim
ClanAPSO-10x3-L	300	2,77e-01 (4,08e-03)	30	5,04e-05	sim
ClanAPSO-3x10-G	300	3,11e-01 (5,98e-03)	30	2,03e-06	sim
ClanAPSO-3x10-L	300	2,63e-01 (7,04e-03)	30	5,60e-06	sim
ClanAPSO-6x5-G	300	3,45e-01 (4,98e-03)	30	1,93e-06	sim
ClanAPSO-6x5-L	300	1,44e-01 (1,34e-03)	30	1,55e-07	sim
ClanPSO-10x3-G	300	1,43e+04 (8,03e+04)	30	1,52e-07	sim
ClanPSO-10x3-L	300	1,42e+04 (8,17e+04)	30	2,38e-08	sim
ClanPSO-3x10-G	300	1,43e+04 (6,38e+04)	30	3,10e-07	sim
ClanPSO-3x10-L	300	1,42e+04 (5,25e+04)	30	4,14e-07	sim
ClanPSO-6x5-G	300	1,43e+04 (6,86e+04)	30	1,68e-05	sim
ClanPSO-6x5-L	300	1,43e+04 (7,76e+04)	30	1,33e-06	sim
Constricted PSO-G	300	1,49e+04 (2,80e+04)	30	1,32e-08	sim
Constricted PSO-L	300	1,49e+04 (2,80e+04)	30	1,32e-08	sim
CPSOS	300	1,37e+00 (3,37e+01)	30	2,35e-01	não
FSS	300	1,24e+04 (7,93e+05)	30	2,01e-06	sim
GCPSO-G	300	1,50e+04 (2,50e+04)	30	5,62e-07	sim
GCPSO-L	300	1,50e+04 (2,50e+04)	30	5,62e-07	sim
MCPSOCOL	300	1,49e+04 (2,19e+04)	30	6,56e-07	sim
PSO-G	300	1,43e+04 (8,27e+04)	30	3,33e-08	sim
PSO-L	300	1,43e+04 (1,47e+05)	30	1,42e-05	sim
ABC	1000	2,03e+01 (4,27e+02)	30	3,38e-03	sim
APSO-G	1000	1,05e+03 (2,74e+04)	30	1,83e-05	sim
APSO-L	1000	4,63e+02 (1,95e+04)	30	7,12e-08	sim
ClanAPSO-10x3-G	1000	4,65e+01 (7,49e+00)	30	8,57e-08	sim
ClanAPSO-10x3-L	1000	2,11e+01 (2,27e+00)	30	7,07e-08	sim
ClanAPSO-3x10-G	1000	4,12e+01 (1,30e+01)	30	2,82e-06	sim
ClanAPSO-3x10-L	1000	3,26e+01 (8,53e+00)	30	2,54e-07	sim
ClanAPSO-6x5-G	1000	3,18e+01 (5,33e+00)	30	3,90e-07	sim
ClanAPSO-6x5-L	1000	1,62e+01 (1,37e+00)	30	1,20e-07	sim
ClanPSO-10x3-G	1000	5,06e+04 (9,92e+04)	30	1,23e-07	sim
ClanPSO-10x3-L	1000	5,06e+04 (8,06e+04)	30	3,32e-07	sim
ClanPSO-3x10-G	1000	5,06e+04 (9,18e+04)	30	1,51e-06	sim
ClanPSO-3x10-L	1000	5,06e+04 (9,26e+04)	30	1,45e-06	sim
ClanPSO-6x5-G	1000	5,06e+04 (7,82e+04)	30	3,06e-05	sim

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ClanPSO-6x5-L	1000	5,06e+04 (9,35e+04)	30	1,85e-06	sim
Constricted PSO-G	1000	5,11e+04 (5,87e+04)	30	1,86e-06	sim
Constricted PSO-L	1000	5,11e+04 (5,87e+04)	30	1,86e-06	sim
CPSOS	1000	-	0	-	-
FSS	1000	5,12e+04 (8,45e+04)	30	1,02e-07	sim
GCPSO-G	1000	5,12e+04 (5,40e+04)	30	1,25e-05	sim
GCPSO-L	1000	5,12e+04 (5,40e+04)	30	1,25e-05	sim
MCPSOCOL	1000	-	0	-	-
PSO-G	1000	5,07e+04 (6,73e+04)	30	7,55e-07	sim
PSO-L	1000	5,07e+04 (6,73e+04)	30	7,55e-07	sim

Tabela 6.7: Resultados e significância estatística ($\alpha = 0,05$) para P8.

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ABC	2	9,59e-20 (7,77e-39)	30	1,41e-04	sim
APSO-G	2	2,36e-31 (0,00e+00)	30	-	-
APSO-L	2	2,36e-31 (0,00e+00)	30	-	-
ClanAPSO-10x3-G	2	2,36e-31 (0,00e+00)	30	-	-
ClanAPSO-10x3-L	2	2,36e-31 (0,00e+00)	30	-	-
ClanAPSO-3x10-G	2	2,36e-31 (0,00e+00)	30	-	-
ClanAPSO-3x10-L	2	2,36e-31 (0,00e+00)	30	-	-
ClanAPSO-6x5-G	2	2,36e-31 (0,00e+00)	30	-	-
ClanAPSO-6x5-L	2	2,36e-31 (0,00e+00)	30	-	-
ClanPSO-10x3-G	2	2,36e-31 (0,00e+00)	30	-	-
ClanPSO-10x3-L	2	2,36e-31 (0,00e+00)	30	-	-
ClanPSO-3x10-G	2	2,36e-31 (0,00e+00)	30	-	-
ClanPSO-3x10-L	2	2,36e-31 (0,00e+00)	30	-	-
ClanPSO-6x5-G	2	2,36e-31 (0,00e+00)	30	-	-
ClanPSO-6x5-L	2	2,36e-31 (0,00e+00)	30	-	-
Constricted PSO-G	2	2,36e-31 (0,00e+00)	30	-	-
Constricted PSO-L	2	2,36e-31 (0,00e+00)	30	-	-
CPSOS	2	7,09e-04 (1,51e-05)	30	3,34e-01	não
FSS	2	7,24e-07 (8,02e-13)	30	1,45e-04	sim
GCPSO-G	2	2,36e-31 (0,00e+00)	30	-	-
GCPSO-L	2	2,36e-31 (0,00e+00)	30	-	-
MCPSOCOL	2	2,36e-31 (0,00e+00)	30	-	-
PSO-G	2	2,36e-31 (0,00e+00)	30	-	-
PSO-L	2	2,36e-31 (0,00e+00)	30	-	-
ABC	10	6,31e-17 (1,86e-34)	30	2,20e-08	sim
APSO-G	10	4,71e-32 (0,00e+00)	30	-	-
APSO-L	10	4,71e-32 (0,00e+00)	30	-	-
ClanAPSO-10x3-G	10	4,71e-32 (0,00e+00)	30	-	-
ClanAPSO-10x3-L	10	4,76e-32 (8,00e-66)	30	3,34e-01	não
ClanAPSO-3x10-G	10	4,71e-32 (0,00e+00)	30	-	-
ClanAPSO-3x10-L	10	4,71e-32 (0,00e+00)	30	-	-
ClanAPSO-6x5-G	10	4,71e-32 (0,00e+00)	30	-	-

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ClanAPSO-6x5-L	10	4,71e-32 (0,00e+00)	30	-	-
ClanPSO-10x3-G	10	4,71e-32 (0,00e+00)	30	-	-
ClanPSO-10x3-L	10	4,71e-32 (0,00e+00)	30	-	-
ClanPSO-3x10-G	10	4,71e-32 (0,00e+00)	30	-	-
ClanPSO-3x10-L	10	4,71e-32 (0,00e+00)	30	-	-
ClanPSO-6x5-G	10	4,71e-32 (0,00e+00)	30	-	-
ClanPSO-6x5-L	10	4,71e-32 (0,00e+00)	30	-	-
Constricted PSO-G	10	4,71e-32 (0,00e+00)	30	-	-
Constricted PSO-L	10	3,11e-02 (9,01e-03)	30	8,24e-02	não
CPSOS	10	6,29e-02 (5,91e-02)	30	1,71e-01	não
FSS	10	4,83e-03 (3,48e-04)	30	1,72e-01	não
GCPSO-G	10	4,71e-32 (0,00e+00)	30	-	-
GCPSO-L	10	2,07e-02 (6,23e-03)	30	1,64e-01	não
MCPSOCOL	10	2,03e+00 (2,75e+00)	30	2,40e-06	sim
PSO-G	10	4,71e-32 (0,00e+00)	30	-	-
PSO-L	10	4,71e-32 (0,00e+00)	30	-	-
ABC	30	6,74e-16 (1,06e-32)	30	6,67e-07	sim
APSO-G	30	6,72e-30 (1,35e-57)	30	3,34e-01	não
APSO-L	30	8,01e-12 (1,92e-21)	30	3,34e-01	não
ClanAPSO-10x3-G	30	1,57e-32 (5,55e-68)	30	3,34e-01	não
ClanAPSO-10x3-L	30	7,47e-32 (7,84e-62)	30	2,68e-01	não
ClanAPSO-3x10-G	30	5,16e-32 (3,78e-62)	30	3,29e-01	não
ClanAPSO-3x10-L	30	1,63e-32 (2,72e-66)	30	7,23e-02	não
ClanAPSO-6x5-G	30	1,76e-32 (7,26e-65)	30	2,40e-01	não
ClanAPSO-6x5-L	30	1,61e-32 (1,74e-66)	30	1,20e-01	não
ClanPSO-10x3-G	30	1,39e-13 (1,69e-25)	30	7,54e-02	não
ClanPSO-10x3-L	30	7,59e-13 (5,23e-24)	30	8,13e-02	não
ClanPSO-3x10-G	30	1,26e-14 (1,27e-27)	30	6,14e-02	não
ClanPSO-3x10-L	30	3,13e-15 (1,58e-28)	30	1,90e-01	não
ClanPSO-6x5-G	30	1,24e-14 (1,81e-27)	30	1,23e-01	não
ClanPSO-6x5-L	30	2,92e-15 (1,07e-28)	30	1,37e-01	não
Constricted PSO-G	30	6,91e-03 (6,92e-04)	30	1,64e-01	não
Constricted PSO-L	30	2,84e-01 (1,82e-01)	30	5,16e-04	sim
CPSOS	30	8,73e-03 (9,14e-04)	30	1,27e-01	não
FSS	30	3,16e-02 (3,12e-03)	30	2,45e-03	sim
GCPSO-G	30	3,46e-03 (3,58e-04)	30	3,34e-01	não
GCPSO-L	30	4,20e-01 (6,67e-01)	30	6,80e-03	sim
MCPSOCOL	30	4,97e+06 (3,80e+13)	30	2,23e-04	sim
PSO-G	30	7,54e-09 (7,44e-16)	30	1,43e-01	não
PSO-L	30	1,38e-02 (1,28e-03)	30	4,06e-02	sim
ABC	100	7,94e-15 (1,07e-29)	30	3,37e-06	sim
APSO-G	100	1,05e-03 (3,22e-05)	30	3,31e-01	não
APSO-L	100	1,25e-07 (4,69e-13)	30	3,34e-01	não
ClanAPSO-10x3-G	100	2,24e-07 (3,87e-14)	30	3,82e-06	sim
ClanAPSO-10x3-L	100	1,11e-10 (1,62e-20)	30	2,13e-04	sim
ClanAPSO-3x10-G	100	8,20e-11 (1,69e-20)	30	1,28e-02	sim
ClanAPSO-3x10-L	100	2,54e-09 (1,45e-16)	30	2,71e-01	não

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ClanAPSO-6x5-G	100	1,11e-09 (1,76e-18)	30	1,52e-03	sim
ClanAPSO-6x5-L	100	4,43e-12 (1,46e-23)	30	5,96e-06	sim
ClanPSO-10x3-G	100	1,06e+01 (7,53e+00)	30	3,33e-07	sim
ClanPSO-10x3-L	100	9,60e+00 (6,84e+00)	30	2,39e-05	sim
ClanPSO-3x10-G	100	1,24e+02 (1,56e+05)	30	1,26e-01	não
ClanPSO-3x10-L	100	2,33e+01 (2,77e+03)	30	1,05e-01	não
ClanPSO-6x5-G	100	1,15e+01 (2,02e+01)	30	2,70e-04	sim
ClanPSO-6x5-L	100	1,33e+01 (1,44e+02)	30	2,18e-04	sim
Constricted PSO-G	100	9,25e+00 (1,71e+03)	30	2,71e-01	não
Constricted PSO-L	100	2,99e+08 (1,21e+17)	30	1,74e-05	sim
CPSOS	100	2,26e+02 (1,53e+06)	30	3,34e-01	não
FSS	100	2,30e+00 (5,85e-01)	30	5,87e-09	sim
GCPSO-G	100	2,78e+00 (1,23e+01)	30	1,03e-03	sim
GCPSO-L	100	3,95e+08 (6,44e+16)	30	5,82e-06	sim
MCPSOCOL	100	1,29e+09 (1,45e+17)	30	2,83e-04	sim
PSO-G	100	2,72e+02 (2,04e+05)	30	3,84e-03	sim
PSO-L	100	3,11e-01 (1,17e-01)	30	7,36e-05	sim
ABC	300	1,48e-09 (3,63e-18)	30	1,51e-03	sim
APSO-G	300	3,54e+00 (1,14e+00)	30	1,65e-06	sim
APSO-L	300	2,66e-02 (4,78e-04)	30	2,10e-06	sim
ClanAPSO-10x3-G	300	1,05e-02 (1,29e-05)	30	1,61e-05	sim
ClanAPSO-10x3-L	300	2,26e-03 (3,78e-06)	30	4,78e-02	sim
ClanAPSO-3x10-G	300	9,29e-03 (2,53e-05)	30	2,21e-05	sim
ClanAPSO-3x10-L	300	7,18e-03 (1,41e-05)	30	1,52e-03	sim
ClanAPSO-6x5-G	300	5,73e-03 (1,22e-05)	30	8,68e-04	sim
ClanAPSO-6x5-L	300	2,76e-03 (1,14e-05)	30	2,09e-02	sim
ClanPSO-10x3-G	300	2,00e+10 (7,41e+17)	30	9,07e-06	sim
ClanPSO-10x3-L	300	1,98e+10 (6,49e+17)	30	8,93e-06	sim
ClanPSO-3x10-G	300	2,01e+10 (6,59e+17)	30	4,32e-06	sim
ClanPSO-3x10-L	300	2,01e+10 (7,15e+17)	30	1,03e-06	sim
ClanPSO-6x5-G	300	2,00e+10 (4,71e+17)	30	7,53e-06	sim
ClanPSO-6x5-L	300	2,00e+10 (6,47e+17)	30	7,57e-07	sim
Constricted PSO-G	300	2,15e+10 (3,77e+17)	30	9,42e-06	sim
Constricted PSO-L	300	2,15e+10 (3,77e+17)	30	9,42e-06	sim
CPSOS	300	6,94e+03 (1,37e+09)	30	3,21e-01	não
FSS	300	1,26e+10 (4,02e+18)	30	8,25e-08	sim
GCPSO-G	300	2,17e+10 (3,25e+17)	30	1,49e-06	sim
GCPSO-L	300	2,17e+10 (3,25e+17)	30	1,49e-06	sim
MCPSOCOL	300	2,14e+10 (2,51e+17)	30	1,64e-08	sim
PSO-G	300	2,02e+10 (3,89e+17)	30	8,57e-08	sim
PSO-L	300	2,01e+10 (7,49e+17)	30	1,41e-05	sim
ABC	1000	4,19e+02 (1,99e+06)	30	1,16e-01	não
APSO-G	1000	8,14e+08 (6,19e+16)	30	9,97e-06	sim
APSO-L	1000	5,95e+08 (1,34e+17)	30	6,14e-05	sim
ClanAPSO-10x3-G	1000	3,49e+02 (2,43e+06)	30	2,65e-01	não
ClanAPSO-10x3-L	1000	1,17e+01 (5,29e+00)	30	1,05e-05	sim
ClanAPSO-3x10-G	1000	9,94e+03 (1,92e+09)	30	2,38e-01	não

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ClanAPSO-3x10-L	1000	1,45e+03 (9,87e+06)	30	1,85e-02	sim
ClanAPSO-6x5-G	1000	2,10e+02 (1,02e+06)	30	3,05e-01	não
ClanAPSO-6x5-L	1000	1,03e+01 (1,51e+01)	30	2,78e-03	sim
ClanPSO-10x3-G	1000	7,52e+10 (6,07e+17)	30	2,01e-06	sim
ClanPSO-10x3-L	1000	7,52e+10 (6,16e+17)	30	1,92e-06	sim
ClanPSO-3x10-G	1000	7,53e+10 (7,40e+17)	30	5,54e-07	sim
ClanPSO-3x10-L	1000	7,53e+10 (6,34e+17)	30	3,33e-07	sim
ClanPSO-6x5-G	1000	7,52e+10 (8,23e+17)	30	5,63e-08	sim
ClanPSO-6x5-L	1000	7,52e+10 (7,90e+17)	30	9,64e-08	sim
Constricted PSO-G	1000	7,64e+10 (6,26e+17)	30	1,12e-07	sim
Constricted PSO-L	1000	7,64e+10 (6,26e+17)	30	1,12e-07	sim
CPSOS	1000	-	0	-	-
FSS	1000	7,69e+10 (1,59e+18)	30	4,74e-08	sim
GCPSO-G	1000	7,67e+10 (6,18e+17)	30	3,47e-07	sim
GCPSO-L	1000	7,67e+10 (6,18e+17)	30	3,47e-07	sim
MCPSOCOL	1000	-	0	-	-
PSO-G	1000	7,54e+10 (7,36e+17)	30	2,68e-06	sim
PSO-L	1000	7,54e+10 (7,36e+17)	30	2,68e-06	sim

Tabela 6.8: Resultados e significância estatística ($\alpha = 0,05$) para P16.

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ABC	2	1,04e-19 (1,03e-38)	30	2,19e-04	sim
APSO-G	2	1,35e-32 (0,00e+00)	30	-	-
APSO-L	2	1,35e-32 (0,00e+00)	30	-	-
ClanAPSO-10x3-G	2	1,35e-32 (0,00e+00)	30	-	-
ClanAPSO-10x3-L	2	1,35e-32 (0,00e+00)	30	-	-
ClanAPSO-3x10-G	2	1,35e-32 (0,00e+00)	30	-	-
ClanAPSO-3x10-L	2	1,35e-32 (0,00e+00)	30	-	-
ClanAPSO-6x5-G	2	1,35e-32 (0,00e+00)	30	-	-
ClanAPSO-6x5-L	2	1,35e-32 (0,00e+00)	30	-	-
ClanPSO-10x3-G	2	1,35e-32 (0,00e+00)	30	-	-
ClanPSO-10x3-L	2	1,35e-32 (0,00e+00)	30	-	-
ClanPSO-3x10-G	2	1,35e-32 (0,00e+00)	30	-	-
ClanPSO-3x10-L	2	1,35e-32 (0,00e+00)	30	-	-
ClanPSO-6x5-G	2	1,35e-32 (0,00e+00)	30	-	-
ClanPSO-6x5-L	2	1,35e-32 (0,00e+00)	30	-	-
Constricted PSO-G	2	1,35e-32 (0,00e+00)	30	-	-
Constricted PSO-L	2	1,35e-32 (0,00e+00)	30	-	-
CPSOS	2	1,35e-32 (0,00e+00)	30	-	-
FSS	2	2,63e-06 (2,82e-11)	30	1,66e-02	sim
GCPSO-G	2	1,35e-32 (0,00e+00)	30	-	-
GCPSO-L	2	1,35e-32 (0,00e+00)	30	-	-
MCPSOCOL	2	1,35e-32 (0,00e+00)	30	-	-
PSO-G	2	1,35e-32 (0,00e+00)	30	-	-
PSO-L	2	1,35e-32 (0,00e+00)	30	-	-

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
ABC	10	5,81e-17 (2,47e-34)	30	1,32e-07	sim
APSO-G	10	1,35e-32 (0,00e+00)	30	-	-
APSO-L	10	1,37e-32 (8,80e-67)	30	1,64e-01	não
ClanAPSO-10x3-G	10	1,37e-32 (2,18e-67)	30	1,92e-02	sim
ClanAPSO-10x3-L	10	2,62e-32 (2,32e-63)	30	1,63e-01	não
ClanAPSO-3x10-G	10	2,01e-32 (1,20e-63)	30	3,15e-01	não
ClanAPSO-3x10-L	10	1,38e-32 (2,81e-67)	30	3,54e-03	sim
ClanAPSO-6x5-G	10	1,36e-32 (1,41e-67)	30	8,24e-02	não
ClanAPSO-6x5-L	10	2,65e-32 (2,32e-63)	30	1,53e-01	não
ClanPSO-10x3-G	10	1,35e-32 (0,00e+00)	30	-	-
ClanPSO-10x3-L	10	1,35e-32 (0,00e+00)	30	-	-
ClanPSO-3x10-G	10	1,35e-32 (0,00e+00)	30	-	-
ClanPSO-3x10-L	10	1,35e-32 (0,00e+00)	30	-	-
ClanPSO-6x5-G	10	1,35e-32 (0,00e+00)	30	-	-
ClanPSO-6x5-L	10	1,35e-32 (0,00e+00)	30	-	-
Constricted PSO-G	10	1,35e-32 (0,00e+00)	30	-	-
Constricted PSO-L	10	3,66e-04 (4,02e-06)	30	3,34e-01	não
CPSOS	10	8,01e-04 (7,81e-06)	30	1,29e-01	não
FSS	10	7,15e-04 (4,31e-06)	30	1,04e-01	não
GCPSO-G	10	1,35e-32 (0,00e+00)	30	-	-
GCPSO-L	10	1,35e-32 (0,00e+00)	30	-	-
MCPSOCOL	10	5,29e-01 (1,23e+00)	30	1,29e-02	sim
PSO-G	10	1,35e-32 (0,00e+00)	30	-	-
PSO-L	10	1,35e-32 (0,00e+00)	30	-	-
ABC	30	6,41e-16 (1,49e-32)	30	1,44e-07	sim
APSO-G	30	7,32e-04 (7,77e-06)	30	1,64e-01	não
APSO-L	30	3,66e-03 (2,78e-05)	30	1,14e-04	sim
ClanAPSO-10x3-G	30	8,69e-22 (2,26e-41)	30	3,34e-01	não
ClanAPSO-10x3-L	30	3,66e-04 (4,02e-06)	30	3,34e-01	não
ClanAPSO-3x10-G	30	1,46e-03 (1,44e-05)	30	4,06e-02	sim
ClanAPSO-3x10-L	30	1,83e-03 (1,73e-05)	30	1,92e-02	sim
ClanAPSO-6x5-G	30	3,66e-04 (4,02e-06)	30	3,34e-01	não
ClanAPSO-6x5-L	30	8,75e-32 (8,95e-63)	30	1,19e-04	sim
ClanPSO-10x3-G	30	3,66e-04 (4,02e-06)	30	3,34e-01	não
ClanPSO-10x3-L	30	1,13e-10 (6,78e-20)	30	2,58e-02	sim
ClanPSO-3x10-G	30	8,69e-11 (2,23e-19)	30	3,30e-01	não
ClanPSO-3x10-L	30	3,66e-04 (4,02e-06)	30	3,34e-01	não
ClanPSO-6x5-G	30	2,06e-12 (2,08e-23)	30	1,69e-02	sim
ClanPSO-6x5-L	30	3,49e-12 (1,14e-22)	30	8,36e-02	não
Constricted PSO-G	30	7,32e-04 (7,77e-06)	30	1,64e-01	não
Constricted PSO-L	30	1,57e-01 (1,16e-01)	30	1,45e-02	sim
CPSOS	30	1,21e+06 (4,40e+13)	30	3,34e-01	não
FSS	30	9,03e-03 (3,97e-05)	30	8,07e-12	sim
GCPSO-G	30	1,36e-32 (4,56e-67)	30	3,34e-01	não
GCPSO-L	30	4,78e-02 (3,05e-02)	30	1,47e-01	não
MCPSOCOL	30	2,87e+07 (5,97e+14)	30	2,20e-06	sim
PSO-G	30	1,10e-03 (1,12e-05)	30	8,24e-02	não

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
PSO-L	30	2,20e-03 (2,00e-05)	30	8,56e-03	sim
ABC	100	7,10e-15 (1,22e-29)	30	1,66e-03	sim
APSO-G	100	4,68e-03 (2,87e-05)	30	3,18e-07	sim
APSO-L	100	1,10e-02 (2,83e-04)	30	3,38e-04	sim
ClanAPSO-10x3-G	100	1,51e-03 (1,46e-05)	30	3,68e-02	sim
ClanAPSO-10x3-L	100	2,93e-03 (2,44e-05)	30	1,32e-03	sim
ClanAPSO-3x10-G	100	1,86e-03 (1,73e-05)	30	1,72e-02	sim
ClanAPSO-3x10-L	100	5,13e-03 (3,11e-05)	30	1,26e-09	sim
ClanAPSO-6x5-G	100	1,83e-03 (1,73e-05)	30	1,92e-02	sim
ClanAPSO-6x5-L	100	1,83e-03 (1,73e-05)	30	1,91e-02	sim
ClanPSO-10x3-G	100	4,07e+02 (1,82e+05)	30	1,13e-03	sim
ClanPSO-10x3-L	100	5,16e+02 (4,40e+05)	30	2,29e-03	sim
ClanPSO-3x10-G	100	6,94e+03 (1,41e+08)	30	4,88e-03	sim
ClanPSO-3x10-L	100	6,06e+03 (8,98e+07)	30	1,23e-03	sim
ClanPSO-6x5-G	100	1,92e+03 (2,20e+07)	30	5,63e-02	não
ClanPSO-6x5-L	100	2,31e+03 (2,00e+07)	30	1,95e-02	sim
Constricted PSO-G	100	3,81e+00 (4,32e+01)	30	3,15e-03	sim
Constricted PSO-L	100	9,48e+08 (5,79e+17)	30	4,74e-06	sim
CPSOS	100	2,88e+05 (2,49e+12)	30	3,34e-01	não
FSS	100	1,72e-01 (3,33e-03)	30	2,33e-05	sim
GCPSO-G	100	2,48e+00 (2,49e+01)	30	1,41e-02	sim
GCPSO-L	100	9,21e+08 (4,57e+17)	30	3,65e-06	sim
MCPSOCOL	100	3,54e+09 (7,26e+17)	30	2,02e-09	sim
PSO-G	100	1,47e+05 (3,85e+10)	30	2,65e-04	sim
PSO-L	100	9,53e+00 (6,25e+01)	30	4,66e-06	sim
ABC	300	6,92e-08 (7,35e-15)	30	9,97e-04	sim
APSO-G	300	6,17e+02 (2,04e+06)	30	1,31e-01	não
APSO-L	300	2,72e+00 (1,41e+00)	30	5,48e-06	sim
ClanAPSO-10x3-G	300	3,36e+00 (5,35e-01)	30	9,36e-07	sim
ClanAPSO-10x3-L	300	5,51e-01 (2,06e-02)	30	3,26e-05	sim
ClanAPSO-3x10-G	300	4,16e+00 (2,93e+00)	30	7,74e-03	sim
ClanAPSO-3x10-L	300	2,86e+00 (4,85e-01)	30	5,74e-09	sim
ClanAPSO-6x5-G	300	2,39e+00 (2,75e-01)	30	2,78e-09	sim
ClanAPSO-6x5-L	300	6,91e-01 (4,54e-02)	30	8,49e-07	sim
ClanPSO-10x3-G	300	3,62e+10 (2,05e+18)	30	3,52e-05	sim
ClanPSO-10x3-L	300	3,60e+10 (1,72e+18)	30	2,70e-05	sim
ClanPSO-3x10-G	300	3,66e+10 (1,58e+18)	30	1,85e-06	sim
ClanPSO-3x10-L	300	3,66e+10 (1,70e+18)	30	5,65e-07	sim
ClanPSO-6x5-G	300	3,63e+10 (1,56e+18)	30	1,60e-05	sim
ClanPSO-6x5-L	300	3,62e+10 (1,80e+18)	30	4,49e-06	sim
Constricted PSO-G	300	3,89e+10 (9,90e+17)	30	7,00e-06	sim
Constricted PSO-L	300	3,89e+10 (9,90e+17)	30	7,00e-06	sim
CPSOS	300	6,70e+05 (1,35e+13)	30	3,34e-01	não
FSS	300	2,44e+10 (1,41e+19)	30	1,52e-08	sim
GCPSO-G	300	3,93e+10 (8,10e+17)	30	9,05e-07	sim
GCPSO-L	300	3,93e+10 (8,10e+17)	30	9,05e-07	sim
MCPSOCOL	300	3,90e+10 (8,67e+17)	30	3,39e-08	sim

Algoritmo	Dim.	Média (Desvio)	Exp.	Nível-p	Sig.
PSO-G	300	3,69e+10 (1,17e+18)	30	4,67e-06	sim
PSO-L	300	3,67e+10 (1,85e+18)	30	4,54e-05	sim
ABC	1000	1,46e+05 (1,58e+11)	30	5,11e-02	não
APSO-G	1000	1,63e+09 (3,31e+17)	30	5,00e-07	sim
APSO-L	1000	9,59e+08 (2,74e+17)	30	4,08e-07	sim
ClanAPSO-10x3-G	1000	4,63e+04 (3,90e+08)	30	6,46e-08	sim
ClanAPSO-10x3-L	1000	5,84e+03 (1,43e+07)	30	4,09e-04	sim
ClanAPSO-3x10-G	1000	1,34e+05 (3,38e+09)	30	6,92e-07	sim
ClanAPSO-3x10-L	1000	8,05e+04 (1,55e+09)	30	3,05e-05	sim
ClanAPSO-6x5-G	1000	2,41e+04 (1,38e+08)	30	8,75e-06	sim
ClanAPSO-6x5-L	1000	5,42e+03 (1,70e+07)	30	8,68e-04	sim
ClanPSO-10x3-G	1000	1,35e+11 (1,39e+18)	30	1,04e-05	sim
ClanPSO-10x3-L	1000	1,35e+11 (1,38e+18)	30	3,21e-05	sim
ClanPSO-3x10-G	1000	1,35e+11 (2,11e+18)	30	6,30e-08	sim
ClanPSO-3x10-L	1000	1,35e+11 (1,70e+18)	30	3,36e-08	sim
ClanPSO-6x5-G	1000	1,35e+11 (2,57e+18)	30	9,79e-08	sim
ClanPSO-6x5-L	1000	1,35e+11 (2,17e+18)	30	4,74e-07	sim
Constricted PSO-G	1000	1,37e+11 (1,58e+18)	30	7,45e-08	sim
Constricted PSO-L	1000	1,37e+11 (1,58e+18)	30	7,45e-08	sim
CPSOS	1000	-	0	-	-
FSS	1000	1,38e+11 (3,27e+18)	30	1,25e-08	sim
GCPSO-G	1000	1,38e+11 (1,51e+18)	30	6,98e-07	sim
GCPSO-L	1000	1,38e+11 (1,51e+18)	30	6,98e-07	sim
MCPSOCOL	1000	-	0	-	-
PSO-G	1000	1,36e+11 (2,29e+18)	30	2,43e-06	sim
PSO-L	1000	1,36e+11 (2,29e+18)	30	2,43e-06	sim

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo