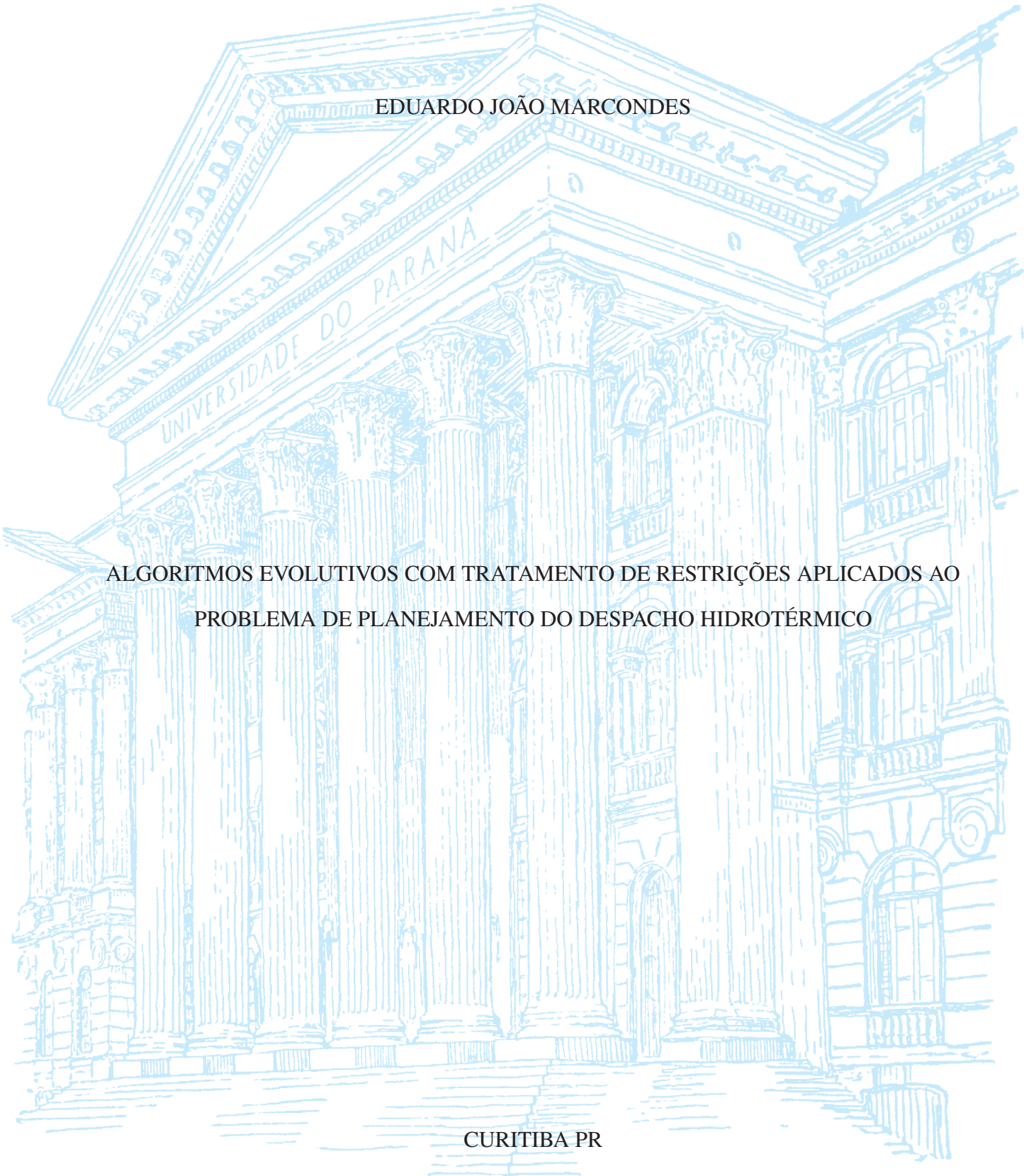
UNIVERSIDADE FEDERAL DO PARANÁ

EDUARDO JOÃO MARCONDES




# ALGORITMOS EVOLUTIVOS COM TRATAMENTO DE RESTRIÇÕES APLICADOS AO PROBLEMA DE PLANEJAMENTO DO DESPACHO HIDROTÉRMICO

CURITIBA PR

2019

EDUARDO JOÃO MARCONDES

# ALGORITMOS EVOLUTIVOS COM TRATAMENTO DE RESTRIÇÕES APLICADOS AO PROBLEMA DE PLANEJAMENTO DO DESPACHO HIDROTÉRMICO

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Informática no Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Ciência da Computação*.

Orientador: Prof. Dr. Daniel Weingaertner.

CURITIBA PR

2019

Catalogação na Fonte: Sistema de Bibliotecas, UFPR
Biblioteca de Ciência e Tecnologia

M321a Marcondes, Eduardo João
Algoritmos evolutivos com tratamento de restrições aplicados ao problema de planejamento do despacho hidrotérmico [recurso eletrônico] / Eduardo João Marcondes. – Curitiba, 2019.

Dissertação - Universidade Federal do Paraná, Setor de Ciências Exatas, Programa de Pós-Graduação em Informática, 2019.

Orientador: Daniel Weingaertner .

1. Algoritmos de computador. 2. Despacho hidrotérmico. 3. Restrições (inteligência artificial). I. Universidade Federal do Paraná. II. Weingaertner, Daniel. III. Título.

CDD: 005.1

Bibliotecário: Elias Barbosa da Silva CRB-9/1894

MINISTÉRIO DA EDUCAÇÃO
SETOR SETOR DE CIENCIAS EXATAS
UNIVERSIDADE FEDERAL DO PARANÁ
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO INFORMÁTICA - 40001016034P5

# TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em INFORMÁTICA da Universidade Federal do Paraná foram convocados para realizar a arguição da Dissertação de Mestrado de **EDUARDO JOÃO MARCONDES** intitulada: **ALGORITMOS EVOLUTIVOS COM TRATAMENTO DE RESTRIÇÕES APLICADOS AO PROBLEMA DE PLANEJAMENTO DO DESPACHO HIDROTÉRMICO**, após terem inquirido o aluno e realizado a avaliação do trabalho, são de parecer pela sua APROVAÇÃO no rito de defesa.

A outorga do título de mestre está sujeita à homologação pelo colegiado, ao atendimento de todas as indicações e correções solicitadas pela banca e ao pleno atendimento das demandas regimentais do Programa de Pós-Graduação.

Curitiba, 28 de Fevereiro de 2019.

DANIEL WEINGAERTNER
Presidente da Banca Examinadora (UFPR)

LUIZ EDUARDO SOARES DE OLIVEIRA
Avaliador Interno (UFPR)

KLAUS DE GEUS
Avaliador Externo (UFPR)

MINISTÉRIO DA EDUCAÇÃO - UNIVERSIDADE FEDERAL DO PARANÁ - Setor de Ciências Exatas - Coordenação de Pós-Graduação em Informática

*A Vanessa, meu amor.*
*Ao Antonio (in memoriam), meu amigão.*

## AGRADECIMENTOS

Agradeço a Vanessa Daniel, minha esposa e companheira, que participou comigo de todos os momentos desta jornada. Sem seu incentivo, encorajamento e inspiração, não poderia ter concluído este trabalho. Agradeço aos familiares e amigos, todos fizeram parte deste trabalho de algum modo.

Agradeço ao meu orientador professor Daniel Weingaertner, por compartilhar seu conhecimento e experiência, perseverando nessa orientação diante das dificuldades do arriscado projeto trabalho/mestrado. Agradeço aos professores que se disponibilizaram a vir para Foz do Iguaçu lecionar as disciplinas. Agradeço aos colegas do PPGInf Bruno Zanette e Peter Perroni, que auxiliaram na área do planejamento do despacho e de metaheurísticas, e remotamente me incluíram na comunidade acadêmica.

Agradeço também a ITAIPU Binacional, ao Parque Tecnológico Itaipu – PTI e a Universidade Federal do Paraná – UFPR, pela parceria na disponibilização deste mestrado em Foz do Iguaçu. Agradeço aos colegas da área de Desenvolvimento de Sistemas da ITAIPU, e aos gerentes João Baptista e Margarita Balbuena, pelo apoio e compreensão neste período de mestrado.

Este trabalho é resultado de conhecimento acumulado, não só da minha carreira acadêmica e profissional, mas também do conhecimento de vários profissionais, aplicados nas diversas ferramentas e técnicas utilizadas.

## RESUMO

Este trabalho apresenta o processo de pesquisa, desenvolvimento, aplicação e análise de algoritmos evolutivos com tratamento de restrições sobre o problema do planejamento do despacho hidroelétrico. Previamente à aplicação dos algoritmos sobre o problema de planejamento do despacho hidroelétrico, os algoritmos EPSO-G ($\epsilon$-*Constrained Particle Swarm Optiomization with Gradient Mutation*) e LSHADE44 (*Linear Success-History based Adaptive Differential Evolution* – quatro configurações de mutação e quatro posições de memória) foram implementados e avaliados sobre o *benchmark* de problemas com restrições CEC-2017 (Wu et al., 2016). Para aplicação no problema de planejamento do despacho hidroelétrico foi necessária sua modelagem na estrutura dos algoritmos, assim como a construção de classes de interfaces entre os algoritmos e o módulo de simulação do despacho hidroelétrico. Além do tratamento de restrições por $\epsilon$-constrained presente nos algoritmos, foi utilizado o tratamento de restrições por reparação implementado no simulador de despacho. As execuções foram realizadas sobre 194 diferentes séries de vazões, com dois cenários de estados iniciais de reservatórios: 70% e 30% da capacidade total. Os resultados apontaram que ambos algoritmos conseguiram convergir de soluções aleatórias para soluções comparáveis com outros trabalhos. O algoritmo LSHADE44 obteve os melhores resultados, com uma média de 3,5 restrições violadas entre as 13.431 modeladas, na configuração de 70% de volume inicial, com uma média de deficit hidroenergético de 142.254,80 MW, apresentando nos melhores casos soluções que chegaram a atender a todas restrições. O EPSO-G com a mesma configuração apresentou uma média de 161.6 restrições violadas, porém com menor deficit hidroenergético (36.190,85 MW). As configurações com volume inicial de 30% apresentaram resultados similares. O experimento permite que sejam testados novos algoritmos sobre o problema de planejamento do despacho hidroelétrico utilizando o simulador de despacho para avaliações. Para trabalhos futuros são sugeridas melhorias como aplicação de algoritmos específicos para altas dimensões, e alteração na modelagem do problema para otimizações por bacias hidrográficas.



Palavras-chave: Planejamento do Despacho Hidrotérmico, PSO Particle Swarm Optimization, DE Differential Evolution, LSHADE44, Técnicas de Tratamento de Restrições, e-constrained

## ABSTRACT

This study presents the process of research, development, application, and analysis of evolutionary algorithms with constraint handling for the hydroelectric dispatch planning problem. Prior to the execution of the algorithms on the hydroelectric dispatch planning problem, the EPSO-G ($\epsilon$-*Constrained Particle Swarm Optimization with Gradient Mutation*) and LSHADE44 (*Linear Success-History based Adaptive Differential Evolution* with four memory settings and four memory locations (LSHADE44)) were implemented and evaluated on the benchmark of problems with restrictions CEC-2017 (Wu et al., 2016). To run the hydroelectric dispatch planning problem it was necessary to fit your structure in the algorithms, which included the development (or implementation) of interface classes between the algorithm itself and the hydroelectric dispatch simulator. In addition to $\epsilon$-constrained, a constraint handling by repair method of infeasible solutions mechanism was used in the dispatch simulator. Executions were performed on 194 different flow series and with two initial reservoir state scenarios: 30% and 70% of the total capacity. Results showed that both algorithms were able to converge from random solutions to solutions comparable with those in other studies. The LSHADE44 algorithm obtained better results, with 3.5 violated constraints within 13,431 models, on the 70% of initial capacity, with 142,254.80 MW of hydropower deficit, with all constraints satisfied in the best cases. The EPSO-G, in the same configuration, presented 161.6 violated constraints on average, but a lower hydropower deficit (36,190.85 MW). Both configurations, 30% and 70%, produced similar results. This experiment allows for new algorithms for the hydroelectric dispatch problem to be tested using the dispatch simulator for evaluation. For future work, the following improvements are suggested: an application of specific algorithms for high dimensions; problem remodeling for optimization in separate watersheds.



Keywords: Economic Dispatch Problem, PSO Particle Swarm Optimization, DE Differential Evolution, LSHADE44, Constraint Handling Techniques, e-constrained

## LISTA DE FIGURAS

## LISTA DE TABELAS

## LISTA DE ACRÔNIMOS

| | |
|---|---|
| Aneel | Agência Nacional de Energia Elétrica |
| CAL-SHADE | Adaptive Constraint Handling and Success History Differential Evolution |
| CEC | Congress on Evolutionary Computation |
| CMA | Covariance Matrix Adaptation Evolutionary |
| CoDE | Composite Differential Evolution |
| COP | Constrained Optimization Problems |
| DE | Differential Evolution |
| $\epsilon$DE | $\epsilon$ Constrained Differential Evolution |
| $\epsilon$DEag | $\epsilon$ Constrained Differential Evolution with an archive and gradient-based mutation |
| $\epsilon$DEg | $\epsilon$ Constrained Differential Evolution with gradient-based mutation |
| ED | Economic Dispatch |
| EPSO-G | $\epsilon$-constrained Particle Swarm Optiomization with Gradient Mutation |
| ESA | European Space Agency |
| FAPSO | Fuzzy Adaptive Particle Swarm Optimizatio |
| FES | function evaluations |
| FLOSS | Free/Libre and Open Source Software |
| GA | Genetic Algorithm |
| LS | Lambda Search |
| LSHADE44 | Linear Success History Differential Evolution |
| LTS | Long-term support |
| MIP | Mixed Integer Programming |
| MPSO | Mutation Particle Swarm Optimizatio |
| ONS | Operador Nacional do Sistema Elétrico |
| PCH | Pequena Central Hidrelétrica |
| PSO | Particle Swarm Optimization |
| SaDE | Self-adaptive Differential Evolution |
| SHADE | Success History Differential Evolution |
| SIN | Sistema Interligado Nacional |
| SQP | Sequential quadratic programming |
| UDE | Unified Differential Evolution |

## LISTA DE SÍMBOLOS

| | |
|---|---|
| CR | Taxa de crossover do algoritmo DE |
| C1 | Fator de influência do gBest no PSO |
| C2 | Fator de influência do pBest no PSO |
| D | Dimensões |
| $\beta$ | Valor de restrições utilizado no tratamento $\epsilon$-constrained |
| $\epsilon$ | Variável de folga aceitação de violação de restrições |
| ftol | Critério de saída por fitness |
| $\delta$ | Limite de probabilidade mínima na escolha de estratégia de mutação |
| F | fator de escala do algoritmo DE |
| $\gamma$ | Threshold para aplicação de mutação de alteração de direção |
| n0 | Coeficiente de controle na escolha da mutação no LSHADE44 |
| $\phi_1$ | Coeficiente de influência do gbest no EPSO-G |
| $\phi_2$ | Coeficiente de influência do pbest no EPSO-G |
| $\sigma$ | Limite inferior de probabilidade no algoritmo LSHADE44 |
| SR | Taxa de soluções viáveis apresentadas entre as 25 execuções |
| $\overline{v}$ | Solução mediana |
| $\overline{vio}$ | Violação média |
| xtol | Critério de saída do algoritmo por posição |

**SUMÁRIO**

# 1 INTRODUÇÃO

O planejamento do despacho hidroelétrico é um problema de grande importância econômica e ambiental, que consiste em determinar as ações operacionais mensais de um conjunto de usinas hidrelétricas, onde sua produção energética é complementada por usinas termoelétricas, que somadas devem atender à demanda energética sob sua responsabilidade. Este complemento energético produzido por outros tipos de usinas (térmicas, eólicas e outras) é definido neste trabalho como deficit hidroenergético, ou simplesmente deficit, e é um dos principais objetivos de minimização do estudo.

A complementação da geração por usinas termoelétricas possuiu um alto custo financeiro e ambiental. Em 2017 as usinas hidrelétricas foram responsáveis por 61% da produção de energia, enquanto as termelétricas foram responsáveis por 27%, e entre as termoelétricas aproximadamente 70% utilizaram combustíveis fósseis poluentes (Agência Nacional de Energia Elétrica, 2017). Levando em consideração o grande mercado energético brasileiro, uma pequena redução percentual nos custos de energia também poderia refletir em um grande benefício econômico para a população em geral.

Este trabalho pretende selecionar os melhores algoritmos entre algoritmos aplicados sobre *benchmark* reconhecido no meio acadêmico e aplicá-los sobre o problema do planejamento do despacho hidroelétrico. Para avaliação das soluções produzidas pelos algoritmos no problema de planejamento do despacho hidroelétrico é utilizado o simulador de despacho desenvolvido por Zanette (2017), que por sua vez, utiliza a modelagem do problema apresentada por Bessa et al. (2013). A modelagem proposta por Bessa et al. (2013) difere do modelo vigente do ONS principalmente pela característica de detalhar os reservatórios individuais das usinas. No método vigente os reservatórios dos subsistemas são agrupados em reservatórios equivalentes.

Com o objetivo de aplicação no problema de planejamento do despacho hidroelétrico, e partindo dos experimentos já realizados e conhecimentos adquiridos nos grupos de pesquisas da área, definiu-se desenvolver um estudo sobre tratamento de restrições em algoritmo evolutivos utilizando o algoritmo PSO–*Particle Swarm Optimization* que apresentou resultados promissores em vários problemas, além de possuir uma implementação simples. Outros algoritmos já foram aplicados sobre o problema de planejamento do despacho hidroelétrico, contudo, sem um estudo específico sobre o tratamento de restrições.

Para um estudo prévio das técnicas de tratamento, com objetivo de comparação com outros algoritmos, foi utilizado um *benchmark* específico para avaliação e comparação de algoritmos evolutivos em problemas com restrições. O *benchmark* escolhido foi o utilizado na competição de algoritmos evolutivos da conferência *IEEE Congress on Evolutionary Computation (CEC-2017)*. Este *benchmark* tem como principal objetivo avaliar a capacidade dos algoritmos de encontrar soluções viáveis, portanto o algoritmo que apresentar mais soluções viáveis, ou menor quantidade de restrições quebradas, é considerado o melhor entre os analisados. Em casos de empates em viabilidades são considerados os valores das funções objetivos.

Ao aplicar o algoritmo PSO com tratamento de restrições sobre o *benchmark* os resultados obtidos não foram bons quando comparados aos demais algoritmos (que possuem resultados disponíveis para comparação sobre o *benchmark*), apresentando uma quantidade muito maior de restrições quebradas. Considerando que os melhores colocados sobre o *benchmark* eram variantes do algoritmo DE – *Differential Evolution*, optou-se por implementar o algoritmo variante do DE melhor colocado, para em seguida aplicá-lo sobre o problema de planejamento do despacho hidroelétrico.

Ao aplicar o algoritmo com tratamento de restrições que apresentou os melhores resultados no *benchmark* sobre o problema de planejamento do despacho hidroelétrico, espera-se alcançar a geração de um plano de despacho que maximize a produção de energia hidrelétrica mantendo bons níveis de reservatórios. Neste trabalho definiu-se como bom nível o reservatório apresentar volume maior ou igual a 70% do seu volume máximo. Considerando analisar o critério de volume de reservatórios das soluções, foram definidos dois cenários de avaliação: manutenção e recuperação de reservatórios. No cenário de manutenção todos reservatórios são inicializados com bons níveis, definindo os volumes iniciais em 70% do volume máximo. No cenário de recuperação os reservatórios são inicializados com níveis baixos com apenas 30% do volume total. Em ambos cenários é definido a restrição de finalização do período simulado com reservatórios com bons níveis, ou seja, maiores ou iguais a 70% da capacidade total.

Considerando: a grande quantidade de restrições dos cenários de testes propostos sobre o problema do planejamento do despacho energético, a alta dimensionalidade das soluções para o problema, e a pequena quantidade de experimentos semelhantes encontrados na literatura, a questão levantada é se os algoritmos evolutivos selecionados conseguirão gerar soluções viáveis para o problema em tempo aceitável. Caso estas soluções viáveis sejam alcançadas, uma questão secundária é a convergência da função objetivo nas soluções, função com o objetivo de manter os reservatórios das usinas mais próximos da capacidade máxima, e, ao mesmo tempo, maximizar a produção de energia.

## 1.1 OBJETIVO

O objetivo do trabalho é encontrar soluções viáveis otimizadas para o problema do planejamento do despacho hidroelétrico do sistema elétrico brasileiro utilizando algoritmos evolutivos com tratamento de restrições.

Os objetivos específicos do trabalho são definidos como:

- Comparação de algoritmos evolutivos com tratamento de restrições utilizando como métrica (*benchmak*) problemas da competição CEC-2017 – *Special Session and Competition on Constrained Real-Parameter Optimization*.
- Aplicação dos algoritmos evolutivos selecionados com o *benchmark* CEC-2017 no problema do planejamento do despacho hidroelétrico, utilizando a modelagem computacional do problema proposta por Bessa et al. (2013).
- Uso dos algoritmos evolutivos para planejamento de cenários de manutenção e recuperação de volume dos reservatórios.

### 1.1.1 Contribuições do Trabalho

Entre as principais contribuições deste trabalho estão:

- Estudo teórico e levantamento bibliográfico sobre tratamento de restrições em algoritmos evolutivos com aplicação da técnica $\epsilon$-constrained.
- Implementações em código aberto dos algoritmos evolutivos LSHADE44 e EPSO-G e aplicação sobre o *benchmark* CEC-2017.
- Desenvolvimento de um sistema de otimização do despacho hidroelétrico de médio prazo, utilizando os algoritmos evolutivos LSHADE44 e EPSO-G integrados com o simulador do sistema elétrico.

### 1.1.2 Organização do Trabalho

A organização do trabalho apresenta a estrutura:

- Capítulo 2 – Fundamentação Teórica: Apresenta os principais fundamentos utilizados no desenvolvimento do experimento deste trabalho.
- Capítulo 3 – Revisão da Literatura: Apresenta trabalhos relacionados ao problema de planejamento do despacho hidroelétrico, tratamento de restrições, e algoritmos Differential Evolution e Particle Swarm Optimization.
- Capítulo 4 – LSHADE44 e EPSO-G aplicados ao problema de planejamento do despacho hidroelétrico: Apresenta o histórico de desenvolvimento do trabalho, principalmente o trabalho prático de implementação dos algoritmos e aplicação sobre os problemas.
- Capítulo 5 – Materiais e Métodos: Apresenta a metodologia utilizada nos testes executados no experimento desenvolvido e informações como ferramentas, parâmetros de execução, ambiente, entre outros.
- Capítulo 6 – Resultados e Discussão: Mostra os resultados finais comentados das execuções dos algoritmos sobre os problemas propostos, de acordo com a metodologia apresentada.
- Capítulo 7 – Conclusão: Apresenta as considerações finais após a finalização do trabalho, com propostas de melhorias e trabalhos futuros.

# 2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo tem o objetivo de apresentar os conceitos teóricos utilizados nesta dissertação, compreendendo os algoritmos e técnicas de tratamento de restrições utilizadas e conceitos fundamentais sobre os problemas a que foram aplicados: problema de planejamento do despacho hidroelétrico e *benchmark* CEC-2017.

## 2.1 ALGORITMO PSO - PARTICLE SWARM OPTIMIZATION

O algoritmo evolucionário PSO, assim como muitas outras meta-heurísticas é baseado no comportamento biológico. Seu funcionamento é inspirado no comportamento de bandos de pássaros, cardumes de peixes ou colônias de abelhas, em busca de alimento ou de um bom refúgio. Bandos de pássaros, por exemplo, sobrevoam "áreas de busca" sem um controle central do bando, ou seja, o conjunto de todos os pássaros define a direção e a velocidade do bando. Caso um pássaro do bando encontre uma boa fonte de alimento, esse conhecimento é compartilhado com os demais pássaros, que também irão se direcionar a essa fonte. Além desse conhecimento transmitido entre todos os pássaros, cada um possui em sua memória os locais onde encontrou bons pontos de alimentação. Assim, não apenas se direciona para o melhor ponto encontrado pelo bando, como também é influenciado pelo melhor ponto guardado em sua memória.

No algoritmo PSO elementos chamados partículas representam os pássaros. Os componentes principais de uma partícula são posição e velocidade, sendo que a posição representa uma solução do problema a ser otimizado. A posição geralmente é modelada em uma estrutura tipo vetor, onde cada posição deste vetor é uma variável da função a ser otimizada no problema modelado. O segundo componente da partícula, é a velocidade, que é diretamente influenciada pela memória da partícula e pela posição da partícula com melhor desempenho da população. A definição inicial dos valores de posição das partículas é aleatória, selecionando aleatoriamente os valores da posição dentro dos limites do problema. Um enxame (*swarm*) é uma população com n partículas, que durante a evolução se movimentam sendo influenciadas pela melhor posição visitada pelo grupo – gBest – e pela melhor posição visitada pela própria partícula – pBest.

Os valores guardados em pBest e gBest são combinados com a velocidade atual de cada partícula para obter uma nova velocidade, que resultará na nova posição da partícula. O processo de atualização de cada posição i do vetor de velocidade v, para cada partícula x, a cada iteração t, é dado pelas fórmulas:

$$v_{t+1}[i] = v_t[i] + p1 * C1 * (pBest[i] - x_t[i]) + p2 * C2 * (gBest[i] - x_t[i]) \tag{2.1}$$

$$x_{t+1}[i] = x_t[i] + v_{t+1}[i] \tag{2.2}$$

Na equação de cálculo de velocidade os coeficientes p1 e p2 são valores aleatórios entre 0 e 1, ajudando na diversidade da população. As variáveis C1 e C2 são denominadas coeficientes de aceleração e controlam a influência dos valores de pBest e gBest. O nível de influência de pBest representa o comportamento cognitivo do método, ou seja, o processo da partícula de obter e guardar conhecimento. Já a influência do gBest representa o aspecto social do PSO, a

influência do grupo sobre cada partícula. As novas velocidades obtidas são então somadas a cada respectiva posição do vetor de posições anteriores da partícula, obtendo assim a nova posição.

O processo iterativo é repetido até que um dos critérios de parada seja atendido: número de gerações, diferença de valores entre as partículas, estagnação ou outros critérios implementados. O resultado presente na partícula gBest na última iteração será o melhor valor encontrado do algoritmo para o problema tratado. O algoritmo 1 apresenta o pseudocódigo simplificado do PSO.

**Algoritmo 1** Sequências de passos do PSO

```
1: inicializa população;
2: repeat
3:     Avaliar todas partículas;
4:     Atualizar velocidades;
5:     Atualizar posições;
6:     Atualizar pBest e gBest;
7: until critério de parada
```

#### *2.1.0.1 Parâmetros e Definições Iniciais do Algoritmo PSO*

Assim como outras meta-heurísticas, existem vários parâmetros que devem ser definidos pelo desenvolvedor ao implementar o algoritmo sem que existam valores pré-determinados. A definição deve levar em conta o problema alvo da implementação e a experiência do desenvolvedor. A literatura fornece algumas sugestões de faixas de valores iniciais, como também pode-se buscar problemas parecidos e aplicar os mesmos valores para testes.

A inicialização das posições das partículas, como já mencionado, é aleatória dentro do espaço de busca do problema. Porém, a inicialização do vetor de velocidades é objeto de estudos e testes, gerando controvérsias. A técnica mais comum é inicializar as velocidades da mesma maneira que as posições, com valores aleatórios dentro dos limites mínimos e máximos do espaço de busca. Contudo, também podem ser inicializadas com valores iguais a zero ou com números aleatórios muito pequenos em relação ao espaço de busca. Neste trabalho adotou-se a inicialização com valores iguais a zero, baseando-se no estudo empírico de Engelbrecht (2012) que concluiu que a inicialização com valores iguais a zero gera menos fugas das partículas do espaço de busca (*roaming particles*) que os demais métodos.

Os coeficientes de aceleração têm valores práticos definidos menores ou iguais a dois (Talbi, 2009), sendo mais frequente o uso do valor dois (Du e Swamy, 2016). Talbi (2009) sugere o número de partículas entre 20 e 60, aplicando também uma variável de inércia (com valor entre 0.4 e 0.9) a ser multiplicada pela velocidade anterior, para controle de impacto desta sobre a nova velocidade. Quanto maior o valor da inércia as partículas tendem a buscar mais globalmente, e quanto menor, as buscas se tornam mais locais.

A versão mais simples do PSO utilizada neste trabalho é chamada SPSO (*Standard PSO*), versão que utiliza o coeficiente de inércia citado anteriormente. Essa versão foi desenvolvida por Shi e Eberhart (1998), a partir do PSO original desenvolvido por Kennedy e Eberhart (1995) (chamado na literatura também de OPSO – *Original PSO*).

## 2.2 ALGORITMO DE - DIFFERENTIAL EVOLUTION

Esta seção apresenta as definições do algoritmo DE - Differential Evolution, com base nas definições do algoritmo apresentadas nos livros de Du e Swamy (2016), Talbi (2009) e

Engelbrecht (2007). O algoritmo DE é um algoritmo elitista, que cria novas soluções em uma estratégia de reprodução com vários pais. Utiliza um mecanismo de comparação entre cada indivíduo e sua nova versão, gerada com o cruzamento de vários indivíduos. O novo indivíduo substitui o anterior, caso seu fitness seja melhor.

Ao contrário dos algoritmos elitistas tradicionais, o algoritmo DE sempre faz perturbações em cada geração, chamadas mutações, com diferentes escalas e com diferentes membros da população.

Para cada indivíduo da geração atual, é feita uma reprodução com outros três indivíduos da população selecionados randomicamente, portanto quatro pais irão gerar o novo indivíduo.

Uma diferença em relação ao algoritmo genético – GA é que sempre ocorre mutação, que no GA pode ocorrer ocasionalmente. Além disso, no DE a mutação ocorre antes do cruzamento, enquanto no GA caso ocorra mutação, esta é realizada sobre o novo indivíduo gerado após o cruzamento. A mutação consiste basicamente na aplicação de operadores matemáticos sobre o conjunto de indivíduos selecionados. A teoria original do DE, sugere criar um vetor completo de indivíduos, gerados por mutação, previamente a entrada da execução no laço de iterações de cruzamentos. Outra possibilidade é executar as mutações dentro do laço de iterações com os grupos de indivíduos, sendo selecionados e sofrendo mutação a cada iteração. Porém, essa mudança não gera resultados diferentes: de qualquer maneira os indivíduos serão selecionados aleatoriamente e os operadores serão aplicados.

A estratégia de mutação considerada padrão do DE é chamada DE/rand/1/bin (Talbi, 2009) e (Ahlers et al., 2018). O autor Talbi (2009) específica que a notação DE/rand/1/bin segue o padrão DE/x/y/z, onde x especifica o vetor ao qual será aplicada a mutação, y o número de vetores usados, e z o esquema de mutação. Na seção que descreve o algoritmo LSHADE44, será explicado o funcionamento de quatro estratégias de mutações incluindo o rand/1/bin. A mutação rand/1/bin é representada na equação 2.3:

$$v_i^t = x_j^t + F(x_k^t - x_l^t), onde j \neq k \neq i \neq l \tag{2.3}$$

Duas estratégias de mutação são mais frequentemente apresentadas na literatura, a binomial e a exponencial. A diferença básica entre as duas é dada por Engelbrecht (2007), que explica que no crossover binomial, nos indivíduos selecionados, os pontos de crossover são selecionados aleatoriamente, dentro do vetor do indivíduo, mediante a probabilidade definida. Já no exponencial, é selecionado um índice aleatório dentro do vetor do indivíduo, e os pontos que serão aplicados ao crossover são adjacentes a esse ponto selecionado, formando assim um array circular.

O Algoritmo 2 apresenta o pseudocódigo do algoritmo DE. O processo de mutação, que envolve três pais na estratégia clássica, ocorre na linha 4 do pseudocódigo. Após a mutação é realizado o crossover com o indivíduo corrente da iteração e o novo indivíduo é avaliado. Se o novo indivíduo gerado pelo crossover apresentar um fitness melhor que o original, ele é inserido na nova população; caso contrário, é mantido o original.

#### *2.2.0.1 Parâmetros e Definições Iniciais do DE*

Os principais parâmetros que devem ser definidos no DE tradicional são: tamanho da população, fator de escala F e taxa de crossover CR. O fator de escala F, apresentado na equação 2.3, define a influência da variação diferencial no momento da mutação. Engelbrecht (2007) explica que valores maiores de F estimulam a exploração maior do espaço de busca, porém podem ter dificuldade em encontrar o ponto ótimo. Já valores menores de F podem demorar muito para convergir para as regiões ótimas.

**Algoritmo 2** Sequência básica de passos do DE

```
1: inicializa populacao;
2: repeat
3:    for all x in populacao do
4:       Gera indivíduo mutante v_i
5:       Gera indivíduo u_i cruzando v_i e x_i
6:       if f(u_i) < f(x_i) then
7:          novaPopulacao.add(u_i)
8:       else
9:          novaPopulacao.add(x_i)
10:      end if
11:   end for
12:   populacao ← novaPopulacao
13: until atingir num. máximo de gerações
```

O parâmetro CR representa a taxa de crossover do indivíduo original com o indivíduo gerado pela mutação, especificando quantos elementos do indivíduo original serão afetados pelo crossover. Ao aumentar o valor de CR, é estimulada a diversidade da população, aumentando a exploração do espaço de busca. Porém, um valor menor de CR faz uma busca local mais apurada, facilitando atingir o ponto ótimo.

Não existe um tamanho pré-determinado para a população, que pode ser definida de acordo com as características do problema a que o DE é aplicado. O tamanho afeta principalmente a diversidade e o tempo de execução do algoritmo. Nos textos analisados valores geralmente são de tamanho entre 30 e 200 indivíduos, ou proporcionais à dimensão do problema aplicado.

## 2.3 TÉCNICAS DE TRATAMENTO DE RESTRIÇÕES

Pode ser observado que nas descrições anteriores dos algoritmos PSO e DE padrões não são apresentados mecanismos ou técnicas que orientem o que deve ser feito em casos em que os problemas possuam limites ou restrições. Michalewicz (1995), reforçando essa ideia, afirma em seu *survey* que, apesar dos estudos apresentarem várias técnicas para definição de operadores, métodos de seleção e representações, em geral, algoritmos de computação evolucionária, não fornecem guias ou orientações de como tratar soluções inviáveis. Pode-se notar que além da classe do algoritmos evolucionários em que está o DE, as demais classes de meta-heurísticas também apresentam a mesma característica de não possuir um tratamento padrão de restrições, incluindo nesse grupo a classe de algoritmos de enxames (*Swarm Inteligence*), classe que abriga o algoritmo PSO.

As restrições podem ser definidas em duas categorias: restrições de bordas e restrições por funções. A primeira categoria, que não é o foco deste trabalho, é caracterizada pelas bordas no espaço de busca das soluções (*Boundary Constraints*). No caso do PSO as partículas que violam este tipo de restrições (atravessam os limites de bordas) são chamadas *Roaming Particles*. A solução típica para as restrições de borda é a substituição do valor extrapolado pelo valor limite da borda. A desvantagem desta técnica é a possibilidade de diminuição da diversidade, nos casos em que várias soluções violem restrições de bordas, a população começa a se tornar cada vez mais homogênea.

Outras técnicas de tratamento de restrições de bordas, descritas nos artigos que tratam restrições de bordas especificamente em partículas do algoritmo PSO, também podem ser

aplicadas sem dificuldade no DE e outros algoritmos de otimização. Os artigos *Experimental Study on Boundary Constraints Handling in Particle Swarm Optimization* de Cheng et al. (2011) e *Handling boundary constraints for numerical optimization by particle swarm flying in periodic search space* de Zhang et al. (2004) apresentam uma categorização de tratamento de restrições de bordas:

- Clássica : Caso o valor de uma dimensão extrapole o limite ela é substituída pelo limite máximo ou mínimo extrapolado.
- Determinística : Ajusta o valor extrapolado com o valor da média entre o valor extrapolado e o valor anterior que seria substituído.
- Estocástica : O valor extrapolado é substituído por um valor aleatório entre o limite mínimo e máximo.

O segundo caso de violação de restrições consiste nas funções de restrições, que devem ser atendidas pela solução final apresentada pelo algoritmo. Neste caso, além da função a ser otimizada, o algoritmo deve atender a uma ou mais funções que podem ser de igualdade ou desigualdade. A função apresentada na expressão 2.4, definida em P N Suganthan (2017), ilustra os tipos de restrições de uma função para otimização.

$$
\begin{aligned}
C03: \ Min\ f(x) &= \sum_{i=1}^{D}\left(\sum_{j=1}^{i} z_j\right)^2 \quad z = x - o \\
g(x) &= \sum_{i=1}^{D}\left[z_i^2 - 5000cos(0.1\pi z_i) - 4000\right] \leq 0 \\
h(x) &= -\sum_{i=1}^{D} z_i sin(0.1\pi z_i) = 0 \\
x &\in [-100, 100]^D
\end{aligned}
\tag{2.4}
$$

onde:

$f(x)$ é a função objetivo de minimização.
$h(x)$ representa uma restrição de igualdade.
$g(x)$ representa uma função de desigualdade.
$x \in [-100, 100]^D$ representa o limite do espaço de busca de cada dimensão D (restrição de bordas).

A seguir são descritas as principais técnicas de tratamento de restrições aplicadas sobre algoritmos evolucionários, com suas definições obtidas principalmente a partir de *surveys* específicas do assunto, como *A Survey of Constraint Handling Techniques in Evolutionary Computation Methods* de Michalewicz (1995), *Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art* de Coello (2002) e *Particle Swarm Optimization for Single Objective Continuous Space Problems: A Review* de Bonyadi e Michalewicz (2017).

#### *2.3.0.1 Funções de Penalidade*

A aplicação de penalidades é aparentemente a técnica mais utilizada de tratamento de restrições, por ser simples de implementar e frequentemente eficaz. A técnica consiste em avaliar o resultado obtido da função de restrição, e caso viole as restrições, uma penalidade é aplicada sobre o fitness da solução avaliada. Em geral os algoritmos de otimização utilizam comparação *one-to-one* na evolução das suas soluções e, com a penalidade aplicada, a solução penalizada tende a ter menor chance de sucesso no processo evolutivo. A principal desvantagem desse método é a dificuldade de parametrização, ou seja, a definição dos valores da penalidade. A seguir serão apresentados alguns tipos de técnicas de aplicação de penalidades.

- **Penalidade estática**: Nesta penalidade o valor ou o coeficiente da penalidade é definido no início da execução do algoritmo, e segue constante até sua finalização. Existem várias implementações deste tipo de estratégia: Homaifar et al. (1994) apresenta uma estratégia multiníveis, onde a função é penalizada de acordo com a quantidade ou valor que foi violado. Já em Morales e Quezada (1998) apud Coello (2002) foi desenvolvida uma técnica que aplica a penalidade ao *fitness* utilizando o valor da função violada proporcional ao total de restrições (que poderiam ter sido violadas). O principal problema das penalidades estáticas é a definição dos parâmetros necessários para aplicação da penalidades.

- **Penalidade Dinâmica**: Coello (2002) define como penalidade dinâmica qualquer tratamento de penalidades que envolva o número da geração para o cálculo da penalidade, sendo que normalmente a penalidade é incrementada com o tempo ou as iterações. A função 2.5 apresenta um cálculo de penalidade dinâmica, definido por Talbi (2009).

$$f_p(x,t) = f(x) + \sum_{i=1}^{m} w_i(t) d_i^k \tag{2.5}$$

onde $w_i(t)$ é uma função que decrementa com aumento de t.

A principal dificuldade da penalidade dinâmica é encontrar uma boa função proporcional ao tempo, que não pode decrementar, nem tão rápido, que não tenha tempo suficiente para encontrar o valor ótimo, e nem tão devagar, para não tornar a busca excessivamente lenta.

- **Penalidade Adaptativa** : Esta estratégia de penalidade, diferente das demais considera informações sobre o processo de busca de soluções do algoritmo (Talbi, 2009). Os coeficientes da penalidade são alterados de acordo com os resultados obtidos pelo algoritmo. Caso sejam encontradas muitas soluções inviáveis a tendência é aumentar a penalização sobre as partículas, e no caso contrário, poucas soluções inviáveis, deve-se diminuir o valor da penalização.

Um exemplo de penalidade adaptativa é apresentado em Hu (2009), onde é implementada sobre um PSO híbrido uma função objetivo composta também pelas funções de restrições, todas com valores normalizados, tornando o problema sem restrições. Um dos coeficientes da função é a taxa de partículas inviáveis na população, portanto a função se adapta com a evolução da população.

- **Penalidade de Morte** : Pode até não ser considerada uma estratégia de penalidade, pois não altera o fitness da solução, apenas elimina a solução inválida da população, necessitando assim a definição de uma estratégia de reposição. A consequência negativa dessa técnica é restringir o espaço de busca apenas aos pontos viáveis, sem aproveitar "pedaços" de soluções inviáveis. Também pode ser um processo muito lento em problemas com muitas restrições, com razoáveis chances de estagnação na população inicial, que geralmente é aleatória e pode não gerar soluções viáveis.

#### 2.3.0.2 *Estratégias de Reparação*

A reparação de uma solução inviável consiste em alterar seus valores de modo a torná-la viável, ou seja, sem violação de restrições. A reparação de soluções é relativamente fácil em casos de problemas de otimização combinatória, como o problema do caixeiro viajante, ou o problema da mochila. Exemplificando com o problema da mochila, caso uma solução ultrapasse a restrição de peso máximo, um algoritmo simples consistiria em um laço que retiraria itens até que a solução fique com peso dentro do limite máximo.

Segundo Coello (2002), não existem técnicas padrões para reparação de solução. A reparação é uma técnica dependente do problema, portanto um tipo de heurística deve ser implementada para cada caso. É uma boa estratégia em problemas combinatórios, porém pode ser muito custosa em tempo para demais problemas. Como será visto no capítulo de desenvolvimento foi aplicada conjuntamente com outra estratégia no experimento deste trabalho.

#### 2.3.0.3 *Técnicas de Otimização Multiobjetivo*

Esta técnica de tratamento de restrições utiliza métodos de otimização multiobjetivos, onde o processo é basicamente utilizar as funções de restrições como funções a serem otimizadas, sendo assim, o algoritmo utilizado deve buscar minimizar as funções de restrições. A partir da transformação das restrições em funções a serem otimizadas, teoricamente pode ser possível utilizar qualquer método de otimização multiobjetivo, como por exemplo, o PSO Multiobjetivo, ou o Algoritmo Genético multiobjetivo.

#### 2.3.0.4 *Técnica $\epsilon$-Constrained*

A técnica proposta por Takahama e Sakai (2005a) recomenda uma priorização das restrições sobre o valor da função objetivo dos problemas. A técnica consiste basicamente em alterar o processo de comparar duas soluções (após a aplicação da velocidade no PSO, ou após o crossover no DE, como exemplos), considerando inicialmente o valor das restrições, onde a solução que viole menos restrições é considerada a melhor. Caso nenhuma das soluções comparadas violem restrições ou os valores violados sejam iguais, o valor da função objetivo (fitness) é utilizado na comparação. Considerando a função objetivo de minimização 2.6.

$$\begin{aligned} \text{Min} \quad & f(x) \\ \text{sujeito a} \quad & g_j(x) \leq 0, j = 1, ..., q \\ & h_j(x) = 0, j = q+1, ..., m \\ & l_i \leq x_i \leq u_i, i = 1, ..., n, \end{aligned} \tag{2.6}$$

onde $x = (x_1, x_2, ..., x_n)$ é um vetor de solução, $f(x)$ é a função objetivo, $g(x)$ são as $q$ funções de desigualdade e $h(x)$ são as $m - q$ funções de igualdades.

O valor das restrições violadas, representado neste trabalho como $\beta$, pode ser definido de duas maneiras: soma de todos os valores de restrições violadas (equação 2.7) ou o valor da restrição com maior violação (equação 2.8).

$$\beta(x) = \sum_j || \max\{0, g_j(x)\}||^p + \sum_j ||h_j(x)||^p \tag{2.7}$$

$$\beta(x) = \max\{\max_j\{0, g_j(x)\}, \max_j |h_j(x)|\} \tag{2.8}$$

onde p é um número positivo.

O mecanismo é complementado por um relaxamento com decremento linear que permite que soluções com restrições sejam comparadas pelo valor da função objetivo. O valor do relaxamento é definido por $\epsilon$ e decresce durante as iterações. A técnica é descrita em 2.9:

$$p_1(f_1,\beta_1) < p_2(f_2,\beta_2) \Leftrightarrow \begin{cases} f_1 < f_2, & \text{se } \beta_1,\beta_2 <= \epsilon \\ f_1 < f_2, & \text{se } \beta_1 = \beta_2 \\ \beta_1 < \beta_2, & \text{caso contrário} \end{cases} \tag{2.9}$$

A expressão 2.9 mostra a comparação entre as soluções $p_1$ e $p_2$, que possuem respectivamente os valores de funções objetivo $f_1$, $f_2$ e restrições $\beta_1$ e $\beta_2$. A solução 1 é considerada menor que a solução 2 caso ambas restrições sejam menores que $\epsilon$ e o fitness $f_1$ seja menor que $f_2$, ou caso os valores de restrições sejam iguais, e o fitness $f_1$ sejam menor que $f_2$. Caso contrário são comparados os valores de restrições, ou seja, $p_1$ é menor que $p_2$ caso a restrição $\beta_1$ seja melhor que $\beta_2$. É comum a utilização do valor de $\epsilon$ igual a zero, desde o início das iterações, para casos em que o tratamento de restrições é prioritário.

## 2.4 EPSO-G – $\epsilon$-CONSTRAINED PARTICLE SWARM OPTIMIZATION MUTATION GRADIENT

O algoritmo EPSO-G é uma das variantes do algoritmo PSO original. Ele é apresentado como algoritmo intermediário no processo de desenvolvimento do algoritmo híbrido EPSO-CMA (Bonyadi et al., 2013). Suas características o diferenciam do PSO original por apresentar tratamento de restrições $\epsilon$-constrained, decremento linear da inércia e aplicação de mutação por gradiente sobre a velocidade das partículas.

A aplicação do tratamento de restrições $\epsilon$-constrained sobre os algoritmos PSO em geral é aplicado nas comparações entre partículas ao definir as partículas pBest e gBest. O artigo de referência de Bonyadi et al. (2013) trata o valor de $\epsilon$ sempre com valor 0.

Outro mecanismo presente no EPSO-G é o decremento linear da inércia, que pretende estimular a exploração de um maior espaço de busca nas iterações iniciais, e com o decremento o estímulo é de refinar as soluções encontradas na fase final da execução.

O procedimento de mutação aplicado sobre a velocidade das partículas tem o objetivo de evitar a estagnação e evitar buscas em linha. Du e Swamy (2016) citam que quase todos os variantes do PSO buscam resolver o problema da estagnação ou mínimo local, um dos maiores problemas do PSO. Já a busca em linha, ocorre quando as partículas oscilam somente entre os valores de pBest e gBest, deixando de buscar nas demais direções (Bonyadi e Michalewicz, 2017). A mutação presente no EPSO-G altera o cálculo da velocidade utilizando a equação 2.10.

$$v_{t+1} = m(\omega v_i + \phi_1 r_1(p_t - x_t) + \phi_2 r_2(g_t - x_t), c, \gamma) \tag{2.10}$$

onde $\omega$ é o coeficiente de inércia, $r_1$ e $r_2$ são dois valores aleatórios entre [0,1], $\phi_1$ e $\phi_2$ são os coeficientes de influência. A função m é a função de mutação, que recebe como parâmetros: vetor de velocidade atualizado e as constantes c e $\gamma$ (definidas com valores baixos definidos no início da execução). A mutação é calculada utilizando a equação 2.11.

$$d' = m(d, c, \gamma) \tag{2.11}$$

onde d é o vetor de velocidade, c e $\gamma$ são as constantes passadas como parâmetros.

A função m é definida pela equação 2.12.

$$d' = d + N(0, \sigma) \tag{2.12}$$

onde N é a distribuição multivariada normal e $\sigma$ é o vetor de variâncias.

O valor de $\sigma$ é definido utilizando a expressão 2.13.

$$\sigma_{\forall 1<j<D} = \begin{cases} c * ||N(0, \vec{\gamma})||, & \text{se } ||\vec{d}|| = 0 \\ c * ||rand(\gamma) * d||, & \text{se } 0 < ||\vec{d}|| < \gamma \\ c * ||\vec{d}||, & \text{outros casos} \end{cases} \tag{2.13}$$

onde ||.|| é a norma do vetor, c é uma constante, $\gamma$ é uma constante real com valor pequeno, $\vec{\gamma}$ é um vetor com todas posições preenchidas por $\gamma$, N é a distribuição normal e rand($\gamma$) gera um valor aleatório entre 0 e $\gamma$.

Na expressão 2.13 é possível observar o comportamento esperado da mutação. No primeiro caso, a norma da velocidade é igual a zero, ou seja, não haverá movimentação da partícula. Neste caso é gerado um vetor aleatório entre 0 e $\gamma$ para o vetor de velocidade, tirando a partícula da estática. No segundo caso a velocidade é muito pequena, então o vetor é "esticado"

multiplicando seu valor por um número aleatório. Nos demais casos, onde a velocidade não é nula nem muito baixa, a mutação multiplica a velocidade pela constante c[1].

## 2.5 LSHADE44 – LINEAR SUCCESS HISTORY ADAPTIVE DIFFERENTIAL EVOLUTION

O algoritmo LSHADE44[2] é uma variante do algoritmo DE com uma série de melhorias desenvolvidas. Apresentado por Polakova (2017) obteve o melhor resultado na competição sobre problemas com restrições na conferência CEC-2017. O algoritmo faz parte do grupo de variantes do DE tipo SHADE (*Success History Adaptive Differential Evolution*). A seguir serão descritos suas principais características e mecanismos de funcionamento do algoritmo.

**Redução linear da população**: Aplica uma equação de redução da população a partir da definição do tamanho da população inicial e final. A função é formulada para que a redução seja proporcional ao número de avaliações definido na execução do algoritmo. Sua intenção é aumentar a diversidade no início da busca e tornar a busca mais específica no fim da execução.

***Success History Adaptive***: Apresenta um conceito de memória e adaptação dos parâmetros F e CR do algoritmo (mesmos parâmetros F e CR do algoritmo DE original). O LSHADE44 trabalha com quatro pares de memória circular de H posições para cada parâmetro (F e CR). Para cada iteração executada sobre os indivíduos da população, um dos quatro pares de memória é sorteado. A posição da memória sorteada é selecionada por um contador que é incrementado em cada vez que é utilizado (retornando ao início após atingir o fim do vetor, daí o conceito de circular). Os resultados das variáveis F e CR aplicadas a indivíduos que forem bem sucedidos nos cruzamentos são armazenados em listas que são posteriormente utilizados em um cálculo de média para atualização dos valores de memória de F e CR. O método original foi proposto por Tanabe e Fukunaga (2013) como uma tentativa de minimizar a dependência da escolha de parâmetros para alcançar um bom desempenho do algoritmo.

**Estratégias de mutação**: O algoritmo trabalha com quatro estratégias diferentes de mutação, que são sorteadas utilizando uma probabilidade dinâmica: um vetor com quatro posições, cada posição representando uma estratégia de mutação, é iniciado com valor igual a 0. Cada vez que uma estratégia é selecionada e o indivíduo por ela produzido é bem sucedido, um contador incrementa a posição do vetor referente a essa estratégia. Os valores dos vetores são utilizados proporcionalmente para definir a probabilidade de sorteio das estratégias. Considerando a probabilidade inicial de cada mutação $l$ ser selecionada ($q_l = 1/4$), as probabilidades seguintes serão atualizadas pela equação 2.14.

$$q_l = \frac{n_l + n_0}{\sum_{k=1}^{4}(n_k + n_0)} \tag{2.14}$$

onde $n_l$ é o contador de sucessos da estratégia nos cruzamentos e $n_0$ é uma constante que previne mudanças drásticas das probabilidades.

Um limiar $\delta$ define um limite inferior para a probabilidade, caso seja atingido este limite os valores do vetor são reiniciados (as chances de cada estratégia serem selecionadas são definidas em 1/4).

As estratégias de mutações usadas no LSHADE44 são duas variações das estratégias current-to-pbest/1/bin e current-to-pbest/1/exp: randrl/1/bin e randrl/1/exp. As variações bin

[1]Em Bonyadi et al. (2013) o valor da constante é definido com valor igual a $1/D^2$, onde D é a dimensão do problema. Os testes foram realizados com dez e trinta dimensões, ou seja, c=0,01 e c=0,001.

[2]O label 44 do algoritmo é dado pela combinação das quatro estratégias de mutação, cada uma das estratégias com quatro pares de vetores de memória para os parâmetros F e CR

e exp foram apresentadas na descrição do algoritmo DE. A mutação current-to-pbest segue a equação 2.15.

$$u_s = x_s + F(x_p best - x_s) + F(r_1 - r_2), \tag{2.15}$$

onde:

$x_s$ é o indivíduo atual da iteração s.

$x_p best$ é um indivíduo escolhido do grupo dos p% melhores indivíduos da população (grupo *best*).

$r_1$ é um indivíduo escolhido aleatoriamente da população.

$r_2$ é escolhido entre a população P unida com o grupo A.

As estratégias randrl/1/bin e randrl/1/exp são versões da mutação clássica rand/1/bin e randr/1/exp. A mutação randrl/1 é representada na equação 2.16.

$$u_s = r_1 + F(r_2 - r_3) \tag{2.16}$$

Pode ser observado que a equação 2.16 é a mesma da mutação rand/1 original (equação 2.3), porém a seleção dos indivíduos é alterada. São selecionados três indivíduos aleatoriamente da população, colocando o melhor entre os três na posição $r_1$. Os dois restantes na posição $r_2$ e $r_3$. Polakova (2017) afirma em seu artigo que a alteração acelera em até 30% o processo de busca sem perda da qualidade.

**Grupo A**: Trata-se de um repositório de indivíduos substituídos entre as gerações. No início da execução o repositório está vazio, porém cada solução substituída é armazenada neste grupo, até o limite $maxsizeA$. Caso o limite seja atingido, um indivíduo aleatório do repositório é substituído. A utilização do grupo A nas mutações estimula a manutenção da diversidade durante a evolução da população.

**Tratamento de Restrições**: Os autores não especificam diretamente a técnica de restrição utilizada no LSHADE44, porém, pelas equações presentes no artigo, a técnica utilizada é a $\epsilon$-constrained com valor de $\epsilon$ igual a 0. Sempre que uma comparação é feita entre dois indivíduos esta técnica é utilizada.

## 2.6 PROBLEMA DE PLANEJAMENTO DO DESPACHO HIDROELÉTRICO

A definição de Despacho utilizada neste trabalho é a apresentada pela Agência Nacional de Energia Elétrica (2008): despacho é a definição de quais usinas devem gerar energia e quais devem ficar em estado de reserva (sem geração de energia), de modo que o volume de produção seja permanentemente igual ao consumo. Para definir se a usina irá gerar energia no período considera-se, neste trabalho, a quantidade de água liberada pelas turbinas (turbinamento). A outra variável controlada é a quantidade de água liberada por vertedouros (vertimento). Basicamente estas duas variáveis são definidas para cada usina em cada período, com seus valores definindo a produção de energia, limitada a restrições como demanda e capacidade de transmissão, afetando também os volumes dos demais reservatórios e usinas a jusante. Analisando apenas superficialmente é possível perceber a complexidade do problema, no qual devem ser coordenadas a geração e distribuição de energia de dezenas de usinas, mediante restrições como capacidade de produção, matéria-prima (reservatórios de água principalmente, mas também combustíveis e outras fontes como vento, biomassa, entre outras), além da capacidade de transmissão entre usinas e consumidores finais. Neste trabalho não é considerado o custo de geração individual

das usinas, assumindo que a energia hidrelétrica sempre tem menor custo que as demais fontes, buscando assim minimizar a necessidade destas.

Em trabalhos pesquisados relacionados sobre despacho energético, o foco de estudo dos trabalhos em geral foi diferente do utilizado neste trabalho, sendo o termo despacho geralmente associado ao despacho econômico - ED, onde o objetivo é o planejamento de produção de uma usina. Neste contexto o objetivo é minimizar custos de produção com o planejando do funcionamento das unidades de geração da usina. São tratadas restrições como zona proibida de operação, limitadas por níveis de vibrações, rotações por segundo, temperatura de operação, entre outros, gerando descontinuidades no espaço de busca do algoritmo de otimização.

Com objetivo de atendimento de demanda energética e econômico, além de atender critérios definidos de riscos, a definição de planejamento de operação utilizado no sistema NEWAVE desenvolvido pelo Centro de Pesquisas de Energia Elétrica (Cepel) é definida: "O objetivo básico do planejamento da operação de um sistema hidroelétrico é calcular a política de operação que estima os valores da água armazenada nos reservatórios e permite determinar, a cada mês, metas de geração para cada usina do sistema que atendam à demanda e minimizem o valor esperado do custo de operação ao longo do período de planejamento e atendendo um critério de aversão ao risco"(Cepel, 2018, n.p.).

No Brasil, o órgão responsável pelo planejamento de despacho hidroelétrico é o Operador Nacional do Sistema Elétrico (ONS). O ONS mantém o controle sobre o Sistema Interligado Nacional (SIN), que é composto por milhares de quilômetros de linhas transmissoras de energia entre diferentes regiões do país, ligando usinas e subestações. A figura 2.1, disponibilizada em ONS (2017), apresenta a rede completa de distribuição de energia do sistema brasileiro em 2017, sobre a qual o ONS realiza o planejamento do despacho.

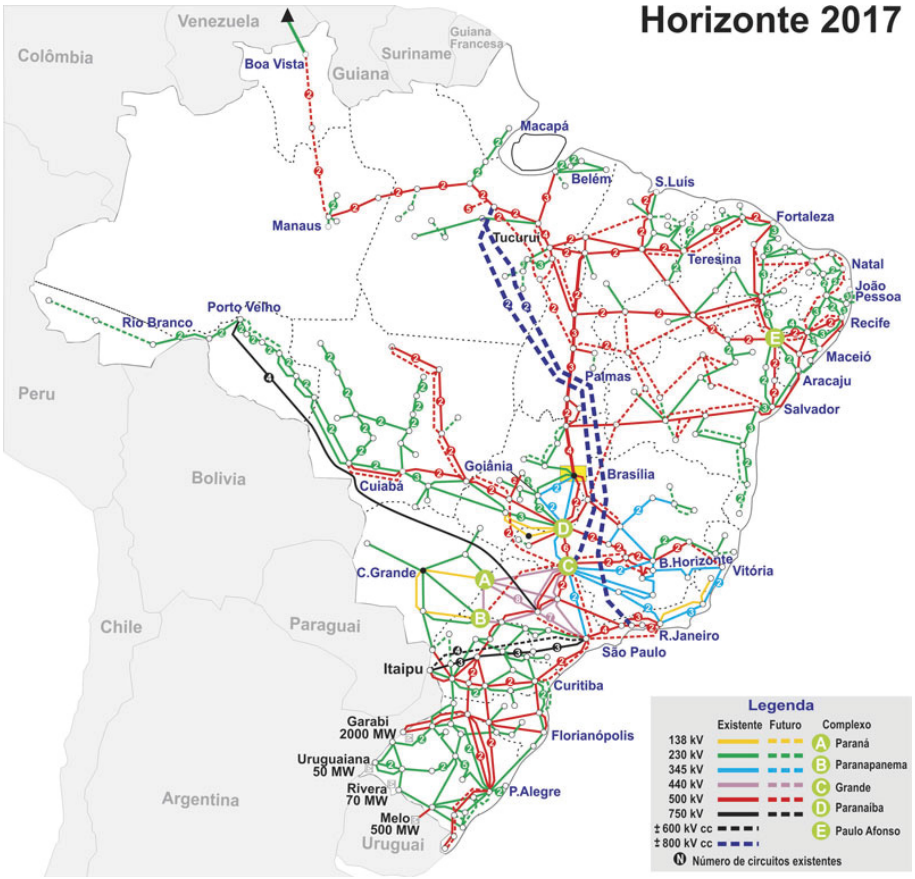



Figura 2.1: Sistemas de Transmissão Brasileiro em 2017 - SIN

O planejamento de despacho realizado pelo ONS usa diferentes modelos de planejamento com diferentes níveis de detalhamento, denominados de longo, médio e curto prazo, além da programação diária de operação – ONS (2017). O foco específico deste trabalho é o horizonte de médio prazo, onde são definidas as ações operacionais mensais em um horizonte de cinco anos.

### 2.6.1 Simulação e Otimização do Despacho

Neste trabalho, como apresentando inicialmente, é utilizado um simulador de despacho hidrelétrico que recebe como entrada um plano de despacho, executa este plano sobre as modelagens de vazões, demandas e capacidades configuradas, e retorna um valor de pontuação para o plano submetido (fitness da solução) juntamente com valores de restrições. O simulador de despacho utilizado foi desenvolvido por Zanette (2017), onde também é utilizado sobre algoritmos de otimização. Nesta seção são descritos os conceitos básicos necessários para compreensão e execução do simulador.

O simulador executa uma simulação em um período selecionado da operação de um grupo de usinas hidrelétricas, apresentando o total de energia gerada, quantidades de água nos reservatórios e vazões. A produção de energia nas usinas é controlada pela quantidade de água turbinada, que afeta por consequência a quantidade de água (ou nível) dos reservatórios. Além de utilizar a água para produção de energia também é possível verter a água para controle dos níveis dos reservatórios. A água que alimenta o volume dos reservatórios é classificada como vazão, e indica o fluxo que alimenta o reservatório. Além dos índices de vazões dos rios, o simulador também conta com dados estatísticos que incluem nas vazões os eventos meteorológicos que também afetam os níveis de reservatórios.

É importante observar que as usinas hidrelétricas são em grande maioria projetadas e construídas de maneira sequencial sobre os rios, formando reservatórios que são dependentes das quantidades de água (vertidas ou turbinadas) das usinas que ficam acima desta (em direção a nascente do rio, ou montante). O termo jusante define a direção do fluxo contrário a nascente.

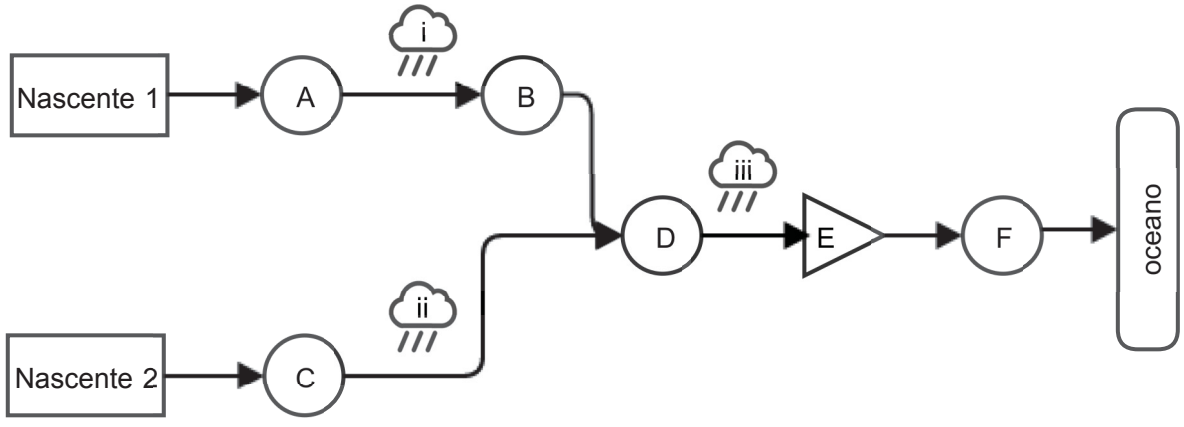



Figura 2.2: Grafo ilustrativo de sequência de usinas compondo uma bacia hidrográfica. As usinas estão nomeadas de A a F. Usinas com reservatórios estão circuladas e fio d'água em triângulos. As vazões naturais meteorológicas são identificadas de (i) a (iii).

A figura 2.2 apresenta um grafo representando uma sequência de usinas de A a F. O fluxo de água se inicia nas nascentes e é finalizado no oceano. Um exemplo da dependência entre usinas pode ser visto na água que alimenta a usina D, proveniente das usinas B e C, somados a vazão natural (ii) presente entre C e D. A usina E representa uma usina fio d'água, ou seja, não

possui reservatório, sendo que toda a vazão recebida da usina D somada a vazão natural (iii) é repassada para F (como turbinamento ou vertimento).

A figura 2.3 apresenta uma arranjo real de usinas sobre partes das bacias hidrográficas Paranapanema e Iguaçu. Estão presentes as usinas sobre os rios Iguaçu (destaque em amarelo), Paraná (destaque em vermelho) e Paranapanema (destaque em azul).

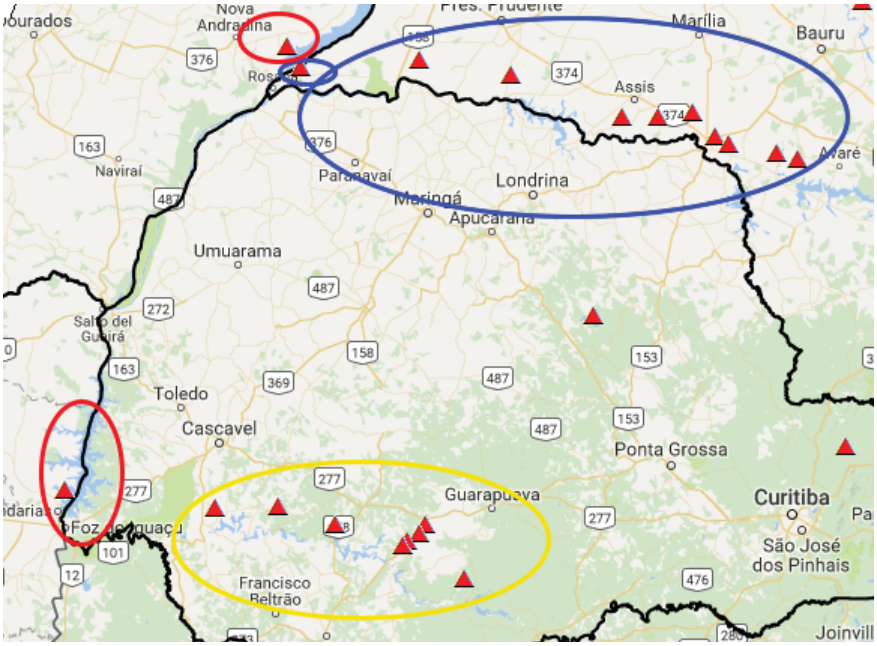



Figura 2.3: Exemplos de Reservatórios da Bacia Paraná, destacando as dependências entre os reservatórios.

Neste experimento o simulador de despacho hidroelétrico é utilizado como função de avaliação, que retorna valor de custo (fitness), e valores de restrições para o conjunto de variáveis enviadas para a avaliação. Dois objetivos principais foram definidos: minimização do deficit de energia e maximização do volume. Neste trabalho o termo deficit de energia se refere a quantidade de energia faltante para atender a demanda elétrica do sistema avaliado considerando utilizar apenas o conjunto de usinas hidrelétricas da simulação. O planejamento do despacho hidroelétrico envolve outras variáveis, como capacidades de produção de energia por usinas termoelétricas e custos de operação, componentes não são considerados neste trabalho.

A função de fitness do simulador inclui os dois objetivos citados anteriormente, a minimização do deficit de energia e maximização do volume. O deficit de energia utiliza as restrições de demandas e energia gerada por subsistemas, e o cálculo do volume verifica qual o deficit em relação a capacidade máxima dos volumes dos reservatórios. O simulador pode retornar dois tipos de fitness, fitness real e fitness penalizado. O fitness penalizado é utilizado para tratamento de restrições por penalidade, e não é utilizado neste trabalho.

A equação de cálculo do fitness real, definida a partir do algoritmo do cálculo de fitness de Zanette (2017), é apresentada na equação 2.17, onde v é o volume resultante da simulação da solução avaliada sobre o sistema elétrico, s é o subsistema ou usina avaliada, p é o período de avaliação, vMax e vMin são os volumes máximos e mínimos da usina s avaliada.

$$fitness = \sum_{s=0}^{u} \sum_{p=0}^{59} fitVolume_{s,p} + fitEnergia_{s,p} \tag{2.17}$$

$$fitVolume_{s,p} = \frac{1,0 - v_{s,p}}{vMax_s - vMin_s} \tag{2.18}$$

$$fitEnergia_{s,p} = abs\left(1 - abs\left(\frac{EnergiaGerada_{s,p}}{demanda_p}\right)\right) \tag{2.19}$$

Além do fitness o simulador retorna o estado dos reservatórios mês a mês, executando o plano proposto. Cada valor de reservatório é uma restrição, e seus limites são as capacidades máximas e mínimas. Não se trata de uma restrição de borda, pois seu cálculo depende de uma série de variáveis dependentes umas das outras, não é uma simples variável da função objetivo.

As restrições que limitam o espaço de busca do problema, ou restrições de bordas, são os limites mínimos e máximos de vertimento e turbinamento, onde o vertimento é a água que é liberada dos reservatórios sem geração de energia e o turbinamento é o fluxo de água através das turbinas produzindo energia. As restrições de volumes mínimos e máximos são funções de restrições, compostas por várias variáveis e são tratadas pelas técnicas de tratamento de restrições estudadas neste trabalho. Outras funções de restrições aplicadas sobre o planejamento do despacho são o volume final mínimo, que define um valor mínimo percentual relativo a capacidade total de armazenamento do reservatório, e o fluxo mínimo por usina, valor definido pelo ONS, que estabelece um valor mínimo de água que deve ser liberado por cada usina, seja por vertimento ou turbinamento.

### 2.6.2 Tratamento de Restrições do Simulador

Além de executar a simulação e avaliação da solução recebida, o simulador também possui a capacidade de correção da solução enviada para avaliação, comportando-se como uma técnica de tratamento de restrições por reparação.

O processo de correção, descrito em Zanette (2017), aplica-se somente nas restrições de volume mínimo de cada usina em cada período. Portanto caso sejam modeladas mais restrições como neste trabalho, ou violadas restrições de volume máximo, elas não serão reparadas pelo simulador. A correção da violação consiste em reduzir a quantidade de água utilizada pela usina de maneira que o volume fique não viole o limite mínimo. O corretor reduz inicialmente a quantidade de água vertida, que não contribui para o atendimento da demanda de energia. Caso o volume continue abaixo do limite mínimo, é reduzida na sequência a quantidade de água turbinada, buscando assim corrigir a restrição de volume mínimo.

# 3 REVISÃO DA LITERATURA

Neste capítulo serão apresentados trabalhos relacionados aos algoritmos PSO, DE e seus variantes, aplicados à problemas com restrições. Também serão apresentados trabalhos relacionados à otimização do planejamento do despacho hidroelétrico. Na última seção, serão expostos de forma concisa os trabalhos aos que os algoritmos foram comparados utilizando o *benchmark* de avaliação.

## 3.1 ALGORITMO PSO APLICADO A PROBLEMAS DE OTIMIZAÇÃO COM RESTRIÇÕES

Pretende-se aqui apresentar uma revisão de trabalhos que aplicaram o tratamento de restrições no algoritmo PSO e suas variantes. Alguns desses trabalhos são aplicados em áreas fins, onde foi utilizado o PSO e era necessário tratar as restrições para atender o problema. Já outros trabalhos são de pesquisa específica sobre o tratamento de restrições no PSO. Neste caso geralmente são utilizados *benchmarks* para comparação com outros algoritmos evolucionários ou outras técnicas de tratamento de restrições. Em todos os casos buscou-se apresentar a descrição do problema, as técnicas utilizadas e os resultados obtidos.

Dentro do levantamento bibliográfico desta pesquisa, um dos trabalhos mais antigos dedicado ao tratamento de restrições no algoritmo PSO, foi o artigo de Coath e Halgamuge (2003), no qual foi desenvolvida uma comparação entre tratamentos de restrições aplicados sobre o algoritmo PSO. Os dois tratamentos de restrições comparados são a aplicação de penalidades e a eliminação de soluções inviáveis (também conhecida como penalidade de morte). A técnica de penalidade é chamada de multiestágio, onde os valores das penalidades são definidos em valores estáticos, no início da execução, e de acordo com o valor da restrição violada, um nível de penalidade é somado ao valor de fitness. Os algoritmos foram testados sobre cinco diferentes funções de otimização, sendo que em duas os resultados foram melhores e mais rápidos no tratamento por penalidades. Em outras duas, os resultados foram idênticos, tanto em valores, quanto tempo. E em uma das funções, o tratamento por funções viáveis não resolveu o problema, não conseguindo sair do processo de inicialização, por não encontrar soluções viáveis de maneira aleatória.

O artigo de Biao et al. (2014) aborda o problema de operação ótima de usinas térmicas, um problema com algumas características em comum com o problema de planejamento do despacho hidroelétrico, porém, com a intenção principal de otimizar o lucro dos operadores. O problema consiste em determinar as unidades que serão acionadas, buscando minimizar o custo deste acionamento mediante restrições como: balanço de carga, saída de energia, *ramp rate*, entre outros. A principal técnica utilizada foi o algoritmo MPSO - *Mutation Particle Swarm Optimization*, uma variante do PSO, onde é aplicado um operador de mutação sobre as partículas que fornece maior diversidade e melhor convergência ao resultado ótimo. Também é aplicado um filtro automático, que seleciona partículas para participar da otimização global, tornando o processo de otimização mais rápido. O MPSO foi comparado com vários algoritmos diferentes, com PSO-LR (*PSO Lagrangian Relaxation*), *improved PSO*, algoritmo Mimético, MILP, entre outros, totalizando 13 algoritmos de otimização, e obteve melhores resultados finais que todos os demais. As restrições foram tratadas por uma técnica de penalidade dinâmica, que multiplica o fitness pelo número de restrições violadas.

Em Moreno e Kaviski (2013), os autores aplicaram uma versão do PSO modificada, chamada M-PSO (*Modified-PSO*), no problema de planejamento de despacho de pequenas

hidrelétricas. O M-PSO, inspirado na versão UPSO (*Unified-PSO*) de Parsopoulos e Vrahatis (2004), que utiliza o conceito de local best no cálculo da velocidade, a melhor partícula da n-vizinhança. Já o M-PSO adiciona o coeficiente inércia, que é decrementado por uma função dependente da iteração e do número de partículas. As restrições são basicamente de volume de reservatório e balanço de água, que são tratadas por técnica de penalidade dinâmica. Os resultados foram considerados promissores, com rápida convergência e evitaram-se ótimos locais.

O artigo de Takahama e Sakai (2005b) demonstra a aplicação da técnica $\epsilon$-constrained no algoritmo PSO, com o objetivo de resolver problemas com restrições, nomeando a técnica de $\epsilon$PSO. A técnica apresentada tem como características melhorar o resultado da função objetivo das partículas que atendem as restrições e melhorar o atendimento das restrições das funções que não as atendem. No experimento o valor de $\epsilon$ é iniciado com a média da violação de restrição da primeira geração e é decrementado até zero quando o algoritmo atinge 80% das iterações. Para os testes foram utilizadas três funções não lineares com restrições de igualdade de desigualdade. O $\epsilon$PSO teve seus resultados comparados com o algoritmo GENOCOP 5.0 (Koziel e Michalewicz, 1999), um variante de algoritmo genético para problemas com restrições. Em todas as funções testadas o $\epsilon$PSO obteve melhores resultados que o GENOCOP5.0, com melhores resultados na função objetivo, e menor violação de restrições, além de apresentar resultados em tempo aproximadamente dez vezes inferior.

No ano seguinte, os autores Takahama e Sakai (2006b) acrescentaram ao algoritmo $\epsilon$PSO a funcionalidade de controle adaptativo de velocidade, pois afirmaram que em alguns testes a velocidade das partículas acaba se tornando muito alta, fazendo com que se dirijam para longe de regiões viáveis. A velocidade é controlada dividindo as partículas em N grupos, que são classificados conforme o número de partículas viáveis pertencentes a estes, sendo o melhor grupo o que contém entre sua população a maior quantidade de partículas viáveis, e o pior, o que contém mais partículas inviáveis. O controle é feito limitando a velocidade do pior grupo à velocidade do melhor grupo. Os testes foram realizados sobre três funções de *benchmarks*, comparando com o algoritmo $\epsilon$PSO original, e mais algoritmos genéticos com diferentes tratamentos de restrições. Em todos os casos o algoritmo com controle de velocidade obteve melhores resultados, em um menor tempo. O controle de velocidade apresentou melhores resultados principalmente quando o espaço de busca viável é pequeno, evitando que as partículas saltem para regiões distantes das regiões viáveis.

O artigo de Bonyadi et al. (2013) apresenta um algoritmo híbrido, PSO e CMA-ES (*Covariance Matrix Adaptation Evolutionary Strategy*) com tratamento de restrições $\epsilon$-constrained. O algoritmo EPSO-G, um algoritmo intermediário no desenvolvimento do EPSO-CMA, foi utilizado neste trabalho. O algoritmo EPSO-G é explicado com detalhes na fundamentação teórica no capítulo 2, e complementando o desenvolvimento do algoritmo foi aplicado o algoritmo CMA-ES, para busca local, refinando os resultados alcançados pelo algoritmo EPSO-G. O algoritmo EPSO-G também é evoluído em uma estratégia *multi swarm*, onde vários enxames são iniciados e evoluídos isoladamente, e a melhor partícula de todos os grupos é considerada a solução gBest final. A técnica reserva um percentual maior das iterações para o PSO e o restante para o CMA como uma estratégia de refinamento ou busca local a partir do melhor resultado. A técnica híbrida foi comparada em 19 funções de *benchmarks* com os algoritmos $\epsilon$DEag (Takahama e Sakai, 2010) e Co-CLPSO (Liang et al., 2010). O EPSO-CMA alcançou 100% de soluções viáveis em todas as funções testadas, enquanto o $\epsilon$DEag, atingiu apenas 12% em uma das 18 funções. E o Co-CLPSO, não conseguiu resultados viáveis em três funções, e em outras duas, conseguiu parcialmente. As funções foram executadas em 10 e 30 dimensões, e em termos de valor de função objetivo, o algoritmo tipo DE obteve melhores resultados que os

demais para 10 dimensões, e o algoritmo PSO-CMA, obteve melhores resultados na configuração com 30 dimensões. Em termos médios das duas dimensões o EPSO-CMA foi superior.

O algoritmo Co-CLPSO (Liang et al., 2010) citado anteriormente, utiliza dois enxames de partículas para tratar o problema, onde partículas podem ser designadas para explorar diferentes restrições. Em seus testes executados com oito variantes diferentes do PSO, os resultados obtidos foram melhores ou iguais aos seus concorrentes.

No *review* de Bonyadi e Michalewicz (2017) são apresentados os principais estudos em PSO recentes. A seção 5 da *review*, *Modification of PSO to Deal with Constrained Optimization Problems (COP)*, apresenta uma revisão das principais técnicas de tratamento de restrições aplicadas no PSO desde sua criação. O trabalho apresenta técnicas simples como: penalidade, preservação de soluções viáveis e proximidade de região viável. A revisão também apresenta versões modificadas do PSO, que possuem técnicas mais avançadas de controles de restrições, entre elas, o algoritmo MLPSO, que foi parcialmente utilizado neste trabalho. Os autores concluem que ainda não foram feitos estudos e testes suficientes para o tratamento de restrições no algoritmo PSO.

## 3.2 ALGORITMO DE APLICADO A PROBLEMAS DE OTIMIZAÇÃO COM RESTRIÇÕES

Nesta seção serão apresentados trabalhos que aplicaram o algoritmo DE e seus variantes à problemas de otimização com restrições, com o objetivo de expor as técnicas utilizadas e os resultados alcançados.

Uma aplicação simples do tratamento de restrições $\epsilon$-constrained sobre o algoritmo DE, chamada $\epsilon$ *Constrained Differential Evolution* ($\epsilon$DE), foi feita no trabalho de Takahama et al. (2006). O algoritmo foi comparado com os algoritmos $\epsilon$PSO e $\epsilon$GA, sobre os problemas *Himmelblau's problem*, *welded beam design problem* e *pressure vessel design problem*. Para todos os algoritmos a configuração foi do valor inicial de $\epsilon$ igual a zero. Nos três problemas o $\epsilon$DE alcançou melhores resultados que os demais algoritmos tradicionais, utilizando o mesmo tratamento de restrições. Além de alcançar valores melhores da função objetivo, ele precisou de um número menor de avaliações para alcançar os melhores valores, aproximadamente 1/5 a 1/2 da quantidade de avaliações dos demais algoritmos.

Parte do grupo de autores que publicou o artigo anterior, Takahama e Sakai (2006a), publicou no mesmo ano uma versão melhorada do $\epsilon$DE, incluindo mais dois mecanismos para tratamento de restrições, mutação gradiente e preservação de elite. A mutação gradiente busca por meios matemáticos converter a função de restrição não linear em linear, de maneira que seja possível definir as variáveis que estão causando violações de restrições e repará-las, porém o método pode consumir muitas avaliações de funções (FES), e ainda assim não conseguir reparar a solução. A terceira técnica aplicada é a preservação de elites, onde os N indivíduos com menos restrições violadas em cada iteração são armazenados em um repositório. Esse repositório é utilizado como se fosse a população no momento de fazer as mutações e crossovers do DE. Caso surjam indivíduos melhores que os presentes no grupo de elite, estes vão sendo substituídos. O algoritmo foi aplicado sobre um *benchmark* com 24 funções de restrições, e obteve sucesso em resolver 23.

Os mesmos autores do $\epsilon$DE com mutação gradiente e elites, Takahama e Sakai (2010), publicaram uma nova versão do algoritmo chamada $\epsilon$DEag, onde é retirado a funcionalidade de preservação de elite e inserido um mecanismo parecido, denominado arquivo A. O arquivo tem como função de manter a diversidade na população. O arquivo é criado inicialmente com indivíduos aleatórios, de onde serão retirados os indivíduos da primeira iteração do DE. Durante a iteração do DE os indivíduos do arquivo são utilizados na mutação DE/rand/1/exp (equação 3.1)

onde o terceiro indivíduo $x_l^t$ será selecionado do arquivo A. Caso o indivíduo resultante deste cruzamento seja melhor que seu antecessor, o antecessor é armazenado no arquivo, substituindo o indivíduo sorteado para a mutação.

$$v_i^t = x_j^t + F(x_k^t - x_l^t), onde j \neq k \neq i \neq l \tag{3.1}$$

#### *3.2.0.1 Algoritmos CEC 2017*

Também como algoritmos tipo DE relacionados estão os algoritmos finalistas da competição CEC-2017 de resolução de problemas com restrições por algoritmos evolutivos. Os quatro algoritmos que são descritos a seguir não são apenas relacionados, mas também fazem parte deste trabalho como comparativos, além do algoritmo LSHADE44 de Polakova (2017) ter sido implementado como um dos principais algoritmos do experimento.

#### *3.2.0.2 CAL-SHADE*

Implementado por Zamuda (2017), CAL-SHADE é o acrônimo de *Constraint Handling With Success History Adaptive Differential Evolution*. É um algoritmo tipo SHADE com redução linear de população (L-SHADE) e tratamento de restrições $\epsilon$-constrained. O artigo apresentado não especifica como é a estratégia SHADE, mas em geral os algoritmos tipo SHADE fazem o ajuste em tempo de execução das variáveis F e CR, com uma ou mais posições de memória para armazenar o histórico de sucesso das configurações utilizadas. No trabalho o autor descreve $\epsilon$ se inicia com um valor baixo, proporcional ao valor violado nas restrições, e após 500 iterações ele alcança o valor zero, caindo linearmente. O algoritmo obteve o quarto lugar entre os quatro finalistas.

#### *3.2.0.3 LSHADE44*

O algoritmo implementado por Polakova (2017) é descrito na fundamentação teórica, pois foi objeto de estudo deste trabalho. O algoritmo obteve o primeiro lugar entre os finalistas da competição, porém apresentou tendência de queda de desempenho com dimensões maiores (mesmo assim obteve o primeiro lugar considerando a média de todos resultados).

#### *3.2.0.4 LSHADE44 + IDE*

O algoritmo implementado por Tvrdik e Polakova (2017) é uma técnica híbrida entre o algoritmo LSHADE44 e o algoritmo IDE. O algoritmo IDE, acrônimo de *Individual-DEpendent Mechanism*, também é um tipo de algoritmo tipo DE. Nesta técnica a qualidade das soluções dentro da população influencia os fatores F e CR (F é o coeficiente que define a influência da variação diferencial no momento da mutação e CR afeta a probabilidade de utilizar elementos do vetor mutante no indivíduo), tornando os indivíduos melhores menos influenciados por mutações ou crossovers com o restante da população, e com maior possibilidade de influência nos cruzamentos.

Os dois algoritmos atuam de maneira isolada na evolução. O algoritmo LSHADE44 age, primeiramente, com um objetivo maior de obter soluções viáveis. Ele é executado até 99% das FEs disponíveis, ou até que sejam encontrados n indivíduos viáveis, onde n é a população mínima do algoritmo (ele conta com redução linear de max a min).

A implementação do IDE não utiliza o $\epsilon$-constrained, apenas é verificado se o valor da restrição é menor que um threshold, onde nessa implementação foi definido como a violação média inicial (após a execução do LSHADE44).

#### *3.2.0.5 UDE*

O quarto algoritmo da competição foi desenvolvido e implementado por Trivedi et al. (2017) para a competição. Trata-se também de um algoritmo tipo DE, onde UDE significa *Unified Differential Evolution*. O termo *Unified* se refere a unificação das principais funcionalidades das seguintes técnicas:

- CoDE: *Composite Differential Evolution* - sua principal característica é combinar várias estratégias de mutação e seus parâmetros para formação do *trial vector*. Nessa implementação são utilizadas três estratégias de mutação, juntamente com um *pool* de parâmetros F e CR. A cada iteração são gerados três *trials vectors*, com parâmetros escolhidos aleatoriamente do *pool*. O melhor entre os três é comparado com o vetor de origem e para ingresso na próxima geração.
- JADE: A principal característica desta variante é não selecionar o melhor da população para as estratégia de mutação tipo best, mas selecionar randomicamente de um grupo de bons indivíduos.
- SaDE: *Self-adaptive DE* - as estratégias de geração de vetores *trial* e os valores de parâmetros do algoritmo são auto adaptados através da experiência das gerações anterioes.
- *Ranking-based mutation operator*: O algoritmo DE original seleciona aleatoriamente entre a população os pais para mutação, já esta técnica dá aos indivíduos com mais qualidade melhores chances de seleção.

  O algoritmo utiliza a penalidade estática, de acordo com a equação 3.2.0.5.

$$F(x_i) = f(x_i) + penalty * cv(x_i)$$

Segundo os autores o algoritmo foi inicialmente desenvolvido para problemas sem restrição, apenas foi incluído a penalidade para este tratamento. Entre os quatro algoritmos comparados este ficou na segunda posição entre os finalistas.

## 3.3 TRABALHOS RELACIONADOS À OTIMIZAÇÃO DO PLANEJAMENTO DO DESPACHO HIDROELÉTRICO

Nesta seção são apresentados trabalhos na área de otimização energética, com foco em trabalhos sobre despacho hidroelétrico e algoritmos evolucionários.

A dissertação de Zanette (2017) é o trabalho com a mesma área fim estudada aqui, que é a otimização do planejamento do despacho elétrico, ou seja, o planejamento da produção de cada usina em um conjunto de usinas, mediante um grande número de restrições (nível de reservatórios, capacidade de produção, consumo e limites de transmissão). Para a otimização foi utilizado um buscador local baseado no método de Monte Carlo e um algoritmo otimizador inspirado nos métodos *Simulated Annealing* e Busca por Subida de Encosta.

O tratamento de restrição utilizado por Zanette (2017) foi aplicação de penalidade. O autor não especifica exatamente qual modalidade de penalidade, mas analisando as funções pode-se concluir que é a penalidade adaptativa, pois o valor penalizado depende do valor obtido pela função objetivo da atual avaliação, elevado a um coeficiente (nos testes o coeficiente teve o valor 2 utilizado).

O resultado apresentado pelas técnicas aplicadas atingiu o objetivo definido no trabalho, sendo que o otimizador conseguiu resultados melhores que o plano original em todas as execuções, com diferentes parâmetros, atendendo as demandas energéticas. Uma observação do autor aponta para alguns casos onde os reservatórios tiveram como resultado final volume abaixo do mínimo, e que uma tentativa de correção seria aumentar a penalidade.

O trabalho de Bessa et al. (2013) apresenta os momentos iniciais da criação do modelo de despacho hidrotérmico PHOENIX utilizado por Zanette (2017) e também utilizado neste trabalho. No trabalho é explicado que o modelo atende a uma chamada da ANEEL para propor um novo modelo de planejamento de despacho energético. A diferença principal neste novo modelo é a individualização das usinas hidrelétricas na modelagem substituindo a modelagem por reservatórios equivalentes. No artigo é feito um experimento utilizando o método lagrangeano aumentado com tratamento de restrições por otimização multiobjetivo. Também é utilizada a meta-heurística subida de encosta para refinamento de resultados, e o autor aponta ser promissora a aplicação do algoritmo PSO sobre o problema.

Os artigos de Moreno e Kaviski (2013) e Biao et al. (2014) foram citados anteriormente com foco no algoritmo PSO, agora são apresentados analisando sua modelagem em relação ao problema energético. O artigo de Biao et al. (2014) apresentou o problema de planejamento ótimo diário de usinas térmicas, do ponto de vista de valores no mercado energético, ou seja, minimizar o custo total de produção do conjunto de termoelétricas, também atendendo à demanda e as demais restrições de capacidades, transmissões etc. Como já citado foi utilizado o MPSO (Mutation PSO) e implementado um MIP (*Mixed Integer Programming*) para comparação. Os algoritmos foram aplicados sobre dados históricos de um dia de operação de um conjunto de três usinas térmicas, duas com quatro unidades de geração e uma com cinco unidades, todas com funcionamento a carvão, localizadas na China. Os resultados obtidos indicaram um aumento de 9.01% no lucro total do conjunto de usinas com a aplicação do algoritmo. O lucro médio por turbina teve uma elevação de 10.73% e o algoritmo de comparação MIP obteve um aumento de 2.73% no lucro, todos comparados aos dados históricos sem otimização.

O já citado artigo de Moreno e Kaviski (2013) apresenta o problema do planejamento diário de pequenas centrais hidrelétricas - PCHs, definidas pela Aneel como usinas com capacidade de menos de 30 mil kW. O objetivo neste caso é maximizar a produção de energia, mediante restrições como capacidade de produção, limite de vazão, limites do reservatório e balanço de água do reservatório (função entre capacidade, fluxo de entrada e saída sobre o tempo). Para

resolução do problema foi proposto o M-PSO, e para comparações foram utilizados o U-PSO, SPSO e *Simulated Annealing*. As técnicas foram aplicadas sobre duas PCHs brasileiras de 24 e 12 MW. Em uma programação de 5 dias o algoritmo proposto foi cerca de 0.03% superior ao U-PSO e 0.08% superior ao SPSO e *Simulated Annealing*. Não foram apresentados dados estatísticos reais das usinas para comparação.

Em Chang (2010) é apresentado um problema semelhante ao proposto neste trabalho, a otimização de planejamento de sistema hidrotérmico. É utilizado um PSO modificado chamado FAPSO (Fuzzy Adaptive PSO). O objetivo é a minimização do custo de operação do sistema, especificamente considerando o custo de produção das unidades térmicas. Devem ser atendidas restrições como produção igual à demanda, limites de produção das unidades, níveis de reservatórios, entre outros. A lógica *fuzzy* neste PSO é responsável por ajustar a variável de inércia no cálculo da velocidade das partículas. As restrições neste trabalho foram tratadas por penalidade adaptativa. Foram implementados um SPSO e um algoritmo genético (GA) para comparações com o FAPSO, e foram executados testes em uma configuração com quatro hidrelétricas e uma usina térmica, para programação do conjunto no período de 24 horas. O FAPSO foi o melhor conforme previsto devido ao ajuste dinâmico da inércia, com resultados 0,8% melhores (nos valores apresentados uma diferença aproximada de 7 mil dólares em um total de 920 mil dólares).

A dissertação de Oliveira (2015) estuda meta-heurísticas aplicadas ao problema do despacho econômico. São consideradas as usinas térmicas e o objetivo é diminuir o custo total de produção, mediante a seleção das unidades que serão acionadas no período simulado. O autor classifica como restrições clássicas o balanço de potência, limites operativos, e como restrições realísticas o rendimento dos combustíveis utilizados e as zonas de operação proibidas. Na otimização são aplicadas duas meta-heurísticas, o algoritmo do morcego e uma variação do PSO chamada Trelea-PSO. O Trelea-PSO simplifica a equação de velocidade, utilizando uma ponderação entre o pBest e o gBest, e um coeficiente que é a média dos dois coeficientes de aprendizagem. Observa-se que foram implementadas diferentes modelagens de partículas para diferentes restrições, por isso não foi necessário aplicar técnicas específicas de tratamento de restrições. Foram testados casos encontrados na literatura, com configurações de 10 a 100 termoelétricas, e a grande maioria dos casos testados o algoritmo do morcego obteve melhores resultados.

O relatório técnico *Advanced Algorithms for Hidropower Optimization*, de Harpman et al. (2012), não aprofunda em resultados de implementações, porém faz um levantamento geral de que problemas podem ser otimizados computacionalmente na área de energia hidrelétrica. O autor classifica os problemas em Despacho Econômico, como planejamento curto de operação de uma ou mais hidrelétricas; Despacho Econômico Dinâmico, planejamento por maiores períodos de um conjunto de hidrelétricas, como 24 horas ou 168 horas; e o Despacho de Unidade, onde é otimizada a operação das unidades de geração, dentro de uma usina hidrelétrica.

O relatório de Harpman et al. (2012) também apresenta os principais algoritmos de otimização utilizados, como GA, DE, PSO e LS (*Lambda Search*), todos com técnicas de inicialização, definição de parâmetros, estratégias de tratamentos de restrições, critérios de parada, avaliações, entre outros. No final do artigo foram feitos testes simples com os quatro algoritmos onde todos alcançaram os resultados ótimos, variando apenas em tempo de convergência.

Outro trabalho mais genérico é o livro Artificial *Intelligence in Power System Optimization* (Ongsakul e Vo, 2013). O livro descreve detalhadamente o problema do despacho econômico em muitas variações e detalhes, como perda em transmissões, restrições de combustíveis, de transmissão, considerando hidrelétricas. O planejamento de despacho hidrotérmico também é apresentado, mostrando tipos de planejamento (curtos ou longos), as principais restrições como

níveis, balanço de carga e vazões. Nos exemplos rápidos são usadas técnicas como *Enhanced Lagrangian Relaxation* para resolução. Nesse livro o PSO é apresentado como uma das soluções para a otimização de Fluxo de Energia - *Power Flow*, e também é usado no Despacho Ótimo de Potência Reativa (minimização da perda de potência entre dois pontos na transmissão).

## 3.4 CONCLUSÃO

Pode ser observado no levantamento bibliográfico que as datas de publicações dos trabalhos indicam um menor uso do algoritmo PSO e seus variantes nos últimos anos, com a ascensão dos algoritmos da família DE tendo maior aplicação. A competição CEC-2017 todos os algoritmos submetidos era do tipo DE. A técnica de tratamento de restrições por penalidades ainda é amplamente utilizada, porém a técnica $\epsilon$-constrained também vem sendo aplicada sobre vários trabalhos. As duas técnicas possuem vantagens e desvantagens e ainda não existe uma técnica definitiva ou com grande vantagem sobre as demais para utilização em algoritmos evolutivos.

Um exemplo da influência do tratamento de restrições pode ser observado no resultado apontado por Zanette (2017). Um problema apontado nos resultados finais é a queda excessiva dos níveis dos reservatórios, e o autor sugere como trabalhos futuros ajuste de parâmetros de penalidades para possível solução. Este problema é apontado na literatura como ponto fraco da técnica de penalidade: a dificuldade da definição dos valores dos parâmetros. Um valor pode até ser muito bom no começo da execução, porém fica inadequado na fase final. Em casos de penalidade dinâmica também existe a dificuldade de definir a taxa e a fórmula de incremento que se ajuste perfeitamente ao problema.

Os resultados apresentados pelos algoritmos de otimização para problemas com restrições participantes da competição CEC-2017 terão seus resultados apresentados com mais detalhes no capítulo de resultados preliminares, onde o algoritmo implementado nesta fase do trabalho será comparado com esse grupo de competidores.

# 4 LSHADE44 E EPSO-G APLICADOS AO PROBLEMA DE PLANEJAMENTO DO DESPACHO HIDROELÉTRICO

Neste capítulo são apresentados o processo de desenvolvimento dos algoritmos LSHADE44 e EPSO-G e a aplicação destes sobre os problemas selecionados: *benchmark* CEC-2017 e o problema do planejamento do despacho hidroelétrico. Serão apresentadas as estratégias de modelagem, arquitetura desenvolvida, implementação e execução dos algoritmos sobre os problemas propostos.

O desenvolvimento do trabalho foi dividido em dois blocos com objetivos bem definidos: (a) seleção e implementação de um bom algoritmo/técnica de tratamento de restrições e (b) aplicação deste algoritmo sobre o problema de planejamento do despacho hidroelétrico. Para constatar se o algoritmo implementado obteve de fato bons resultados sobre os problemas de otimização com restrições foi utilizado um *benchmark* que possui trabalhos publicados com resultados possíveis de serem comparados, com diferentes algoritmos.

Para a fase inicial de seleção de algoritmo foi escolhido o algoritmo PSO, por fazer parte do campo de estudos do projeto de pesquisa atuado, além de constar na literatura como uma tendência na área de algoritmos evolutivos. O algoritmo foi implementado com tratamento de restrições simples por penalidades e comparado com demais algoritmos considerados como "estado da arte", já aplicados sobre o mesmo *benchmark*.

Analisando os resultados da aplicação do PSO sobre o *benchmark* foi decidido implementar a segunda fase do trabalho – aplicação sobre o problema de planejamento do despacho hidroelétrico – com uma versão melhorada do PSO que possui a proposta de atuar sobre os pontos fracos apresentados, e também implementando o algoritmo LSHADE44 de Tvrdik e Polakova (2017), que apresentou o melhor resultado entre os algoritmos comparados sobre o *benchmark*. Os algoritmos selecionados foram implementados também sobre o *benchmark* para comprovação dos resultados, e em seguida aplicados sobre o problema de planejamento do despacho hidroelétrico.

Como apresentado no decorrer deste capítulo, apesar do algoritmo tipo PSO não ter apresentado bons resultados no *benchmark* (assunto discutido no capítulo 6), ele também foi aplicado sobre o problema de planejamento do despacho hidroelétrico com seus resultados usados como comparativos no problema. A implementação do algoritmo neste trabalho é de fácil aplicação no problema do despacho e seus resultados podem servir de base para trabalhos futuros com algoritmos ou técnicas similares.

## 4.1 ARQUITETURA DO EXPERIMENTO

O produto final composto pela implementação dos algoritmos e modelagem dos problemas, ainda que tenha sido construído utilizando linguagens e ferramentas distintas, pode ser visto como um *framework* de teste de algoritmos e problemas, onde podem ser incluídos novos algoritmos para testes sobre os problemas modelados, ou no sentido inverso, também podem ser aplicados outros algoritmos aos problemas modelados.

A intenção no projeto de construção do experimento foi de preservar a principal característica de meta-heurística em ambos algoritmos, de resolver problemas sem conhecimento prévio. Adicionalmente, a implementação isolada dos algoritmos permite sua aplicação em diferentes problemas evitando assim duplicação de código, facilitando o processo de desenvolvimento.

A figura 4.1 apresenta uma visão de alto nível da arquitetura do desenvolvimento do experimento, considerando a versão final com a aplicação dos algoritmos EPSO-G e LSHADE44 sobre o benchmark CEC-2017 e sobre o problema do planejamento do despacho hidroelétrico. Na figura são mostradas em verde as duas linguagens utilizadas no desenvolvimento dos algoritmos e modelagem dos problemas: C++ e Python 3.5 (as razões das escolhas serão apresentadas nas seções seguintes que descrevem cada um dos módulos). O uso da linguagem Python na implementação do experimento tornou necessária a adoção de ferramentas de integração como pybind11 (Jakob et al., 2018) e gRPC/Protobuf[1].

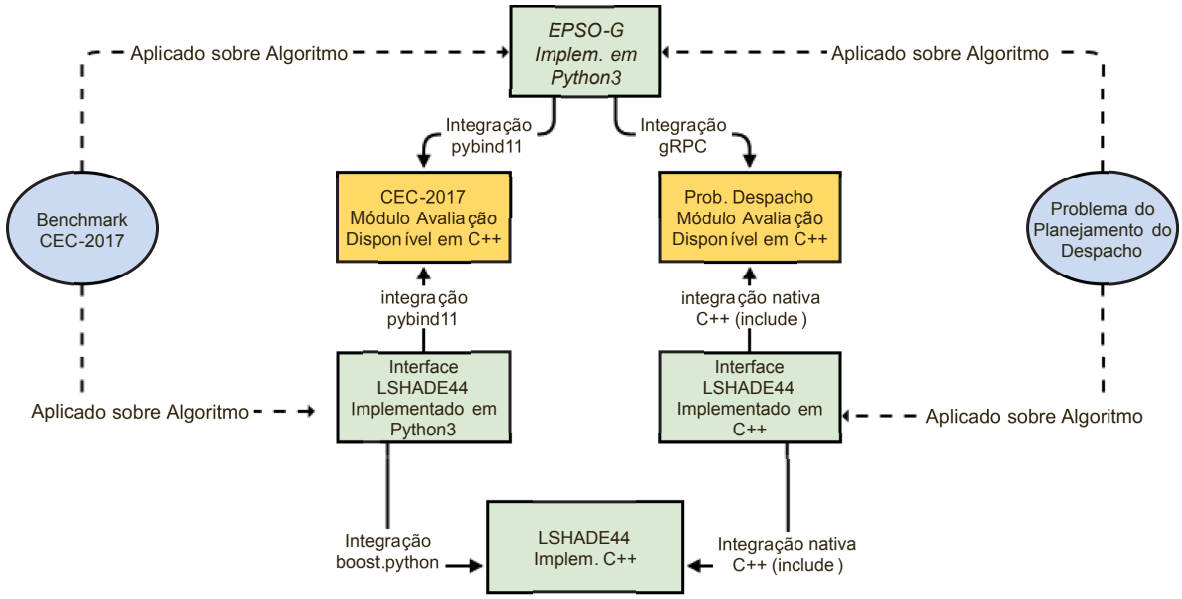



Figura 4.1: Arquitetura do experimento, apresentando nas elipses os problemas tratados, nos retângulos verdes os algoritmos implementados nas respectivas linguagens de programação e nos retângulos amarelos os módulos de avaliação disponíveis dos problemas. As linhas contínuas apresentam as integrações e ferramentas necessárias entre os módulos e as tracejadas indicam a linguagem de implementação dos problemas.

No gráfico de arquitetura também é possível observar a característica de isolamento e reaproveitamento dos algoritmos. Em caso da aplicação dos algoritmos em novos problemas em trabalhos futuros este framework já disponibilizaria as interfaces em C++ ou Python.

A seguir serão apresentados os detalhes de desenvolvimento individuais dos algoritmos implementados e problemas modelados, assim como detalhes das ferramentas de integração.

## 4.2 IMPLEMENTAÇÃO DO ALGORITMO LSHADE44

A base de implementação do LSHADE44 foi o algoritmo DE disponível no *framework* de otimização paralela Pagmo/Pygmo – (Biscani et al., 2019). O *framework* está disponível com licença FLOSS (*Free/Libre and Open Source Software*) e é usado e patrocinado pela Agência Espacial Europeia - ESA. O *framework* contém a versão básica do algoritmo DE apresentado na fundamentação teórica, na qual foram implementados progressivamente as funcionalidades descritas no algoritmo LSHADE44. O módulo do *framework* é representado como Core DE na figura 4.1 que representa a arquitetura do experimento.

O *framework* utilizado é composto por duas versões, Pagmo para C++ e Pygmo para Python. A intenção inicial foi utilizar somente a linguagem de programação Python, porém ao

[1]gRPC/Protocol Buffers - `https://grpc.io`

estudar detalhadamente a implementação da biblioteca verificou-se que o *framework* não contém de fato os algoritmos evolutivos implementados em Python, contendo apenas interfaces para bibliotecas C++ que podem ser invocadas pela linguagem Python, ou seja, para alteração dos algoritmos é necessário implementar em C++.

Além do algoritmo DE o *framework* contém uma série de algoritmos de otimização e problemas modelados. Algoritmos como PSO, Genético, Colônia de Abelhas, além de algoritmos de otimização local e hiperheurísticas estão disponíveis para uso e adaptação. A modelagem do *framework* é de certa maneira similar a este experimento, com possibilidade de trocar os problemas e algoritmos, porém de maneira muito mais completa e com outros objetivos.

Neste trabalho foi explorada a funcionalidade UDP (*User Defined Problem*) do *framework* Pagmo, onde utilizando uma interface padrão é possível modelar novos problemas, neste caso os problemas do planejamento do despacho hidroelétrico e o *benchmark* CEC-2017. Para modelar um novo problema sobre o *framework* é obrigatório a implementação dos métodos fitness e get_bounds. O método get_bounds retorna em vetores os limites mínimos e máximos de borda das variáveis do problema, e o método fitness retorna o valor de fitness concatenado com os valores de restrições, da solução recebida como parâmetro.

O algoritmo 3 apresenta o fluxo simplificado de execução do algoritmo LSHADE44 sobre o problema do despacho hidroelétrico, utilizando o *framework* Pagmo. A versão do problema LSHADE44 usando a funcionalidade UDP do *framework* Pagmo é instanciada na linha 2 do algoritmo, passando ao *framework* Pagmo informações do problema como número de dimensões, limites de bordas, funções de fitness e restrições. Os valores de cada série sintética de vazão são carregados para simulação na linha 4 do algoritmo, na mesma estrutura UDP. O algoritmo LSHADE44, que foi implementado neste trabalho sobre o algoritmo DE presente no framework Pagmo, é instanciado na linha 5, recebendo como parâmetro as variáveis necessárias. O processo de evolução da população, que otimiza as soluções iniciais geradas aleatoriamente, ocorre na linha 7 e, após seu término na linha 8 é selecionada a melhor solução como solução gerada pelo algoritmo.

**Algoritmo 3** Pseudocódigo de invocação do algoritmo LSHADE44 sobre o framework Pagmo

1: Inicializa: $nSeriesVazoes = 194$, $maxIter = 500000$, $ftol = xtol = e^{-6}$, $nPopMin = 4$, $nPopulation = 30$
2: $udp = ProblemaDoDespacho()$ ▷*instancia modelagem do problema do despacho*
3: **for** $nExec = 1$ to $nSeriesVazoes$ **do**
4: $udp.carregaSerie(nExec)$
5: $algoritmoDE = LSHADE44(maxIter, ftol, xtol, nPopMin)$
6: $populacao$ = inicializaPopulação($udp$,$nPopulation$)
7: $populacao = algoritmoDE$.evoluir($populacao$)
8: $melhorSolucao = populacao.best()$
9: **end for**

Na implementação do algoritmo LSHADE44 foi optado pela ampla utilização de estruturas simples como vetores. Por exemplo, a memória circular da variável CR é implementada como uma matriz de quatro por seis posições (quatro pares de memórias com seis posições cada), com variáveis controlando linha e coluna, correspondente a memória e posição. A cada utilização uma das memórias é selecionada aleatoriamente e em seguida a variável que controla sua posição é incrementada utilizando a estratégia *round-robin*.

A população é implementada no LSHADE44 da mesma maneira nos dois problemas executados, seguindo o padrão do *framework* de base. A população fica dentro da classe

*population* do *framework*, onde os principais atributos são *m_x* e *m_f*, estruturas dinâmicas tipo vector onde são adicionados os indivíduos da população e seus respectivos fitness. Além da estruturas *m_x* e *m_f* estão presentes métodos para inicialização aleatória da população, seleção de melhor solução entre a população, e demais métodos para definição e consulta dos atributos.

A seções seguintes apresentam exemplos da interface destes métodos com o simulador do sistema energético no caso do problema de despacho, e do módulo de fitness do *benchmark*.

## 4.3 IMPLEMENTAÇÃO DO ALGORITMO EPSO-G

O algoritmo PSO e seus variantes, incluindo o EPSO-G, foram implementados em Python, e diferentemente do LSHADE44 não usaram como base bibliotecas com versões básicas do algoritmos. Como os problemas têm seus módulos de avaliação desenvolvidos em C++ foram necessárias estratégias de interface entre estes, que serão especificadas em cada problema.

Uma versão simplificada do código implementado, mostrando um pseudocódigo do trecho principal de execução do algoritmo EPSO-G, é apresentada no algoritmo 4. Na linha 2 do algoritmo é instanciado o objeto que faz a conexão com o módulo avaliador do problema, neste caso do problema do planejamento do despacho. Este módulo avaliador é guardado no objeto população (linha 3) e é utilizado ao avaliar a população (linha 7).

O teste de critério de saída AvaliaCondicoesSaida() apresentado na linha 9 verifica se a diferença de fitness entre o melhor e pior indivíduo é menor que um limiar, retornando a condição para saída do laço principal de iterações. Esta função também avalia a distância de localização entre estas duas partículas, que é também é um critério de saída caso seja menor que um limiar. Para determinar a diferença entre a localização entre as duas partículas é utilizado um cálculo de distância euclidiana entre suas posições.

**Algoritmo 4** Pseudocódigo EPSO-G

EPSO-G[nIteracoes, nPopulacao]

```
1:  for nExec = 1 to nSeriesVazoes do
2:     CPPDesp ← CPPDesp(nExec)      ▷CPPDesp: módulo de avaliação do problema
3:     Pop.interface ← Pop(CPPDesp)
4:     Pop.populacao ← PopulacaoAleatoria(nPopulacao)
5:     iteracao ← 0
6:     repeat
7:        Pop.avaliacoes ← AvaliaPopulacao(Pop)
8:        Pop.melhores ← AtualizaMelhores(Pop)
9:        if AvaliaCondicoesSaida() then
10:          break      ▷finaliza execução da série por xtol ou ftol
11:       end if
12:       Pop.populacao ← AtualizaPosicoes(Pop)
13:       iteracao ← iteracao + 1
14:    until iteracao < nIteracoes
15: end for
```

A população implementada nos algoritmos tipo PSO, assim como no LSHADE44, tem o mesmo formato para ambos problemas tratados. No PSO ela foi definida apenas como um vetor contínuo de elementos tipo *double*, com seu tamanho definido pela dimensão do problema multiplicado pelo número de indivíduos. A velocidade fica em um vetor com o mesmo tamanho.

O processo de inicializar a população consiste em definir para cada posição do vetor um valor aleatório dentro dos limites de borda para a dimensão selecionada.

Como não foram previstos neste trabalho a implementação de mais problemas aplicados ao PSO, seu código não foi modularizado com interfaces para diferentes problemas. Os códigos fontes dos algoritmos implementados juntamente com as aplicações sobre os problemas estão disponíveis para download em Marcondes (2019).

## 4.4 IMPLEMENTAÇÕES DOS ALGORITMOS SOBRE O PROBLEMA DE PLANEJAMENTO DO DESPACHO HIDROELÉTRICO

Como apresentado na arquitetura do projeto, o problema de planejamento do despacho hidroelétrico foi modelado em duas linguagens: inicialmente em C++ por facilidade de integração com módulo avaliador, e em seguida replicado na linguagem Python para aplicação no algoritmo EPSO-G. Cada solução, chamada partícula nos algoritmos PSOs ou indivíduo nos algoritmos DEs, é representada na tabela 4.1, e nomeada Solução Percentual no texto:

Tabela 4.1: Solução modelada no problema de planejamento do despacho hidroelétrico para uso nos algoritmos. Representa uma solução em percentual da capacidade total de turbinamento e vertimento para 111 usinas com 60 períodos.

| Usina 1 | | | | | ... | Usina 111 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| %qt1 | %qv1 | ... | %qt60 | %qv60 | ... | %qt1 | %qv1 | ... | %qt60 | %qv60 |

As variáveis %qt e %qv representam os percentuais de vertimento e turbinamento em relação as capacidades máximas de cada usina. Como o valor de vertimento máximo não é definido nas usinas foi utilizado um valor alto padrão para todas usinas, e o percentual deve se ajustar para cada usina de acordo com a evolução da população. O conjunto total de 2 variáveis vezes 60 períodos vezes 111 usinas corresponde a uma solução com 13320 dimensões (partícula no PSO ou indivíduo no DE/LSHADE44).

Nas fases iniciais de implementação do experimento foram utilizados os valores reais de vertimento e turbinamento nas variáveis das soluções, porém, como os cruzamentos ou posições afetam o indivíduo como um todo, vertimentos e turbinamentos de usinas com escalas de valores muito diferentes eram "misturadas", gerando valores muito longe dos aceitáveis. O corretor que atua no simulador para reparação de soluções não consegue agir sobre valores muito fora do espaço de busca, impedindo assim as soluções de convergirem para regiões viáveis. Porém, ao armazenar a solução como percentual nos algoritmos, é necessário convertê-la para dados reais para execução no simulador. Seu formato é representado na tabela 4.2 e chamada neste trabalho de Solução Real.

Tabela 4.2: Solução modelada no problema de planejamento do despacho hidroelétrico para uso no simulador energético. Representa uma solução com turbinamento e vertimento, em $hm^3/s$, para 111 usinas com 60 períodos.

| Usina 1 | | | | | ... | Usina 111 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| qt1 | qv1 | ... | qt60 | qv60 | ... | qt1 | qv1 | ... | qt60 | qv60 |

Aplicando a modelagem da população apresentada nas seções anteriores sobre os algoritmos, no caso do problema de planejamento do despacho hidroelétrico aplicado ao EPSO-G foi criado um grande vetor com os indivíduos concatenados, com o tamanho correspondente ao tamanho da população. Já no LSHADE44 cada indivíduo é implementado em uma classe que possui um vetor da tabela 4.1, além das outras variáveis de fitness, identificador, entre outras.

O retorno dos valores de fitness e restrições também é armazenado um vetor contínuo no caso do PSO, seguindo o formato do vetor apresentado na tabela 4.3 e chamado neste trabalho de Fitness Completo.

Tabela 4.3: Vetor de fitness e restrições do problema de planejamento do despacho hidroelétrico.

| Usina 1 | | | | ... | Usina 111 | | | |
|---|---|---|---|---|---|---|---|---|
| custo | Vol. 1 | ... | Vol.. 60 | ... | custo | Vol. 1 | ... | Vol. 60 |

Uma característica importante do problema de planejamento do despacho hidroelétrico é a atuação do simulador, ou módulo avaliador, na população dos algoritmos. O simulador, além de avaliar a solução enviada, também possui a funcionalidade de correção da solução enviada. Para implementação desta funcionalidade foi necessário, além de enviar para avaliação, também receber o indivíduo com seus valores de vertimento e turbinamento alterados, o corrigindo na população.

Para correção dos indivíduos no algoritmo LSHADE44 as soluções são enviadas para avaliação como ponteiros, ou seja, passagem de parâmetros por referência. O simulador altera diretamente o valor do indivíduo na população. No PSO, como é utilizado um mecanismo de chamadas remotas, foi necessária a implementação de uma função que retorna o indivíduo corrigido, que é copiado sobrepondo sua versão antiga. O pseudocódigo da função de cálculo de fitness e restrições é apresentada no algoritmo 5.

**Algoritmo 5** Função de Fitness (SolucaoPercentual)

```
1:  // converte solução percentual para solução real
2:  SolucaoReal ← SolucaoPercentual
3:  // Executa simulação reparando solução enviada, calculando volumes e energia gerada
4:  SolucaoReal ← Simulador.run(SolucaoReal)
5:  // Calcula Fitness da solução simulada
6:  FitnessCompleto ← Simulador.calculaFitness(SolucaoReal)
7:  // converte para solução percentual
8:  SolucaoPercentual ← SolucaoReal
9:  return FitnessCompleto, SolucaoPercentual
```

Como é apresentado na metodologia, foram feitos vários testes, entre não corrigir, corrigir apenas o vertimento, ou corrigir apenas o turbinamento, até definir a versão de final de testes completos com correção dos valores de turbinamento e vertimento de modo que os volumes não sejam excedidos.

#### *4.4.0.1 Integrações no Problema de Planejamento do Despacho Hidroelétrico*

A integração do problema de planejamento do despacho hidroelétrico ocorre de maneira diferente para cada algoritmo implementado, devido à linguagem específica em que cada algoritmo foi desenvolvido, ocasionando escolhas de ferramentas de integração específicas para cada caso.

No algoritmo LSHADE44, como mencionado anteriormente, a integração do algoritmo com o simulador de despacho é feita pela implementação das duas funções de interface, fitness e get_bounds. O método fitness apresentado no código recebe uma solução completa, carrega esta solução nas estruturas do módulo que faz a simulação e retorna os valores de fitness da solução enviada devolvendo o fitness para o algoritmo.

O fitness na estrutura do algoritmo Pagmo é composto por um vetor onde a primeira posição contém o valor de fitness da função objetivo, seguido pelos valores de restrições do problema. Já o método get_bounds retorna para o despacho os limites mínimos e máximos de vertimento e turbinamento.

Como todos os módulos necessários para execução do problema de planejamento do despacho hidroelétrico no LSHADE44 estão implementados em C++ a integração dos códigos é feita simplesmente importando as classes e bibliotecas necessárias; incluindo no módulo principal do LSHADE44 (DE C++ na figura 4.1) as bibliotecas do *framework* Pagmo e do *framework* de simulação de despacho hidroelétrico.

No caso do algoritmo EPSO-G desenvolvido em Python a integração não foi trivial, sendo necessária a utilização de ferramenta auxiliar. Foram feitas tentativas de utilizar ferramentas como pybind11 ou boost.python, para geração de um biblioteca do simulador para adicionar e invocar diretamente no código Python, porém não foi possível pela complexidade do código do simulador, que contém estruturas dinâmicas de alocação e tipos de dados definidos em tempo de execução. A alternativa para a interface foi a utilização da biblioteca gRPC, que através da configuração de um servidor e implementação de *stubs* faz a conexão entre diferentes linguagens, gerando classes de interfaces para a comunicação.

O algoritmo EPSO-G faz a chamada ao servidor gRPC enviando as partículas individualmente. Por restrições de limite de envio de dados da plataforma gRPC não foi possível enviar a população toda. Ao receber a solução o servidor invoca a mesma função fitness chamada pelo LSHADE44, retornando o valor de custo da solução e suas restrições.

## 4.5 IMPLEMENTAÇÕES SOBRE O *BENCHMARK* CEC-2017

As soluções dos problemas do *benchmark* também foram modeladas em duas versões, porém ambas na linguagem Python. A versão para o algoritmo PSO foi construída como um grande vetor de variáveis com tamanho igual a dimensão do problema. Já no problema de planejamento do despacho hidroelétrico, como foi utilizado o *framework* Pygmo (versão do Pagmo para Python), ele segue a mesma estrutura apresentada no problema de planejamento do despacho hidroelétrico, com objetos tipo indivíduos e um vetor de população.

#### *4.5.0.1 Integrações*

Como a modelagem do *benchmark* foi construída na linguagem Python para os dois algoritmos, foi necessário construir uma interface entre avaliador do *benchmark* implementado em C++ com os algoritmos em Python. Foi utilizado a biblioteca pybind11, que gera bibliotecas Python a partir do código C++. As mudanças necessárias na implementação do avaliador do CEC são simples, apenas com definição das funções que serão expostas para o Python e utilização de tipos de dados especiais da biblioteca, que são compartilhados entre C++ e Python.

A integração do *benchmark* implementado sobre o algoritmo LSHADE44 foi transparente no processo de implementação, mesmo com o núcleo do algoritmo LSHADE44 implementado em C++. A biblioteca Pygmo já prevê a integração Pygmo/Pagmo, e modelando o algoritmo de acordo com seus requisitos o *framework* gerencia essa integração. O processo básico de execução do algoritmo é semelhante ao código fonte apresentado no Código 4.2, porém substituindo a população por uma referente ao *benchmark*.

## 4.6 TRATAMENTO DE RESTRIÇÕES UTILIZADO NOS ALGORITMOS

O tratamento de restrições em ambos algoritmos testados foi feito por $\epsilon$-constrained, porém foram feitas personalizações de acordo com o algoritmo e problema. Especificamente para o caso do *benchmark* CEC-2017 foi adicionada mais uma condição, analisando primeiramente o número de restrições violadas, seguido pelo valores das restrições. Essa alteração foi acrescentada devido ao processo de avaliação, que analisa como primeiro critério de qualidade a quantidade de soluções viáveis, em seguida os valores de restrições violadas, e por último o valor da função objetivo.

Em ambos problemas o valor de restrição $\beta$ utilizado nas comparações é a soma de todas as restrições, apresentado na equação 2.7 da seção 2.3.0.4. Em ambos algoritmos foram alterados os módulos de comparação entre duas soluções, $\epsilon$-constrained onde eram comparados apenas os valores das funções objetivos.

# 5 MATERIAIS E MÉTODOS

Este capítulo descreve os experimentos realizados e procedimentos para validações de resultados. São apresentadas informações sobre o processo de execução como parâmetros utilizados, quantidade de execuções, resultados armazenados, assim como estratégias de validação dos resultados. Os dois algoritmos são aplicados sobre os dois problemas, detalhados nos seguintes tópicos:

- Execução do EPSO-G e LSHADE-44 sobre *benchmark* CEC-2017
- Execução do EPSO-G e LSHADE-44 sobre problema de planejamento do despacho hidroelétrico: versão recuperação de reservatórios e versão de manutenção de reservatórios

Também são apresentadas aqui as informações de hardware e software utilizados na implementação e execução dos testes, assim como descrição das bases de dados utilizadas.

## 5.1 HARDWARE E SOFTWARE

Os testes foram realizados sobre equipamento desktop com processadores Intel Xeon(R) E5530 @ 2.40GHz com 8 núcleos e 12 gigabytes de memória, sobre sistema operacional Linux Ubuntu 14.04 LTS.

Para implementação e execução do algoritmos tipo PSO foi utilizada a linguagem Python 3.5, além da modelagem dos dois problemas (*benchmark* CEC-2017 e problema de planejamento do despacho hidroelétrico). O algoritmo LSHADE44 foi implementado em C++11, utilizando a biblioteca de algoritmos evolutivos Pagmo/Pygmo – Biscani et al. (2019). O problema de planejamento do despacho hidroelétrico também foi modelado em C++11 para a versão de execução com o LSHADE44.

Para integração dos algoritmos desenvolvidos em Python com os módulos de avaliação dos problemas disponibilizados somente na linguagem C++, foram utilizadas as bibliotecas pybind11 (Jakob et al., 2018) no *benchmark*, e gRPC/Protobuf [1] no problema de planejamento do despacho hidroelétrico. A biblioteca boost.python também é utilizada para a integração entre o algoritmo DE versão Python com o core de execução LSHADE44 desenvolvido em C++.

O software Matlab foi utilizado como parte do processo de validação dos resultados sobre o *benchmark*, executando o processo de avaliação e comparação das soluções dos algoritmos.

## 5.2 EPSO-G E LSHADE-44 SOBRE O *BENCHMARK* CEC-2017

A metodologia de execução deste experimento segue o padrão do *benchmark* CEC-2017 (P N Suganthan, 2017), que foi utilizado na conferência de mesmo nome em competição com algoritmos evolutivos sobre problemas com restrições. A seguir são descritos os parâmetros principais de execução:

- Execuções: 25
- Funções avaliadas: 28

[1] gRPC/Protocol Buffers - `https://grpc.io`

- Dimensões avaliadas (D): 10, 30, 50 e 100
- Avaliações máximas (FES): 2.000*D, 10.000*D, 20.000*D

As funções que definem a base de dados do *benchmark* estão apresentados no apêndice A, assim como os resultados dos algoritmos concorrentes a que os algoritmos deste trabalho foram comparados. A seguir são descritos os parâmetros de execução específicos de cada algoritmo.

### 5.2.1 Método de Execução do Algoritmo EPSO-G

Os parâmetros de execução do algoritmo EPSO-G foram definidos a partir de testes de curta duração (1 a 3 minutos) com 1000 avaliações, iniciando com valores utilizados nos trabalhos de referências, fazendo testes com pequenas variações positivas e negativas. Os parâmetros com melhores resultados, considerando viabilidade das soluções e fitness total foram utilizados nos testes completos:

- Número de partículas: 30.
- C1: 1,0 (fator de influência do gBest).
- C2: 1,0 (fator de influência do pBest).
- Velocidade inicial: inicializada com 0,05 de número aleatório dentro dos limites de borda.
- Inércia [inicial , final] : [1,0 ; 0,2] (decresce linearmente de acordo com as gerações).
- $\gamma$: 1e-10 (*threshold* para aplicação de mutação de alteração de direção).

### 5.2.2 Método de Execução do Algoritmo LSHADE44

Os parâmetros foram definidos com os mesmos valores do artigo de referência de Tvrdik e Polakova (2017), pois a intenção foi a replicação do resultado apresentado no artigo citado (algoritmo foi reimplementado sobre o mesmo problema). Seguem os parâmetros utilizados (suas definições estão apresentadas na seção 2.5 que define o algoritmo LSHADE44):

- Tamanho da população [inicial, final]: [18*D , 4].
- Tamanho grupo A: 2,6 * tamanho atual da população.
- Tamanho grupo best: 0,11 * tamanho atual da população.
- Posições de memória: vetor com tamanho de 6 posições.
- Vetor F: 6 posições inicializadas com com 0,5.
- Vetor CR: 6 posições inicializadas com 0,5.
- $n_0 = 2$.
- $\delta = 0{,}05$.

### 5.2.3 Tratamento de Restrições

Ambos algoritmos utilizam o tratamento de restrições $\epsilon$-constrained. A opção para definição dos valor de $\beta$ foi a equação 2.7, que considera a soma de todas as restrições:

- $\beta$ = equação 2.7 (soma dos valores de todas as restrições).
- $\epsilon = 0$.

O valor de $\epsilon$ prioriza o tratamento das restrições sobre o valor da função objetivo.

### 5.2.4 Resultados Coletados

Para permitir a comparação com os algoritmos de comparação (que foram aplicados sobre o mesmo *benchmark* e disponibilizam os resultados), também foi necessário seguir as regras de coleta de resultados do *benchmark* CEC-2017. Os seguintes resultados são calculados e armazenados considerando todas as funções (28 funções):

- Best: Melhor resultado da função objetivo para as soluções apresentadas entre as 25 execuções do algoritmo testado. Como todas as funções são de minimização, é o menor valor de fitness retornado entre as 25 soluções.
- Median: Valor mediano de fitness das 25 soluções.
- c: Sequência de três números indicando a quantidade de restrições violadas com valores maiores que 1,0 , entre 0,01 e 1,0 , e entre 0,0001 e 0,01, respectivamente.
- $\overline{v}$: Valor total das restrições violadas na solução mediana.
- Mean: Valor médio das soluções das 25 execuções.
- Worst: Pior solução entre as 25 execuções.
- std: Desvio padrão do conjunto das 25 execuções.
- SR: Taxa de soluções viáveis apresentadas entre as 25 execuções.
- $\overline{vio}$: Valor médio das restrições violadas entre as 25 execuções.

### 5.2.5 Avaliação e Comparação Entre Algoritmos

A comparação entre os algoritmos é feita por pontuação após o ranqueamento das soluções de cada algoritmo. O framework do *benchmark* fornece código fonte de um classificador que faz este ranqueamento e pontuação dos algoritmos comparados, fornecendo como saída uma lista com a pontuação resultante de cada algoritmo comparado.

Para execução do classificador é necessário inserir manualmente no código fonte os valores citados na seção anterior, configurando as variáveis que indicam os números de algoritmos a serem comparados.

As seguintes regras são executadas pelo algoritmo de pontuação para avaliação dos resultados:

- Ranqueamento por solução média:

  - Ordenar por taxa de soluções viáveis SR (maiores = melhores).
  - Ordenar por violação média - $\overline{vio}$.
  - Ordenar por valor da função objetivo (Best).

- Ranqueamento por solução mediana:
  - Ordenar por soluções viáveis e inviáveis.
  - Ordenar por valor da função objetivo da solução mediana (Median).
  - Ordenar por violação de restrição da solução mediana $\overline{v}$.

Após o ranqueamento o primeiro colocado recebe a pontuação 1, o segundo 2, terceiro 3, seguindo até o último algoritmo, nos dois ranqueamentos. A pontuação final é a soma das duas, e o algoritmo que obtiver a menor pontuação final é classificado como melhor algoritmo entre os comparados.

## 5.3 EPSO-G E LSHADE-44 SOBRE O PROBLEMA DE PLANEJAMENTO DO DESPACHO HIDROELÉTRICO

Foram executadas duas séries de testes para os dois algoritmos EPSO-G e LSHADE44, cada série com cenário referente ao percentual inicial e final do nível dos reservatórios, considerando capacidade máxima e o período planejado de 60 meses:

1. Manutenção de reservatórios: Nível inicial de 70% da capacidade máxima com requisito de entrega de nível de 70%.

2. Recuperação de reservatórios: Nível inicial de 30% da capacidade máxima com requisito de entrega de nível de 70%.

### 5.3.1 Execução do Experimento

As execuções foram realizadas iterando sobre um conjunto de 194 séries sintéticas de vazões que compõe a base de dados do problema, resultando em um conjunto de 194 execuções para cada algoritmo/nível. As séries contêm informações referentes a 111 usinas sobre gestão do ONS, as mesmas utilizadas no trabalho de mestrado de Zanette (2017). Os arquivos da base de dados do problema de despacho contêm, além dos valores de vazões, informações sobre cada usina, como capacidades de turbinamento, volumes mínimos e máximos, demanda e capacidades de transmissão.

De acordo com o número de vazões disponíveis na base de dados, o número de usinas e o período padrão de cálculo do problema de planejamento do despacho hidroelétrico, os parâmetros de execução ficaram definidos com os seguintes valores:

- 194 execuções, cada execução com uma série diferentes.
- Dimensão avaliada (D): 13.320 (111 usinas x 60 períodos x 2 (quantidade vertida e quantidade turbinada)).
- Avaliações máximas: 500.000 avaliações.
- Critério de saída por avaliação máxima ou,

- ftol $\leq 1 * e^{-6}$ ou,
- xtol $\leq 1 * e^{-6}$.

As restrições modeladas no problemas foram: volume mínimo e máximo de cada reservatório, por período (111 usinas por 60 períodos = 6.660 restrições), fluxo mínimo de vazão por período (111 usinas por 60 períodos = 6.660 restrições) e percentual final do nível do reservatório de cada usina (111 usinas). O total de restrições modeladas resulta em 13.431 restrições.

### 5.3.2 Definição dos Parâmetros de Execução

Partindo dos valores de parâmetros utilizados nos algoritmos aplicados sobre o *benchmark*, foram feitos testes curtos sobre o problema de planejamento do despacho hidroelétrico, com execuções de aproximadamente 20 minutos ou 500 gerações. A partir de variações positivas e negativas dos parâmetros foram selecionados os valores que apresentaram os melhores resultados, que são mostrados a seguir.
Parâmetros EPSO-G:

- Partículas: 120.
- C1 (coeficiente de influência do gBest): 0,8.
- C2 (coeficiente de influência do pBest): 0,8.
- Velocidade inicial: inicializada com 0,05 de número aleatório dentro dos limites de borda.
- Inércia [inicial , final]: [1,0 ; 0,2]
- $\gamma : 1 * e^{-10}$ (threshold para aplicação de mutação de alteração de direção).
- c = $1/dimensao^2$.

Parâmetros LSHADE44:

- Tamanho população [inicial, final]: [100 , 4].
- Número de indivíduos no grupo A: 2,6 * size(população)
- Percentual Grupo Best's : 0,11.
- Posições de memória para definição de parâmetros: 6.
- F inicial = vetor de 6 posições preenchido com 0,5 .
- CR inicial = vetor de 6 posições preenchido com 0,5 .
- n0 = 2 .
- $\delta = 0,05$ .

### 5.3.3 Definição dos Parâmetros do Simulador

O simulador de despacho hidrelétrico possui uma série de parâmetros que devem ser configurados antes da sua execução, a seguir são citados os relevantes para o experimento deste trabalho:

- Método de simulação tipo 'R' - Real: considera para o cálculo de fitness os limites reais de turbinamento, enquanto o guloso considera o volume mínimo e o conservador o volume máximo.
- Período inicial: 0 - Período de início da simulação do sistema, de acordo com a modelagem proposta será sempre a partir do período inicial 0.
- makeCorrections: Verdadeiro - Aplica correções na solução.
- relaxedViolations: Verdadeiro - Permite a avaliação de soluções inválidas, retornando fitness com restrições violadas.

A definição dos parâmetros de habilitação de correções envolveu uma conjunto de testes preliminares até alcançar uma configuração que gerasse resultados que puderam ser considerados satisfatórios. Os testes preliminares foram feitos com 500 iterações. As combinações de parâmetros no simulador e modelagem listadas abaixo foram realizadas:

- Habilitar correção apenas no vertimento – makeCorrections e relaxedViolations igual a True, com alteração do fonte do simulador.
- Habilitar correção total - makeCorrections e relaxedViolations igual a True.
- Habilitar correção total e utilizar percentual de turbinamento/vertimento – makeCorrections e relaxedViolations igual a True, com percentual aplicado na modelagem da solução.
- Habilitar correção total, utilizar percentual de turbinamento e incluir indivíduos prontos – makeCorrections e relaxedViolations igual a True, alteração da modelagem da solução e alteração dos algoritmos.

No quarto item da lista acima o teste foi incluir indivíduos prontos na população na proporção de 5%. Os indivíduos prontos foram gerados utilizando os métodos de otimização aplicados por Zanette (2017) em seus experimentos descritos em sua dissertação.

Entre as quatro opções apresentadas no corretor foi selecionada a terceira delas, com habilitação do corretor total e utilização de percentual de turbinamento/vertimento. Na primeira opção não há convergência para resultados viáveis. Na segunda e terceira opções há convergência, porém apresentando resultados melhores quando utilizado os percentuais nas soluções. Na quarta opção não houve diferença de resultados com a inclusão dos indivíduos, porém em trabalhos futuros podem ser feitos testes mais aprofundados.

### 5.3.4 Tratamento de Restrições

Assim como no problema do benchmark, em ambos algoritmos que foram aplicados sobre o problema do planejamento de despacho hidrelétrico, o método utilizado foi o $\epsilon$-constrained. Os parâmetros de tratamento de restrições foram definidos com a mesma forma de cálculo que o problema do *benchmark*:

- $\beta$ = equação 2.7 (soma dos valores de todas as restrições).
- $\epsilon = 0$.

### 5.3.5 Resultados Calculados e Coletados

Para cada uma das 194 séries executadas é selecionado o melhor indivíduo da população, ou seja, a melhor solução. Desta melhor solução são armazenados as seguintes informações:

- Fitness total e cada parcela da sua composição: fitness de energia e fitness de volume, calculados conforme as equações 2.17, 2.19 e 2.18, no capítulo 2.
- Valor total de restrições e cada parcela da sua composição : volume mínimo/máximo dos períodos (soma dos volumes excedidos no valor mínimo ou máximo em cada período da simulação), volume final e fluxo mínimo.
- Número de restrições violadas: valores totais e individuais.
- Deficit de energia e deficit de volume.

A partir dos valores citados acima são calculados valores de média, desvio padrão, pior e melhor resultado considerando restrições e fitness. São também calculados e guardados os tempos de execuções.

Além dos valores finais em cada iteração também são armazenados de fitness e restrições totais, permitindo assim observar a evolução do resultados durante a execução do experimento. O valores finais apresentados nos gráficos de evolução dos resultados apresentam a média dos valores de fitness e restrições de todas as execuções.

# 6 RESULTADOS E DISCUSSÃO

Neste capítulo são apresentados em tabelas e gráficos os resultados obtidos da aplicação dos algoritmos implementados sobre o *Benchmark* e sobre o problema do planejamento do despacho hidroelétrico, seguindo os métodos apresentados no capítulo 5. A ordem de apresentação segue a cronologia de desenvolvimento, iniciando com os resultados da aplicação dos algoritmos no *benchmark*, apresentando os resultados comparados com os demais algoritmos, passando em seguida aos resultados dos algoritmos sobre o problema de planejamento do despacho hidroelétrico.

## 6.1 ALGORITMOS APLICADOS SOBRE O *BENCHMARK* CEC-2017

Nesta seção são apresentados os resultados finais da aplicação dos algoritmos aplicados sobre o *benchmark* CEC-2017 e comparados aos quatro algoritmos que competiram na conferência CEC-2017 sobre problemas com restrições. Os resultados obtidos por algoritmos intermediários no processo de desenvolvimento também são apresentados e comparados (entre o DE simples e o LSHADE44, por exemplo), mesmo não sendo aplicados no problema de planejamento do despacho hidroelétrico. Os resultados dos algoritmos intermediários permitem observar o impacto das adições de funcionalidades sobre os algoritmos. Os seguintes algoritmos participaram da competição sobre o *benchmark* (detalhados no capítulo 3):

- CAL-SHADE : Adaptive Constraint Handling and Success History Differential Evolution (Zamuda, 2017)
- LSHADE+IDE : Success-History Based Adaptive Differential Evolution Algorithm (SHADE) with Linear population size reduction + Individual-dependent Approach (Tvrdik e Polakova, 2017)
- UDE: Unified Differential Evolution Algorithm for Constrained Optimization Problems (Trivedi et al., 2017)
- LSHADE44: Success-History Based Adaptive Differential Evolution Algorithm (SHADE) with Linear population size reduction - four competing DE strategies with extra pair (Polakova, 2017)

Os quatro algoritmos listados foram comparados com os algoritmos implementados neste trabalho, sendo o último uma implementação própria do LSHADE44 de Polakova (2017), denominado neste trabalho para fins de identificação como LSHADE44b. A lista a seguir apresenta resumidamente os algoritmos implementados neste trabalho:

- DE : Algoritmo *Differential Evolution* com tratamento de restrições $\epsilon$-constrained.
- SPSO : Standard PSO com tratamento de restrições por penalidades.
- EPSO : PSO com tratamento de restrições $\epsilon$-constrained.
- EPSO-G : Standard PSO com mutação por gradiente com tratamento de restrições $\epsilon$-constrained.

- LSHADE44b : Implementação própria do LSHADE44 de Tvrdik e Polakova (2017).

A figura 6.1 ilustra o resultado final com pontuação alcançado pelos algoritmos no *benchmark*, após a execução da avaliação comparada entre os resultados de todos algoritmos avaliados, conforme métodos apresentados no capítulo 5 que apresenta a metodologia do experimento.

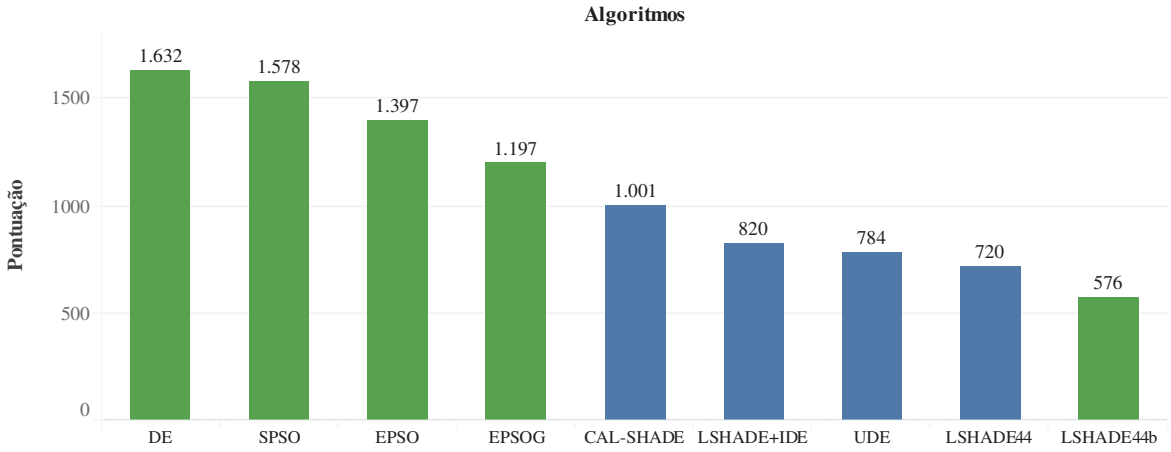



Figura 6.1: Pontuação dos algoritmos comparados sobre o *benchmark* CEC-2017, destacando em verde os implementados no trabalho e azul os algoritmos da competição CEC-2017. O valor se refere à pontuação após avaliação dos resultados finais, sendo os menores valores os melhores resultados.

A tabela 6.1 apresenta a pontuação detalhada por dimensões e resultados médios e medianos, que está apresentada concentrada no gráfico da figura 6.1. A coluna Pontuação está representada no gráfico e representa a soma da pontuação de cada algoritmo nas execuções das quatro dimensões avaliadas, considerando os resultados médios e medianos.

Tabela 6.1: Pontuação final de algoritmos sobre *benchmark*, obtida pelo módulo de avaliação das soluções do benchmark considerando todos os algoritmos comparados. Em destaque as pontuações que representam os melhores resultados.

| *Ranking* | Resultados Médias | | | | Resultados Medianos | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 10D | 30D | 50D | 100D | 10D | 30D | 50D | 100D | Pontuação |
| **LSHADE44b** | 49 | 47 | 61 | 95 | 66 | 69 | 85 | 104 | 576 |
| LSHADE44 | 117 | 86 | 85 | 79 | 95 | 92 | 83 | 83 | 720 |
| UDE | 113 | 99 | 99 | 106 | 87 | 88 | 91 | 101 | 784 |
| LSHADE+IDE | 103 | 109 | 108 | 94 | 101 | 106 | 110 | 89 | 820 |
| CAL-SHADE | 145 | 130 | 140 | 122 | 121 | 111 | 114 | 118 | 1001 |
| **EPSO-G** | 136 | 145 | 152 | 153 | 154 | 147 | 154 | 156 | 1197 |
| **EPSO** | 170 | 174 | 180 | 177 | 172 | 176 | 178 | 170 | 1397 |
| **SPSO** | 171 | 210 | 206 | 211 | 153 | 206 | 206 | 215 | 1578 |
| **DE** | 173 | 203 | 213 | 220 | 188 | 203 | 211 | 221 | 1632 |

O resultado final que gerou a tabela 6.1 executando o algoritmo de ranqueamento pode ser visto no apêndice B. No apêndice são exibidas as colocações dos algoritmos em cada critério avaliado (resultando na tabela 6.1) e em seguida são apresentados os resultados reais alcançados, a partir dos quais as colocações foram designadas aos algoritmos.

Observando o resultado final, fica evidente que os algoritmos implementados neste trabalho não apresentaram bons resultados quando comparados aos algoritmos finalistas da competição CEC-2017 (exceto pela versão implementada do LSHADE44).

Um fato a ser destacado no resultado final sobre o *benchmark* é a pontuação dos dois algoritmos LSHADEs apresentarem resultados diferentes, com o LSHADE44b apresentando resultado final melhor que o algoritmo original. Não é possível afirmar com total certeza o motivo da diferença, mas as possibilidades levantadas são as diferenças de implementação em relação ao algoritmo original, ocasionadas pelo motivo dos mecanismos não estarem totalmente descritos nos artigos de referência. Os principais pontos, que podem ou não estar diferentes do algoritmo original, são o mecanismo de redução linear do arquivo A e o tratamento de bordas. Neste trabalho optou-se por manter o tamanho do arquivo A proporcional ao tamanho da população (que é linearmente reduzida durante as gerações) e ao reduzir o arquivo A optou-se por retirar do grupo o pior indivíduo, considerando a estratégia $\epsilon$-constrained. Já o tratamento de bordas implementado foi o tradicional, onde o valor da dimensão violada é substituído pelo seu valor limite. Além destes destas diferenças, a linguagem de implementação também foi diferente, sendo que o algoritmo original foi implementado na linguagem Matlab e a versão deste trabalho foi implementada em C++, possibilitando diferentes resultados em execuções com grande número de iterações e altamente dependente da geração de números aleatórios.

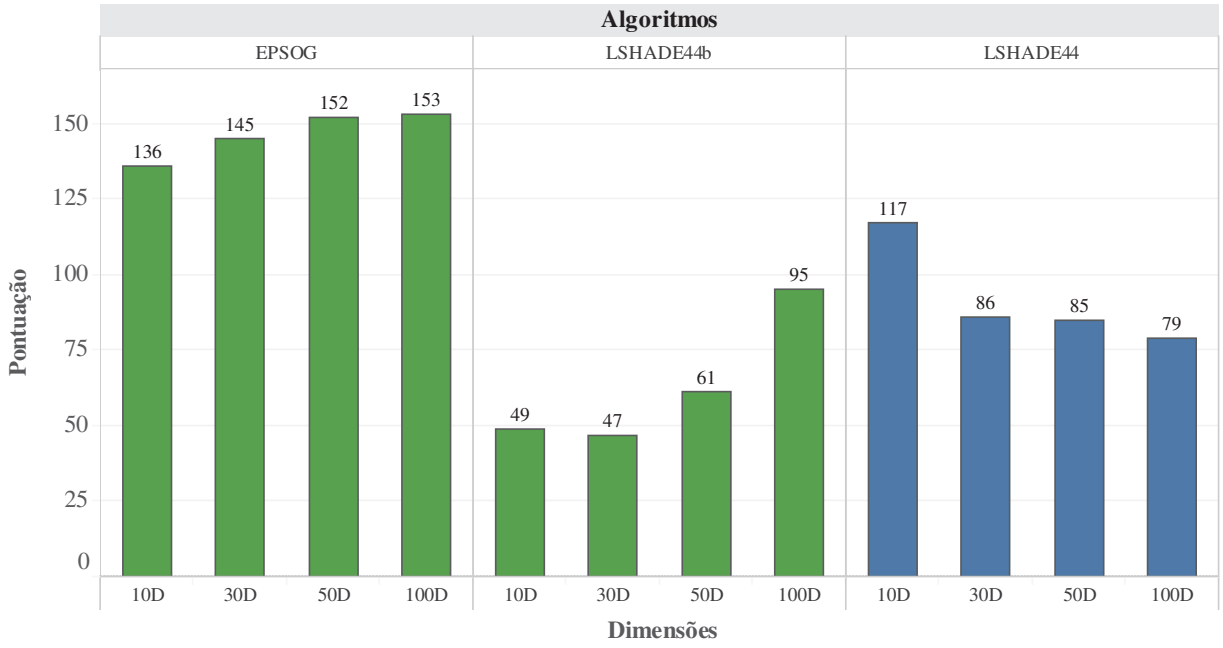



Figura 6.2: Pontuação no *benchmark* por dimensões dos algoritmos EPSO-G e LSHADE44, versão original e implementada no trabalho. Visualização da queda de desempenho com aumento das dimensões dos problemas.

O gráfico apresentado na figura 6.2 expõe a diferença de resultados entre os algoritmos EPSO-G, LSHADE44 e LSHADE44b, onde é possível observar que o LSHADE44 original apresenta uma tendência de melhor resultado com maiores dimensões. Já o algoritmo LSHADE44b apresenta uma tendência de queda de desempenho com maiores dimensões, mesmo comportamento apresentado no algoritmo EPSO-G.

A tabela 6.2 apresenta os tempos de execuções dos dois principais algoritmos sobre o conjunto completo do *benchmark*, com destaque para o tempo gasto muito superior no algoritmo EPSO-G, devido principalmente à linguagem de implementação.

Tabela 6.2: Tempos de execuções dos algoritmos EPSO-G e LSHADE44 sobre *benchmark* por dimensão aplicada. Tempo exibido no formato horas:minutos:segundos.

| Algoritmo | Tempo por Dimensões | | | |
|---|---|---|---|---|
| | 10D | 30D | 50D | 100D |
| EPSO-G | 00:39:16 | 11:18:14 | 19:34:35 | 72:52:26 |
| **LSHADE-44** | **00:00:41** | **00:05:47** | **00:18:44** | **01:42:58** |

Com a análise dos resultados apresentados nesta seção optou-se por aplicar o algoritmo LSHADE44b sobre o problema de planejamento do despacho hidroelétrico. Aproveitando a modelagem e o algoritmo já implementado, também foi aplicado o algoritmo EPSO-G, servindo assim para comparação com os resultados do LSHADE44b.

## 6.2 ALGORITMOS APLICADOS SOBRE O PROBLEMA DE PLANEJAMENTO DO DESPACHO HIDROELÉTRICO

Nesta seção são apresentados os resultados finais da aplicação dos algoritmos EPSO-G e LSHADE44b sobre o problema do planejamento do despacho hidroelétrico, de acordo com a metodologia de execução descrita no capítulo anterior. Apesar do tempo de execução não ser um objetivo específico deste trabalho, os tempos necessários para os algoritmos alcançarem as soluções foram registrados, com o algoritmo LSHADE44b consumindo um tempo médio de execução de 2h34, com maior tempo de execução em 5h24 e a menor 1h08, considerando ambos cenários de reservatórios inicializados com 70% e 30% da capacidade máxima. O tempo de execução médio para o algoritmo EPSO-G foi de 2h50, com a maior execução durando 4h04 e a mais rápida com 1h47, idem considerando ambos cenários.

A tabela 6.3 apresenta um resumo dos principais resultados das execuções dos algoritmos sobre o problema de planejamento do despacho hidroelétrico, mostrando resultados médios de restrições violadas $\beta$ (2.7), valores de fitness, fitness de energia, fitness de volume e deficit de energia. Os resultados representam as médias das execuções dos dois algoritmos sobre as 194 séries sintéticas de vazões disponíveis, aplicados sobre os dois cenários: de manutenção de reservatórios, com níveis iniciais em 70%, e de recuperação de reservatórios, com níveis iniciais em 30%. Os melhores resultados de cada categoria estão destacados.

Tabela 6.3: Resultados Médios de Restrições Violadas $\beta$, Fitness e Deficit de Energia por Algoritmo e Configuração Inicial de Reservatório. Valores de Restrições se referem às restrições violadas de Volume Mínimo e Máximo por período, Fluxo Mínimo e Volume Final.

| Algoritmo | Volume Inicial | Total Restr. Violadas $\beta$ | Fitness Total | Fitness Energia | Fitness Volume | Deficit Energia (MW) |
|---|---|---|---|---|---|---|
| LSHADE44 | 70% | **96,90** | 248,98 | 146,06 | **102,92** | 142.254,80 |
| | 30% | 164,68 | **235,52** | 124,71 | 110,81 | 117.210,23 |
| EPSO-G | 70% | 101.222,50 | 305,57 | 94,30 | 211,26 | 36.190,85 |
| | 30% | 100.722,08 | 303,90 | **91,33** | 212,57 | **22.342,67** |

Na tabela 6.3 pode ser observado a relação de proporcionalidade entre violações de restrições de volume e geração de energia: os resultados do algoritmo LSHADE44b violaram

menos restrições de volume (melhores valores de fitness de volume) apresentando maiores deficit de energia. Já os resultados de fitness de volume do algoritmo EPSO-G foram piores, aproximadamente o dobro dos valores apresentados pelo LSHADE44b, significando que mais restrições foram violadas, permitindo assim maior geração de energia e consequentemente menor deficit para atendimento da demanda, porém, em um resultado irreal na prática devido a grande inviabilidade das soluções (representaria por exemplo reservatórios com volumes negativos).

A coluna Total Restrições Violadas da tabela 6.3 apresenta o valor médio de $\beta$ definido pela equação 2.7 (soma de todas restrições), valor que apresenta o produto das restrições médias apresentadas na tabela 6.4. O total de restrições violadas representa um dos principais objetivos com a implementação do $\epsilon$-constrained nos algoritmos, a priorização da resolução das restrições. Conforme pode ser visto na tabela, o algoritmo LSHADE44 obteve o melhor resultado neste item, e a consequência de corretamente priorizar a resolução da restrição são os piores valores apresentados na coluna "Fitness". A coluna "Fitness" por sua vez é o resultado da soma do fitness de energia e fitness de volume representados na terceira e quarta coluna de resultados (fitness de volume representa uma função de minimização sobre o volume máximo do reservatório e o fitness de energia a minimização sobre o atendimento da demanda energética).

Tabela 6.4: Médias de Restrições violadas, equivalente ao Total de Restrições violadas da Tabela 6.3. Restrição de Volume representa a soma de volume mínimo e máximo de cada reservatório em cada período. Volume Final é o volume mínimo de 70% ou 30% e fluxo mínimo é a restrição por período de cada reservatório.

| Algoritmo | Volume Inicial | Restrição Violada Volumes($hm^3$) | Restrição Violada Volume Final ($hm^3$) | Restrição Violada Fluxo Mínimo ($m^3/s$) |
|---|---|---|---|---|
| LSHADE44 | 70% | **0** | **81,08** | **15,82** |
| | 30% | 0 | 141,14 | 23,54 |
| EPSOG | 70% | 0 | 94.417,82 | 6.804,14 |
| | 30% | 0 | 93.397,08 | 7.308,71 |

Não foram encontrados dados comparativos de deficit de energia específicos para o conjunto de usinas e condições aplicadas, mas é possível uma comparação básica com os dados do trabalho de Zanette (2017) apresentados na lista abaixo. A chamada solução original, presente na terceira linha, é referente à solução inicial utilizada por Zanette (2017) sobre o problema de planejamento do despacho hidroelétrico, onde a partir da solução inicial são feitas buscas locais.

- deficit melhor solução: 63.819,936 MW
- deficit médio cinco soluções aleatórias: 18.337,96 MW
- deficit solução original: 14.414,34 MW

O pior resultado para o deficit de energia apresentado neste trabalho (63.819,936 MW) quando comparado a outros trabalhos é esperado, pois prioriza diferentes objetivos. Neste trabalho a priorização é no tratamento de restrições (com a técnica de tratamento de restrições $\epsilon$-constrained), enquanto nos demais trabalhos pesquisados o foco era na minimização do deficit hidroenergético ou no custo financeiro. O deficit de energia, como modelado neste trabalho, é um dos objetivos e não uma restrição priorizada pelo $\epsilon$-constrained.

A tabela 6.5 apresenta os melhores e piores resultados entre as 194 execuções com o algoritmo LSHADE44b. O melhor resultado alcançado pelo algoritmo na configuração com 70% do volume inicial atingiu o objetivo de atender as restrições modeladas, além de obter um deficit de energia menor que a média das demais soluções com a mesma configuração. A resolução de todas as restrições altera a priorização do algoritmo com tratamento de restrições $\epsilon$-constrained, permitindo assim o algoritmo buscar soluções que otimizem a função objetivo principal do problema, o que neste caso representa o deficit de energia. Casos em que não houve essa otimização da função objetivo ocorreram principalmente porque o algoritmo priorizou até o fim das gerações a resolução das restrições.

Tabela 6.5: Resultados de Restrições Violadas $\beta$, Fitness Total, Fitness de Energia e Fitness de Volume (Fitness Total representa a soma de fitness de Energia e Volume) e Deficit de Energia total do sistema, obtidos com a aplicação do algoritmo LSHADE44b sobre o problema do planejamento do despacho hidroelétrico. As linhas apresentam as soluções para os melhores e piores resultados, médias de todos os resultados e desvio padrão, nos cenários de manutenção e recuperação de reservatórios (reservatórios inicializados em 70% e 30%, respectivamente).

| Nível Inicial Reservatório | Resultado | Restrições violadas $\beta$ | Fitness | Fitness Energia | Fitness Volume | Deficit Energia (MW) |
|---|---|---|---|---|---|---|
| 70% | melhor | **0,00** | 162,79 | 70,09 | 92,70 | 63.819,94 |
| | pior | 1.527,95 | 357,41 | 257,91 | 99,50 | 290.147,47 |
| | média | 96,90 | 248,98 | 146,06 | 102,92 | 142.254,80 |
| | desvio | 233,015 | 39,80 | 37,41 | 12,29 | 46.046,12 |
| 30% | melhor | **0,00** | **138,22** | **57,26** | **80,96** | **50.148,45** |
| | pior | 2.296,68 | 234,80 | 126,29 | 108,51 | 115.579,40 |
| | média | 164,68 | 235,52 | 124,71 | 110,81 | 117.210,23 |
| | desvio | 387,05 | 37,22 | 34,44 | 13,07 | 43.786,62 |

Observa-se na tabela 6.5, que entre as duas melhores soluções apresentadas, o cenário com reservatórios iniciais em 30% apresentou resultados finais superiores que a configuração de 70%. Mesmo iniciando com menos recursos o algoritmo LSHADE44b atendeu as restrições e obteve um menor deficit de energia e melhores volumes finais que a configuração que se iniciou com mais recursos. Uma provável explicação para este resultado está no fato de que iniciando o algoritmo com mais soluções inválidas em sua população (mais violações de restrições) o espaço de busca de soluções se torna maior, possibilitando ao longo das gerações obtenção de melhores soluções do que as apresentadas pelo algoritmo com mais soluções válidas. Este mecanismo é o mesmo utilizado ao designar valores maiores que zero para a variável $\epsilon$ na técnica de tratamento de restrições $\epsilon$-constrained, permitindo assim um maior campo de busca inicial para o algoritmo, com gradativa redução buscando atender as restrições.

A tabela 6.6 apresenta os resultados de melhores e piores soluções entre as 194 execuções do algoritmo EPSO-G. Nos resultados médios apresentados na tabela 6.6 observa-se o algoritmo não obteve sucesso em encontrar soluções viáveis, com os valores altos de restrições violadas $\beta$, quando comparados aos valores apresentado pelo algoritmo LSHADE44b no mesmo problema. Os únicos resultados que foram consideravelmente melhores, comparando com os resultados do algoritmo LSHADE44b, foram os valores de deficit de energia: a pior solução do EPSO-G (considerando valores de restrições violadas $\beta$) apresentou o melhor resultado de deficit entre todas as configurações dos dois algoritmos testados, porém, se trata da solução com maior valor de restrições violadas.

Tabela 6.6: Resultados de Restrições Violadas $\beta$, Fitness Total, Fitness de Energia e Fitness de Volume (Fitness Total representa a soma de fitness de Energia e Volume) e Deficit de Energia total do sistema, obtidos com a aplicação do algoritmo EPSO-G sobre o problema do planejamento do despacho hidroelétrico. As linhas apresentam as soluções para os melhores e piores resultados, médias de todos os resultados e desvio padrão, nos cenários de manutenção e recuperação de reservatórios (reservatórios inicializados em 70% e 30%, respectivamente).

| Nível Inicial Reservatório | Resultado | Restrições Violadas $\beta$ | Fitness | Fitness Energia | Fitness Volume | Deficit Energia (MW) |
|---|---|---|---|---|---|---|
| 70% | melhor | **64.072,04** | **298,88** | **89,107** | **209,77** | 54.772,49 |
| | pior | 122.356,26 | 304,43 | 91,25 | 213,18 | **12.217,91** |
| | media | 101.222,49 | 305,56 | 94,30 | 211,26 | 36.190,85 |
| | desvio | 10.928,29 | 10,45 | 8,38 | 5,72 | 35.186,92 |
| 30% | melhor | 66.437,66 | 312,59 | 100,63 | 211,95 | 30.886,59 |
| | pior | 127.590,07 | 308,14 | 97,46 | 210,68 | 78.063,95 |
| | media | 100.722,08 | 303,90 | 91,32 | 212,57 | 22.342,67 |
| | desvio | 10.667,18 | 10,01 | 7,19 | 5,36 | 27.499,40 |

É importante observar que as melhores e piores soluções, apresentadas pelos dois algoritmos nas tabelas 6.5 e 6.6, são dependentes das séries de vazões aplicadas sobre o problema. Os melhores casos em ambos algoritmos coincidem com as melhores séries de vazões, ou seja, séries com mais recursos para atender as demandas.

O gráfico 6.3 apresenta as médias das quantidades de restrições violadas por algoritmo entre as 194 execuções, juntamente com a média dos valores de deficit de energia no eixo secundário. Observa-se no gráfico a superioridade do algoritmo LSHADE44b em resolver as restrições quando comparado ao EPSO-G. Outro ponto a ser destacado no gráfico da figura 6.3 é a relação entre a menor quantidade de restrições violadas e o aumento do deficit de energia. Para o cumprimento de restrições como volume final, por exemplo, foi necessário deixar de utilizar a água para turbinamento com consequente geração de energia e diminuição de deficit. Ao mesmo tempo para atender a demanda de fluxo mínimo pode ter sido exigido o vertimento em usinas mesmo com volume baixo, deixando de produzir energia em períodos diferentes onde seria necessário uma maior quantidade de água.

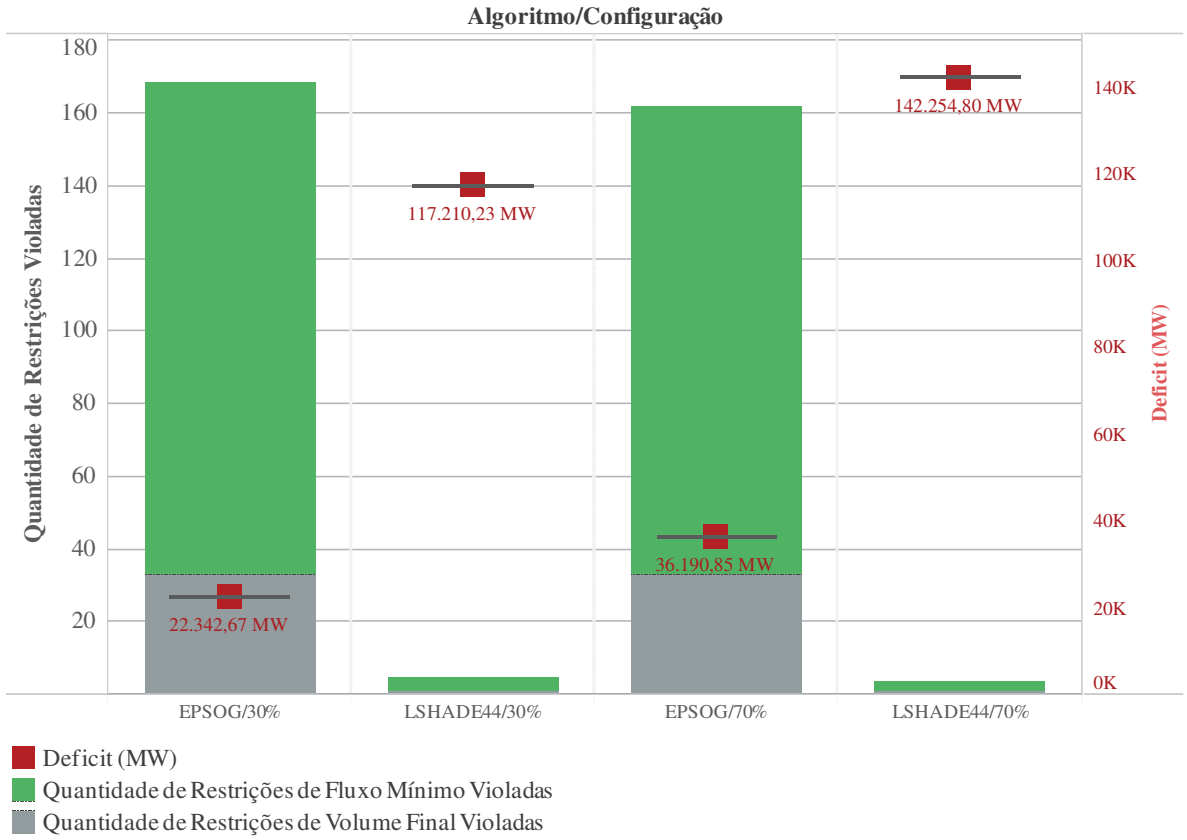



Figura 6.3: Comparação entre quantidade de restrições violadas e aumento do deficit de energia para os algoritmos EPSO-G e LSHADE44b aplicados ao problema do planejamento do despacho hidroelétrico com níveis iniciais dos reservatórios em 30% e 70%.

A tabela 6.7 apresenta algumas amostras de valores de vertimentos, turbinamento, volume resultante e energia gerada, de uma solução gerada pelo algoritmo LSHADE44b. A primeira coluna representa o identificador da usina (na amostra apresentada nesta tabela são apresentadas apenas quatro usinas entre as 194 usinas). A segunda coluna identifica o período (são apresentados 5 períodos para cada usina, entre os 60 períodos por usina do problema). As duas colunas seguintes representam o vertimento e turbinamento configurados pelo algoritmo. O Volume resultante apresenta o volume do reservatório após as ações operacionais de vertimento e turbinamento somadas às afluências naturais da série simulada (chuvas) e turbinamento e vertimento das usinas a montante. A coluna Energia Gerada apresenta a energia produzida em MW pela usina considerando a água turbinada e sua capacidade de produção.

Tabela 6.7: Exemplo parcial de solução gerada pelo LSHADE44b. Identificador da Usina Id representa a identificação numérica da usina no problema. Para cada usina Id: Coluna Período representa os períodos de 0 a 4 (problema completo apresenta períodos de 0 a 59); coluna Vertimento representa a quantidade de água vertida e coluna Turbinamento representa a quantidade de água utilizada na geração de energia, ambos em $m^3/s$; Volume Resultante representa o volume resultante no período em $hm^3$; Energia gerada representa a energia produzida no período.

| Identificador da Usina Id | Período | Vertimento ($m^3/s$) | Turbinamento ($m^3/s$) | Volume Resultante($hm^3$) | Energia Gerada (MW) |
|---|---|---|---|---|---|
| 304 | 0 | 0 | 77,44 | 5,00 | 58,85 |
| 304 | 1 | 18,91 | 80,00 | 5,00 | 60,80 |
| 304 | 2 | 1,74 | 80,00 | 5,00 | 60,80 |
| 304 | 3 | 0 | 71,50 | 5,00 | 54,34 |
| 304 | 4 | 0 | 55,46 | 5,00 | 42,15 |
| 290 | 0 | 34,58 | 44,77 | 133,83 | 19,11 |
| 290 | 1 | 25,80 | 58,16 | 102,34 | 24,28 |
| 290 | 2 | 21,34 | 38,28 | 143,38 | 16,25 |
| 290 | 3 | 30,01 | 45,00 | 133,28 | 19,22 |
| 290 | 4 | 19,59 | 36,91 | 131,59 | 15,84 |
| 278 | 0 | 176,46 | 207,11 | 4.675,59 | 95,74 |
| 278 | 1 | 88,46 | 138,36 | 4.582,90 | 63,60 |
| 278 | 2 | 70,34 | 251,14 | 4.519,77 | 113,90 |
| 278 | 3 | 42,96 | 145,12 | 4.575,66 | 66,55 |
| 278 | 4 | 25,90 | 59,91 | 4.585,50 | 27,78 |
| 272 | 0 | 105,88 | 96,56 | 279,41 | 12,59 |
| 272 | 1 | 83,05 | 86,47 | 365,19 | 11,51 |
| 272 | 2 | 137,34 | 130,88 | 386,03 | 17,26 |
| 272 | 3 | 128,90 | 136,86 | 360,43 | 18,03 |
| 272 | 4 | 120,34 | 141,08 | 360,42 | 18,37 |

Os gráficos apresentados nas figuras 6.4 e 6.5 as evoluções por geração das médias de fitness e valores violados de restrições $\beta$ (2.9) para os algoritmos LSHADE44b e EPSO-G no problema do planejamento de despacho. Cada figura possui dois gráficos que representam os cenários de manutenção (inicializados em 70%) e recuperação (inicializados em 30%) de reservatórios. As médias dos valores de fitness e violações são calculados para os 30 melhores e 30 piores casos, considerando a técnica $\epsilon$-constrained (que prioriza as soluções viáveis).

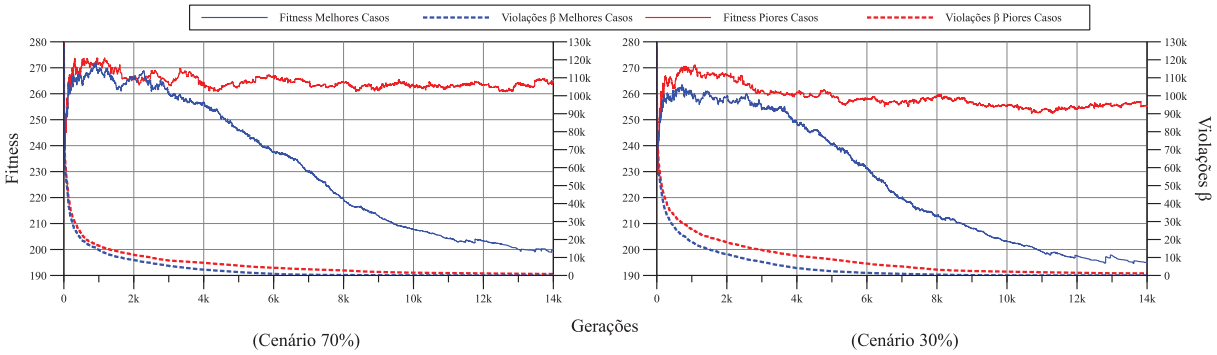



Figura 6.4: Fitness Médio (linhas sólidas) e Restrições Violadas $\beta$ Médias (linhas tracejadas) por geração para soluções geradas pelo algoritmo LSHADE44b para o problema do despacho hidroelétrico. Cenários configurados com reservatórios inicializados com 70% e 30% da capacidade máxima. Melhores e Piores Casos representam a média das 30 melhores (linhas azuis) e piores (linhas vermelhas) soluções.

Analisando o comportamento do gráfico da figura 6.4 é possível verificar a ação do tratamento de restrições $\epsilon$-constrained, em ambos cenários. A ação do tratamento de restrições, priorizando a viabilidade, provoca subidas e quedas nos valores do fitness, comportamento diferentes do tradicional em algoritmos evolutivos. A variável que sofre uma queda constante é a violação $\beta$. Nos melhores casos (linhas azuis) observa-se que somente após as restrições terem sido resolvidas o valor de fitness inicia sua minimização, ainda que neste gráfico as restrições não aparentem estar totalmente resolvidas devido a apresentação da média de várias soluções. Nos piores caso (linhas vermelhas), o algoritmo não consegue resolver as restrições, desta maneira ele não altera a priorização para a otimização da função objetivo, que se mantém praticamente estável durante todas as gerações.

No gráfico da figura 6.4 também é possível observar resultado semelhante ao apresentado na tabela 6.5, onde a configuração de inicialização com 30% da capacidade (com menor quantidade inicial de recursos) obteve os melhores resultados para a função objetivo, sendo que a hipótese para explicar este comportamento é a mesma: o valor inicial mais inviável aumenta o espaço de busca inicial. Os gráficos permitem observar, como informação adicional a tabela 6.5, que a configuração de 70% apresenta uma curva de convergência mais brusca que a de 30%, com o início de execução com mais restrições resolvidas. Com mais restrições violadas o algoritmo consegue encontrar melhores valores para a função objetivo (atendimento da demanda e bons volumes nos reservatórios), e a partir destas soluções com melhores fitness são tratadas as restrições. Este comportamento pode ser visto comparando os gráficos de 70% e 30% na geração 4.000, por exemplo, onde nos melhores casos são violadas praticamente a mesma quantidade de restrições, porém o valor de fitness é consideravelmente melhor para o configuração de 30% (55.000 para a configuração de 30% e 60.000 para a configuração de 70%).

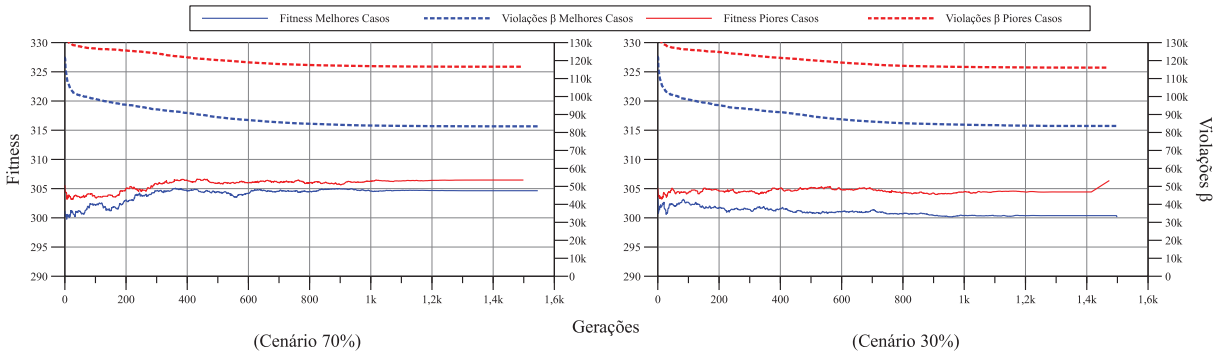



Figura 6.5: Fitness Médio (linhas sólidas) e Restrições Violadas $\beta$ Médias (linhas tracejadas) por geração para soluções geradas pelo algoritmo EPSO-G para o problema do despacho hidroelétrico. Cenários configurados com reservatórios inicializados com 70% e 30% da capacidade máxima. Melhores e Piores Casos representam a média das 30 melhores (linhas azuis) e piores (linhas vermelhas) soluções.

Nos gráficos do algoritmo EPSO-G aplicado ao planejamento do despacho hidroelétrico apresentados na figura 6.5, os comportamentos se destacam são a falta de convergência e a parada precoce, quando comparados ao gráficos do algoritmo LSHADE44b. O EPSO-G faz a parada de sua evolução por volta da geração 1.500, em todas as configurações, enquanto o LSHADE44b finaliza sua execução por volta da geração 14.000. As paradas dos dois algoritmos obedecem ao critério de saída por diferença mínima de fitness ou de posição, indicando no caso do EPSO-G que houve perda de diversidade da população, com todos os indivíduos apresentando resultados muito semelhantes. Pode-se concluir que o principal problema do algoritmo EPSO-G aplicado no problema foi a falta de diversidade.

Analisando os valores de fitness e restrições violadas apresentadas na figura 6.5 observa-se que houve uma queda nos valores de violação $\beta$ de aproximadamente 20% em relação as soluções iniciais aleatórias, em ambos cenários, com os valores de fitness com uma leve piora, resultados muito longe de encontrar soluções viáveis. Assim como no algoritmo LSHADE44b houve melhor resultado para fitness no cenário com menos recursos, cabendo a mesma hipótese de que mais soluções inviáveis aumentam o espaço de busca inicial. Em ambos cenários a quantidade final de restrições violadas foi a mesma, resultado alcançado na média dos melhores e piores casos.

# 7 CONCLUSÃO

Neste trabalho foi desenvolvido um experimento com aplicação dos algoritmos evolutivos EPSO-G e LSHADE44 com tratamento de restrições ao problema de planejamento do despacho hidroelétrico. O processo de desenvolvimento incluiu fases de pesquisa e levantamento bibliográfico. Em seguida, com o propósito de testes comparativos, foram implementados e executados testes de algoritmos candidatos sobre o *benchmark* de problemas com restrições da conferência *IEEE Congress on Evolutionary Computation (CEC 2017)*. A partir da análise dos resultados apresentados pelos algoritmos sobre o *benchmark*, os algoritmos EPSO-G e LSHADE44 foram selecionados para aplicação no problema de planejamento do despacho hidroelétrico.

A aplicação dos algoritmos selecionados no problema de planejamento do despacho hidroelétrico foi bem-sucedida, com resultados considerados bons principalmente com o algoritmo LSHADE44. Neste caso houve evolução das soluções aleatórias para soluções com resultados comparáveis aos demais algoritmos e nos melhores casos atendimento total das restrições, atingindo assim o principal objetivo proposto do trabalho. Como esperado, os resultados do algoritmo EPSO-G apresentaram muitas violações de restrições, resultado previsto pois sua aplicação no problema do planejamento de despacho possuía apenas o objetivo de comparação e direcionamento para novas implementações, pois o algoritmos já não havia apresentado bons resultados no *benchmark* de problemas com restrições.

O algoritmo LSHADE44 apresentou sucesso na funcionalidade de recuperação e manutenção dos níveis de reservatórios, atendendo totalmente essa restrição nos melhores casos com configuração de reservatórios iniciais com 70% da capacidade. No cenário de recuperação com reservatório inicial em 30% a restrição apresentou praticamente o mesmo resultado.

Ambos algoritmos obtiveram sucesso ao manipular soluções com alta dimensionalidade, com tempos de execução aceitáveis e convergência à melhores soluções, ainda que a convergência tenha sido pequena no caso do algoritmo EPSO-G. Trabalhos relacionados em metaheurística citam problemas com alta dimensionalidade como 100 a 500 dimensões, mas a modelagem do problema de planejamento do despacho hidroelétrico proposta apresentou 13.320 dimensões. Como citado no desenvolvimento do trabalho, somente os algoritmos somados à técnica de restrição $\epsilon$-constrained não apresentaram convergência, porém, com a atuação do tratamento de restrições por reparação do simulador, boas soluções foram geradas.

A utilização das ferramentas de integração escolhidas cumpriu satisfatoriamente a função pretendida, possibilitando integrar os módulos de simulador de despacho energético com os algoritmos implementados em diferentes linguagens, assim como no caso do benchmark.

## 7.1 TRABALHOS FUTUROS

Durante o desenvolvimento e análise de resultados deste trabalho, foram identificados pontos que podem ser objeto de pesquisa e desenvolvimento em novos projetos:

- O tempo de execução pode ser melhorado com análise detalhada dos códigos implementados, pois a prioridade deste trabalho não foi a otimização de tempo de execução. No caso do algoritmo EPSO-G a mudança de linguagem de Python para C++ já resultaria em melhores tempos. Bibliotecas como MKL–*Intel® Math Kernel Library (Intel® MKL)* podem ser aplicadas nos cálculos dos algoritmos, assim como processamento

paralelo das avaliações das partículas/indivíduos. Finalmente, o uso de equipamentos dedicados e com maior poder de processamento também pode contribuir neste aspecto, visto que os processos deste trabalho foram executados em computadores domésticos ou de escritório, com hardwares defasados.

- O algoritmo PSO pode ser mais explorado, aplicando novas versões e mecanismos. O EPSO-G é um dos algoritmos intermediários do MLPSO - *Mutation Linear PSO*, que envolve mecanismos avançados e hibridização com algoritmos CMA-ES –*Covariance Matrix Adaptation Evolutionary Strategy* e SQP–*Sequential quadratic programming*, que por priorização da aplicação do algoritmo LSHADE44 não foi aplicado.

- A modelagem do problema de planejamento do despacho hidroelétrico pode ser alterada para otimização por bacias hidrográficas, compondo uma solução completa no fim da execução. Esta alteração deve proporcionar um melhor desempenho dos algoritmos evolutivos, já que não foram projetados para um número tão alto de dimensões como foi aplicado neste trabalho.

- As últimas versões do simulador de despacho hidroelétrico contam com cálculos de custo financeiro das programações, que podem ser incluídos para uma maior completude do experimento.

- Ao utilizar o tratamento de restrições por reparação do simulador de despacho as restrições modeladas nos algoritmos evolutivos podem estar sendo tratadas desnecessariamente. Podem ser aplicados testes sobre que restrições são necessárias tratar com $\epsilon$-constrained para que o algoritmo priorize o que de fato é necessário.

- Novos testes com variações dos valores de $\epsilon$-constrained podem ser feitos, permitindo uma área de busca maior.

- Diante da modelagem com um número alto de dimensões, podem ser pesquisados algoritmos específicos para aplicação sobre esta categoria de problemas.

- Os códigos desenvolvidos podem ser melhorados com construções de interfaces amigáveis, documentação e organização, de modo que seja simples variar os problemas de aplicação e algoritmos para resolução, semelhante ao framework Pagmo de Biscani et al. (2019).

# REFERÊNCIAS


Agência Nacional de Energia Elétrica, A. (2008). *Atlas de energia elétrica do Brasil*, volume 3. ANEEL.

Agência Nacional de Energia Elétrica, A. (2017). Boletim de Informações Gerenciais. Relatório técnico, ANEEL - Agência Nacional de Energia Elétrica.

Ahlers, F. J., Carlo, W. D., Fleiner, C., Godwin, L., Keenan), M. M., Nath, R. D., Neumaier, A., Phillips, J. R., Price, K., Storn, R., Turney, P., Wang, F.-S., Zandt, J. V., Geldon, H., Gauden, P. A., Brauer, C., Shivaram, K. e Novikov, D. (2018). Differential evolution homepage - icsi, berkeley. `http://www.icsi.berkeley.edu/~storn/code.html`. Acessado em 19/12/2018.

Bessa, M. R., Vallejos, C., Detzel, D., Mine, M., Marcílio, D. C., Oening, A. P., Matioli, L. C., Haas, P., Fernandes, T., Silva, F. et al. (2013). Otimização do despacho hidrotérmico mediante algoritmos híbridos com computação de alto desempenho: modelo phoenixvii. Em *Congresso de Inovação Tecnológica em Energia Elétrica CITENEL. Anais... Rio de Janeiro: Inovação Tecnológica em Energia Elétrica*.

Biao, S., Chang hua, H., Xin hua, Y. e Chuan, H. (2014). Mutation particle swarm optimization algorithm for solving the optimal operation model of thermal power plants. *Journal of Renewable and Sustainable Energy*, 6(4):043118.

Biscani, F. et al. (2019). esa/pagmo2: pagmo 2.10. `https://esa.github.io/pagmo2`.

Bonyadi, M., Li, X. e Michalewicz, Z. (2013). A hybrid particle swarm with velocity mutation for constraint optimization problems. Em *Proceedings of the 15th annual conference on Genetic and evolutionary computation*, páginas 1–8. ACM.

Bonyadi, M. R. e Michalewicz, Z. (2017). Particle Swarm Optimization for Single Objective Continuous Space Problems: A Review. *Evolutionary Computation*, 25(1):1–54.

Cepel (2018). Newave - modelo de planejamento da operação de sistemas hidrotérmicos interligados de longo e médio prazo. `http://www.cepel.br/pt_br/produtos/newave-modelo-de-planejamento-da-operacao-de-sistemas-hidrotermicos-interligados-de-longo-e-medio-prazo.htm`. Acessado em 14/03/2019.

Chang, W. (2010). A Novel Particle Swarm Optimization for Optimal Scheduling of Hydrothermal System. *Energy and Power Engineering*, 02(04):223–229.

Cheng, S., Shi, Y. e Qin, Q. (2011). Experimental Study on Boundary Constraints Handling in Particle Swarm Optimization. *International Journal of Swarm Intelligence Research*, 2(3):43–69.

Coath, G. e Halgamuge, S. K. (2003). A comparison of constraint-handling methods for the application of particle swarm optimization to constrained nonlinear optimization problems. Em *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*, volume 4, páginas 2419–2425. IEEE.

Coello, C. A. C. (2002). Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. *Computer methods in applied mechanics and engineering*, 191(11-12):1245–1287.

Du, K.-l. e Swamy, M. (2016). *Search and Optimization by Metaheuristics*. Birkhäuser.

Engelbrecht, A. (2012). Particle swarm optimization: Velocity initialization. Em *2012 IEEE Congress on Evolutionary Computation*, páginas 1–8. IEEE.

Engelbrecht, A. P. (2007). *Computational Intelligence: An Introduction*. Wiley Publishing, 2nd edition.

Harpman, D., Platt, J. e Coberly, S. (2012). Advanced Algorithms for Hydropower Optimization. Relatório Técnico Optimization Algorithms, U.S. Bureau of Reclamation's Science and Technology Program, Denver, Colorado.

Homaifar, A., Qi, C. X. e Lai, S. H. (1994). Constrained optimization via genetic algorithms. *Simulation*, 62(4):242–253.

Hu, Y. (2009). Hybrid-fitness function evolutionary algorithm based on simplex crossover and pso mutation for constrained optimization problems. *International Journal of Pattern Recognition and Artificial Intelligence*, 23(01):115–127.

Jakob, W., Rhinelander, J. e Moldovan, D. (2018). pybind11 — seamless operability between c++11 and python. `https://github.com/pybind/pybind11`. Acessado em 21/12/2018.

Kennedy, J. e Eberhart, R. (1995). Particle swarm optimization. Em *Proceedings of ICNN'95 - International Conference on Neural Networks*, volume 4, páginas 1942–1948 vol.4.

Koziel, S. e Michalewicz, Z. (1999). Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. *Evolutionary computation*, 7(1):19–44.

Liang, J. J., Zhigang, S. e Zhihui, L. (2010). Coevolutionary comprehensive learning particle swarm optimizer. Em *Evolutionary Computation (CEC), 2010 IEEE Congress on*, páginas 1–8. IEEE.

Marcondes, E. J. (2019). Lshade44b code repository. `https://gitlab.c3sl.ufpr.br/ejmarcondes/LSHADE44b`.

Michalewicz, Z. (1995). A survey of constraint handling techniques in evolutionary computation methods. Em *Proceedings of the 4th Annual Conference on Evolutionary Programming*, páginas 135–155. MIT Press.

Morales, A. K. e Quezada, C. V. (1998). A universal eclectic genetic algorithm for constrained optimization. Em *Proceedings of the 6th European congress on intelligent techniques and soft computing*, volume 1, páginas 518–522.

Moreno, S. R. e Kaviski, E. (2013). Daily Scheduling of Small Hydro Power Plants Dispatch With Modified Particles Swarm Optimization. *Pesquisa Operacional*, 35(1):25–37.

Oliveira, E. d. S. (2015). *Metaheurísticas aplicadas ao problema do despacho econômico de energia elétrica*. Tese de doutorado, Universidade Federal de Juiz de Fora, Faculdade de Engenharia.

Ongsakul, W. e Vo, D. N. (2013). *Artificial Intelligence in Power System Optimization*. CRC Press.

ONS (2017). ONS - Operador Nacional do Sistema Elétrico. `http://www.ons.org.br/pt/paginas/sobre-o-sin/mapas`. Acessado em 18/12/2017.

P N Suganthan, Mostafa Z Ali, G. W. R. M. J. J. L. B. Y. Q. (2017). Special session & competitions on real-parameter single objective optimization (3 different cases). `http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2017/CEC2017.htm`. Acessado em 28/10/2017.

Polakova, R. (2017). L-SHADE with competing strategies applied to constrained optimization. Em *2017 IEEE Congress on Evolutionary Computation (CEC)*, páginas 1683–1689. IEEE.

Shi, Y. e Eberhart, R. (1998). A modified particle swarm optimizer. Em *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, páginas 69–73. IEEE.

Takahama, T. e Sakai, S. (2005a). Constrained optimization by $\epsilon$ constrained particle swarm optimizer with $\epsilon$-level control. Em Abraham, A., Dote, Y., Furuhashi, T., Köppen, M., Ohuchi, A. e Ohsawa, Y., editores, *Soft Computing as Transdisciplinary Science and Technology*, páginas 1019–1029, Berlin, Heidelberg. Springer Berlin Heidelberg.

Takahama, T. e Sakai, S. (2005b). Constrained Optimization by $\epsilon$ Constrained Particle Swarm Optimizer with $\epsilon$-level Control. *Soft Computing as Transdisciplinary Science and Technology*, 29(3):1019–1029.

Takahama, T. e Sakai, S. (2006a). Constrained optimization by the $\epsilon$ constrained differential evolution with gradient-based mutation and feasible elites. Em *2006 IEEE International Conference on Evolutionary Computation*, páginas 1–8.

Takahama, T. e Sakai, S. (2006b). Solving constrained optimization problems by the $\epsilon$ constrained particle swarm optimizer with adaptive velocity limit control. Em *2006 IEEE Conference on Cybernetics and Intelligent Systems*, páginas 1–7.

Takahama, T. e Sakai, S. (2010). Constrained optimization by the $\epsilon$ constrained differential evolution with an archive and gradient-based mutation. Em *Evolutionary Computation (CEC), 2010 IEEE Congress on*, páginas 1–9. IEEE.

Takahama, T., Sakai, S. e Iwane, N. (2006). Solving nonlinear constrained optimization problems by the $\epsilon$ constrained differential evolution. Em *2006 IEEE International Conference on Systems, Man and Cybernetics*, volume 3, páginas 2322–2327.

Talbi, E. (2009). *Metaheuristics: From Design to Implementation*. Wiley Series on Parallel and Distributed Computing. Wiley.

Tanabe, R. e Fukunaga, A. (2013). Success-history based parameter adaptation for differential evolution. Em *Evolutionary Computation (CEC), 2013 IEEE Congress on*, páginas 71–78. IEEE.

Trivedi, A., Sanyal, K., Verma, P. e Srinivasan, D. (2017). A unified differential evolution algorithm for constrained optimization problems. Em *2017 IEEE Congress on Evolutionary Computation (CEC)*, páginas 1231–1238. IEEE.

Tvrdik, J. e Polakova, R. (2017). A simple framework for constrained problems with application of L-SHADE44 and IDE. Em *2017 IEEE Congress on Evolutionary Computation (CEC)*, páginas 1436–1443. IEEE.

Wu, G., Mallipeddi, R. e Suganthan, P. (2016). Problem definitions and evaluation criteria for the cec 2017 competition on constrained real-parameter optimization. *National University of Defense Technology, Changsha, Hunan, PR China and Kyungpook National University, Daegu, South Korea and Nanyang Technological University, Singapore, Technical Report*.

Zamuda, A. (2017). Adaptive constraint handling and Success History Differential Evolution for CEC 2017 Constrained Real-Parameter Optimization. Em *2017 IEEE Congress on Evolutionary Computation (CEC)*, páginas 2443–2450. IEEE.

Zanette, B. N. (2017). Aplicação de métodos clássicos de busca local no problema de planejamento do despacho hidrelétrico. Dissertação de Mestrado, Universidade Federal do Paraná - UFPR.

Zhang, W.-J. Z. W.-J., Xie, X.-F. X. X.-F. e Bi, D.-C. B. D.-C. (2004). Handling boundary constraints for numerical optimization by particle swarm flying in periodic search space. *Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No.04TH8753)*, 2:2307–2311.

# APÊNDICE A – FUNÇÕES DO *BENCHMARK* CEC-2017

Este apêndice contém as funções utilizadas na implementação que apresentou os resultados preliminares sobre o benchmark CEC 2017 - P N Suganthan (2017). Todas as funções aqui representadas estão presentes em Wu et al. (2016):

$$Minimizar: \ f(X), \ \ X = (x_1, x_2, .., x_n) \ and \ X \in S \tag{A.1}$$

$$Sujeito\ a: \begin{cases} g_i(X) \leq 0, \ \ i = 1, ..., p \\ h_j(X) = 0, \ \ j = p+1, ..., m \end{cases} \tag{A.2}$$

$$\left|h_j(X)\right| - \epsilon \leq 0, \ for \ j = p+1, ..., m \tag{A.3}$$

$$C01: \ Min \ f(x) = \sum_{i=1}^{D} \left( \sum_{j=1}^{i} z_j \right)^2 \quad z = x - o$$

$$g(x) = \sum_{i=1}^{D} \left[ z_i^2 - 5000cos(0.1\pi z_i) - 4000 \right] \leq 0$$

$$x \in [-100, 100]^D$$

$$C02: \ Min \ f(x) = \sum_{i=1}^{D} \left( \sum_{j=1}^{i} z_j \right)^2 \quad z = x - o, \ y = M * z$$

$$g(x) = \sum_{i=1}^{D} \left[ y_i^2 - 5000cos(0.1\pi y_i) - 4000 \right] \leq 0$$

$$x \in [-100, 100]^D$$

$$C03: \ Min\ f(x) = \sum_{i=1}^{D}\left(\sum_{j=1}^{i} z_j\right)^2 \quad z = x - o$$

$$g(x) = \sum_{i=1}^{D}\left[z_i^2 - 5000cos(0.1\pi z_i) - 4000\right] \leq 0$$

$$h(x) = -\sum_{i=1}^{D} z_i sin(0.1\pi z_i) = 0$$

$$x \in [-100, 100]^D$$

$$C04: \ Min\ f(x) = \sum_{i=1}^{D}\left[z_i^2 - 10cos(2\pi z_i) + 10\right] \quad z = x - o$$

$$g_1(x) = -\sum_{i=1}^{D} z_i sin(2z_i) \leq 0$$

$$g_2(x) = \sum_{i=1}^{D} z_i sin(z_i) \leq 0$$

$$x \in [-10, 10]^D$$

$$C05: \ Min\ f(x) = \sum_{i=1}^{D-1}\left(100(z_i^2 - z_{+1})^2 + (z_i - 1)^2\right) \quad z = x - o,\ y = M_1 * z,\ w = M_2 * z$$

$$g_1(x) = \sum_{i=1}^{D}\left[y_i^2 - 50cos(2\pi y_i) - 40\right] \leq 0$$

$$g_2(x) = \sum_{i=1}^{D}\left[w_i^2 - 50cos(2\pi w_i) - 40\right] \leq 0$$

$$x \in [-10, 10]^D$$

$$C06: \ Min\ f(x) = \sum_{i=1}^{D} \left[ z_i^2 - 10cos(2\pi z_i) + 10 \right] \quad z = x - o$$

$$h_1(x) = -\sum_{i=1}^{D} z_i sin(z_i) = 0$$

$$h_2(x) = \sum_{i=1}^{D} z_i sin(\pi z_i) = 0$$

$$h_3(x) = -\sum_{i=1}^{D} z_i cos(z_i) = 0$$

$$h_4(x) = \sum_{i=1}^{D} z_i cos(\pi z_i) = 0$$

$$h_5(x) = \sum_{i=1}^{D} (z_i sin(2 * \sqrt{|z_i|})) = 0$$

$$h_6(x) = -\sum_{i=1}^{D} (z_i sin(2 * \sqrt{|z_i|})) = 0$$

$$x \in [-20, 20]^D$$

$$C07: \ Min\ f(x) = \sum_{i=1}^{D} \left( z_i sin(z_i) \right) \quad z = x - o$$

$$h_1(x) = \sum_{i=1}^{D} \left( z_i - 100cos(0.5z_i) + 100 \right) = 0$$

$$h_2(x) = -\sum_{i=1}^{D} \left( z_i - 100cos(0.5z_i) + 100 \right) = 0$$

$$x \in [-50, 50]^D$$

$$C08: \ Min\ f(x) = max(z) \ \ z = x - o,\ y_i = z_{2l-1},\ w_l = z_{2l} \ onde\ l = 1, ..., D/2$$

$$h_1(x) = \sum_{i=1}^{D/2} \left( \sum_{j=1}^{i} y_j \right)^2 = 0$$

$$h_2(x) = \sum_{i=1}^{D/2} \left( \sum_{j=1}^{i} w_j \right)^2 = 0$$

$$x \in [-100, 100]^D$$

$$C09: \ Min\ f(x) = max(z) \ \ z = x - o, \ y_l = z_{2l-1}, \ w_l = z_{2l} \ onde \ l = 1, ..., D/2$$

$$g(x) = \prod_{i=1}^{D/2} w_i \leq 0$$

$$h(x) = \sum_{i=1}^{D/2-1} (y_i^2 - y_{i+1})^2 = 0$$

$$x \in [-10, 10]^D$$

$$C10: \ Min\ f(x) = max(z) \ \ z = x - o$$

$$h_1(x) = \sum_{i=1}^{D} \left( \sum_{j-1}^{i} z_j \right)^2 = 0$$

$$h_2(x) = \sum_{i=1}^{D-1} (z_i - z_{i+1})^2 = 0$$

$$x \in [-100, 100]^D$$

$$C11: \ Min\ f(x) = \sum_{i=1}^{D} (z_i) \ \ z = x - o$$

$$g(x) = \prod_{i=1}^{D} z_i \leq 0$$

$$h(x) = \sum_{i=1}^{D-1} (z_i - z_{i+1})^2 = 0$$

$$x \in [-100, 100]^D$$

$$C12: \ Min\ f(x) = \sum_{i=1}^{D} (y_i^2 - 10cos(2\pi y_i) + 10), \ y = x - o$$

$$g_1(x) = 4 - \sum_{i=1}^{D} \left| y_i \right| \leq 0$$

$$g_2(x) = \sum_{i=1}^{D} y_i^2 - 4 = 0$$

$$x \in [-100, 100]^D$$

$$C13: \ Min\ f(x) = \sum_{i=1}^{D-1} (100(y_i^2 - y_{i+1})^2 + (y_i - 1)^2), \ y = x - o$$

$$g_1(x) = \sum_{i=1}^{D}(y_i^2 - 10cos(2\pi y_i) + 10) - 100 \leq 0$$

$$g_2(x) = \sum_{i=1}^{D} y_i - 2D \leq 0$$

$$g_3(x) = 5 - \sum_{i=1}^{D} y_i \leq 0$$

$$x \in [-100, 100]^D$$

$$C14: \ Min\ f(x) = -20exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} y_i^2}\right) + 20 - exp\left(\frac{1}{D}\sum_{i=1}^{D} cos(2\pi y_i)\right) + e, \ y = x - o$$

$$g(x) = \sum_{i=2}^{D} y_i^2 + 1 - |y_1| \leq 0$$

$$h(x) = \sum_{i=1}^{D} y_i^2 - 4 = 0$$

$$x \in [-100, 100]^D$$

$$C15: \ Min\ f(x) = max\{|y_i|, 1 \leq i \leq D\}, \ y = x - o$$

$$g(x) = \sum_{i=1}^{D} y_i^2 - 100D \leq 0$$

$$h(x) = cosf(x) + sinf(x) = 0$$

$$x \in [-100, 100]^D$$

$$C16: \ Min\ f(x) = \sum_{i=1}^{D} |y_i|, \ y = x - o$$

$$g(x) = \sum_{i=1}^{D} y_i^2 - 100D \leq 0$$

$$h(x) = (cosf(x) + sinf(x))^2 - exp(cosf(x) + sinf(x)) - 1 + exp(1) = 0$$

$$x \in [-100, 100]^D$$

$$C17: \ Min\ f(x) = \frac{1}{4000}\sum_{i=1}^{D} y_i^2 + 1 - \prod_{i=1}^{D} cos\left(\frac{y_i}{\sqrt{i}}\right),\ y = x - o$$

$$g(x) = 1 - \sum_{i=1}^{D} sgn(|y_i| - \sum_{j=1,2...D, j\neq i}^{D} y_j^2 - 1) \leq 0$$

$$h(x) = \sum_{i=1}^{D} y_i^2 - 4D = 0$$

$$x \in [-100, 100]^D$$

$$C18: \ Min\ f(x) = \sum_{i=1}^{D} (z_i^2 - 10cos(2\pi z_i) + 10),\ z_i = \begin{cases} y_i, \text{ se } |y_i| < 0.5 \\ 0.5\text{*round}(2\text{*}y_i), \text{ senão} \end{cases}, y = x - o$$

$$g_1 = 1 - \sum_{i=1}^{D} |y_i| \leq 0$$

$$g_2(x) = \sum_{i=1}^{D} y_1^2 - 100D \leq 0$$

$$h(x) = \sum_{i=1}^{D} 100(y_i^2 - y_{i+1}) + \prod_{i=1}^{D} sin^2(y_i - 1)\pi = 0$$

$$x \in [-100, 100]^D$$

$$C19: \ Min\ f(x) = \sum_{i=1}^{D} (|y_i|^{0.5} + 2siny_i^3),\ y = x - o$$

$$g_1(x) = \sum_{i=1}^{D} ((-10exp(-0.2\sqrt{y_i^2 + y_{i+1}^2})) + (D-1)\frac{10}{exp(-5)} \leq 0$$

$$g_2(x) = \sum_{i=1}^{D} sin^2(2y_i) - 0.5D \leq 0$$

$$x \in [-50, 50]^D$$

$$C20: \ min\ f(x) = \sum_{i=1}^{D-1} g(y_i, y_{i+1}) + g(y_D, y_1), g(y_i, y_{i+1}) = 0.5 + \frac{\left(sin^2\left(\sqrt{y_i^2 + y_{i+1}^2}\right) - 0.5\right)}{\left(1 + 0.001\left(\sqrt{y_i^2 + y_{i+1}^2}\right)\right)^2},$$

$$y = x - o$$

$$g_1(x) = cos^2\left(\sum_{i=1}^{D} y_i\right) - 0.25cos\left(\sum_{i=1}^{D} y_i\right) - 0.125 \leq 0$$

$$g_2(x) = exp\left(cos\left(\sum_{i=1}^{D} y_i\right)\right) - exp(0.25) \leq 0$$

$$x \in [-100, 100]^D$$

$$C21: \ Min\ f(x) = \sum_{i=1}^{D}(y_i^2 - 10cos(2\pi y_i) + 10),\ z = M(x - o)$$

$$g_1(x) = 4 - \sum_{i=1}^{D}|z_i| \leq 0$$

$$g_2(x) = \sum_{i=1}^{D} z_i^2 - 4 = 0$$

$$x \in [-100, 100]^D$$

$$C22: \ Min\ f(x) = \sum_{i=1}^{D}(100(z_i^2 - x_{i+1})^2 + (z_i - 1)^2),\ z = M(x - o)$$

$$g_1(x) = \sum_{i=1}^{D}(z_i^2 - 10cos(2\pi z_i) + 10) - 100 \leq 0$$

$$g_2(x) = \sum_{i=1}^{D} z_i - 2D \leq 0$$

$$g_3(x) = 5 - \sum_{i=1}^{D} z_i \leq 0$$

$$x \in [-100, 100]^D$$

$$C23: \ Min\ f(x) = -20exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} z_i^2}\right) + 20 - exp\left(\frac{1}{D}\sum_{i=1}^{D} cos(2\pi z_i)\right) + e,$$

$$z = M(x - o)$$

$$g(x) = \sum_{i=2}^{D} z_i^2 + 1 - |z_1| \leq 0$$

$$h(x) = \sum_{i=1}^{D} z_i^2 - 4 = 0$$

$$x \in [-100, 100]^D$$

$$C24: \ Min\ f(x) = max\left\{|z_i|, 1 \leq i \leq D\right\},\ z = M(x-o)$$

$$g(x) = \sum_{i=1}^{D} z_i^2 - 100D \leq 0$$

$$h(x) = cosf(z) + sinf(z) = 0$$

$$x \in [-100, 100]^D$$

$$C25: \ Min\ f(x) = \sum_{i=1}^{D} |z_i|,\ z = M(x-o)$$

$$g(x) = \sum_{i=1}^{D} z_i^2 - 100D \leq 0$$

$$h(x) = (cosf(z) + sinf(z))^2 - exp(cosf(z) + sinf(z)) - 1 + exp(1) = 0$$

$$x \in [-100, 100]^D$$

$$C26: \ Min\ f(x) = \frac{1}{4000}\sum_{i=1}^{D} y_i^2 + 1 - \prod_{i=1}^{D} cos(\frac{y_i}{\sqrt{i}}),\ z = M(x-o)$$

$$g(x) = 1 - \sum_{i=1}^{D} sgn(|z_i| - \sum_{j=1,2...D, j\neq i}^{D} z_j^2 - 1) \leq 0$$

$$h(x) = \sum_{i=1}^{D} z_i^2 - 4D = 0$$

$$x \in [-100, 100]^D$$

$$C27: \ Min\ f(x) = \sum_{i=1}^{D} (z_i^2 - 10cos(2\pi z_i) + 10),\ z_= \begin{cases} y_i, \text{ se } |y_i| < 0.5 \\ 0.5 * round(2 * y_i), \text{ senão} \end{cases} \quad z = M(x-o)$$

$$g_1 = 1 - \sum_{i=1}^{D} |y_i| \leq 0$$

$$g_2 = (x) = \sum_{i=1}^{D} y_i^2 - 100D \leq 0$$

$$h(x) = \sum_{i=1}^{D} 100(y_i^2 - y_{i+1}) + \prod_{i=1}^{D} sin^2(y_i - 1)\pi = 0$$

$$x \in [-100, 100]^D$$

$$C28: \ \text{Min } f(x) = \sum_{i=1}^{D} (|z_i|^{0.5} + 2\sin z_i^3), \ = M(x - o)$$

$$g_1(x) = \sum_{i=1}^{D-1} \left( -10exp\left( -0.2\sqrt{z_i^2 + z_{i+1}^2} \right) \right) + (D-1)\frac{10}{exp(-5)} \leq 0$$

$$g_2(x) = \sum_{i=1}^{D} sin^2(2z_i) - 0.5D \leq 0$$

$$x \in [-50, 50]^D$$

# APÊNDICE B – RESULTADOS DO *BENCHMARK* CEC-2017

Este apêndice apresenta os resultados da execução dos algoritmos selecionados sobre o benchmark CEC-2017, conforme apresentado no capítulo 6. Os resultados dos algoritmos implementados neste trabalho foram coletados nas execuções, já os resultados dos algoritmos de comparações estão disponíveis no site da conferência CEC-2017 (P N Suganthan (2017)).

As colunas das tabelas contém as funções nomeadas de C01 a C28. Cada linha apresenta os resultados de um dos algoritmos, com as implementações feitas neste trabalho destacadas nas últimas linhas das tabelas.

A tabela B.1 apresenta a pontuação dos resultados **médios** do ranqueamento entre os algoritmos comparados em funções com **10 dimensões**. A soma da pontuação desta tabela e a tabela de resultados médios leva a pontuação total dos algoritmos para 10 dimensões. A soma dos pares de tabelas com resultados de execuções em 10, 30, 50 e 100 dimensões leva ao resultado final apresentado na tabela 6.1 no capítulo 6.

Tabela B.1: Pontuação sobre média dos resultados das 25 execuções dos algoritmos comparados sobre o *benchmark* CEC-2017. Execuções sobre 10 dimensões. Colunas representam funções 1 a 28

| 10 Dimensões – Colocação por Resultado Médio | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CAL-SHADE | 1 | 1 | 8 | 7 | 9 | 2 | 4 | 5 | 8 | 1 | 4 | 1 | 7 | 3 | 7 | 8 | 6 | 9 | 1 | 6 | 7 | 6 | 3 | 7 | 9 | 5 | 8 | 2 | 145 |
| LSHADE44IDE | 1 | 1 | 3 | 4 | 1 | 6 | 3 | 1 | 4 | 2 | 1 | 3 | 1 | 5 | 3 | 4 | 2 | 8 | 8 | 3 | 1 | 3 | 5 | 6 | 5 | 2 | 9 | 8 | 103 |
| LSHADE44 | 1 | 1 | 5 | 3 | 1 | 5 | 2 | 1 | 7 | 2 | 3 | 7 | 1 | 4 | 8 | 5 | 3 | 7 | 7 | 2 | 5 | 4 | 4 | 4 | 6 | 3 | 7 | 9 | 117 |
| UDE | 1 | 1 | 1 | 6 | 8 | 3 | 1 | 1 | 6 | 2 | 6 | 1 | 6 | 2 | 5 | 2 | 4 | 6 | 9 | 7 | 6 | 5 | 2 | 3 | 2 | 4 | 6 | 7 | 113 |
| SPSO | 1 | 1 | 6 | 5 | 6 | 8 | 8 | 9 | 1 | 9 | 8 | 8 | 8 | 8 | 1 | 7 | 9 | 5 | 5 | 4 | 8 | 7 | 8 | 8 | 3 | 9 | 5 | 6 | 171 |
| EPSO | 1 | 1 | 4 | 9 | 5 | 9 | 9 | 6 | 1 | 6 | 9 | 9 | 5 | 9 | 9 | 1 | 8 | 4 | 4 | 9 | 9 | 8 | 9 | 9 | 1 | 8 | 4 | 4 | 170 |
| EPSOG | 1 | 1 | 7 | 8 | 4 | 1 | 7 | 7 | 1 | 7 | 5 | 5 | 4 | 6 | 6 | 9 | 5 | 2 | 6 | 8 | 3 | 2 | 6 | 5 | 7 | 6 | 2 | 5 | 136 |
| DE | 9 | 9 | 9 | 1 | 7 | 7 | 6 | 8 | 9 | 8 | 7 | 6 | 9 | 7 | 4 | 6 | 7 | 3 | 3 | 5 | 4 | 9 | 7 | 2 | 8 | 7 | 3 | 3 | 173 |
| **LSHADE44b** | 1 | 1 | 2 | 1 | 1 | 4 | 5 | 1 | 4 | 2 | 1 | 3 | 1 | 1 | 2 | 3 | 1 | 1 | 2 | 1 | 1 | 1 | 1 | 1 | 4 | 1 | 1 | 1 | **49** |

Complementando a tabela anterior, a tabela B.2 apresenta a pontuação dos algoritmos por resultados medianos para 10 dimensões.

Tabela B.2: Pontuação sobre resultado mediano entre as 25 execuções dos algoritmos comparados sobre o *benchmark* CEC-2017. Execuções sobre 10 dimensões.Colunas representam funções 1 a 28

| 10 Dimensões – Colocação por Resultado Mediano | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CAL-SHADE | 1 | 1 | 8 | 7 | 1 | 3 | 2 | 5 | 8 | 1 | 1 | 2 | 1 | 3 | 6 | 8 | 7 | 9 | 1 | 6 | 3 | 1 | 4 | 8 | 7 | 5 | 9 | 3 | 121 |
| LSHADE44IDE | 1 | 1 | 7 | 1 | 1 | 5 | 3 | 1 | 1 | 2 | 1 | 3 | 1 | 5 | 4 | 5 | 1 | 8 | 8 | 3 | 4 | 1 | 5 | 5 | 6 | 3 | 8 | 7 | 101 |
| LSHADE44 | 1 | 1 | 4 | 3 | 1 | 4 | 4 | 1 | 1 | 2 | 1 | 8 | 1 | 4 | 7 | 4 | 2 | 6 | 7 | 1 | 1 | 1 | 3 | 4 | 5 | 2 | 7 | 9 | 95 |
| UDE | 1 | 1 | 2 | 5 | 1 | 1 | 1 | 1 | 7 | 2 | 6 | 1 | 1 | 1 | 2 | 2 | 3 | 7 | 9 | 7 | 1 | 1 | 1 | 3 | 2 | 4 | 6 | 8 | 87 |
| SPSO | 1 | 1 | 3 | 6 | 8 | 7 | 8 | 9 | 1 | 9 | 8 | 5 | 6 | 8 | 1 | 3 | 8 | 5 | 5 | 4 | 9 | 8 | 8 | 1 | 3 | 9 | 5 | 4 | 153 |
| EPSO | 1 | 1 | 1 | 9 | 7 | 9 | 9 | 6 | 1 | 6 | 9 | 9 | 9 | 9 | 9 | 1 | 9 | 4 | 4 | 8 | 8 | 7 | 9 | 9 | 1 | 8 | 4 | 5 | 172 |
| EPSOG | 1 | 1 | 5 | 8 | 6 | 2 | 7 | 7 | 1 | 7 | 5 | 6 | 7 | 6 | 8 | 7 | 6 | 2 | 6 | 9 | 6 | 6 | 6 | 7 | 8 | 6 | 2 | 6 | 154 |
| DE | 9 | 9 | 9 | 4 | 9 | 8 | 6 | 8 | 9 | 8 | 7 | 7 | 8 | 7 | 5 | 9 | 5 | 3 | 3 | 5 | 7 | 9 | 7 | 6 | 9 | 7 | 3 | 2 | 188 |
| **LSHADE44b** | 1 | 1 | 6 | 1 | 1 | 6 | 5 | 1 | 1 | 2 | 1 | 3 | 1 | 2 | 3 | 6 | 4 | 1 | 2 | 2 | 4 | 1 | 2 | 2 | 4 | 1 | 1 | 1 | **66** |

A seguir serão apresentados os pares de tabelas de colocação para os resultados com 30, 50 e 100 dimensões.

Tabela B.3: Pontuação sobre média dos resultados das 25 execuções dos algoritmos comparados sobre o *benchmark* CEC-2017. Execuções sobre 30 dimensões. Colunas representam funções 1 a 28

| 30 Dimensões – Colocação por Resultado Médio | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CAL-SHADE | 1 | 1 | 6 | 6 | 6 | 2 | 5 | 1 | 7 | 1 | 4 | 5 | 6 | 2 | 7 | 9 | 6 | 9 | 2 | 4 | 2 | 6 | 5 | 7 | 7 | 6 | 6 | 1 | 130 |
| LSHADE44IDE | 1 | 1 | 4 | 3 | 1 | 4 | 3 | 5 | 1 | 5 | 3 | 3 | 3 | 5 | 3 | 4 | 2 | 7 | 8 | 5 | 5 | 4 | 4 | 4 | 3 | 2 | 8 | 8 | 109 |
| LSHADE44 | 1 | 1 | 3 | 2 | 1 | 3 | 2 | 2 | 1 | 2 | 2 | 2 | 2 | 4 | 5 | 5 | 1 | 4 | 7 | 2 | 4 | 5 | 3 | 3 | 4 | 1 | 5 | 9 | 86 |
| UDE | 1 | 1 | 1 | 4 | 5 | 1 | 1 | 4 | 6 | 4 | 6 | 6 | 4 | 3 | 2 | 2 | 3 | 5 | 9 | 6 | 3 | 2 | 2 | 2 | 1 | 4 | 4 | 7 | 99 |
| SPSO | 8 | 8 | 5 | 5 | 7 | 8 | 8 | 9 | 1 | 9 | 9 | 9 | 8 | 9 | 8 | 6 | 9 | 8 | 5 | 3 | 9 | 9 | 9 | 9 | 8 | 9 | 9 | 6 | 210 |
| EPSO | 1 | 1 | 9 | 9 | 4 | 9 | 9 | 6 | 8 | 6 | 8 | 7 | 7 | 7 | 9 | 1 | 7 | 3 | 3 | 9 | 7 | 7 | 6 | 8 | 9 | 7 | 3 | 4 | 174 |
| EPSOG | 1 | 1 | 8 | 7 | 8 | 7 | 6 | 7 | 1 | 7 | 5 | 4 | 5 | 6 | 6 | 7 | 5 | 2 | 6 | 8 | 6 | 3 | 7 | 5 | 5 | 5 | 2 | 5 | 145 |
| DE | 9 | 9 | 7 | 8 | 9 | 6 | 7 | 8 | 9 | 8 | 7 | 8 | 9 | 8 | 4 | 8 | 8 | 6 | 4 | 7 | 8 | 8 | 8 | 6 | 6 | 8 | 7 | 3 | 203 |
| **LSHADE44b** | 1 | 1 | 2 | 1 | 1 | 5 | 4 | 2 | 1 | 2 | 1 | 1 | 1 | 1 | 1 | 3 | 4 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 3 | 1 | 2 | **47** |

Tabela B.4: Pontuação sobre resultado mediano entre as 25 execuções dos algoritmos comparados sobre o *benchmark* CEC-2017. Execuções sobre 30 dimensões.Colunas representam funções 1 a 28

| 30 Dimensões – Colocação por Resultado Mediano | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CAL-SHADE | 1 | 1 | 5 | 5 | 1 | 2 | 5 | 1 | 2 | 1 | 4 | 4 | 5 | 3 | 7 | 9 | 6 | 8 | 1 | 3 | 1 | 6 | 3 | 7 | 7 | 5 | 7 | 1 | 111 |
| LSHADE44IDE | 1 | 1 | 4 | 1 | 1 | 4 | 4 | 5 | 2 | 5 | 3 | 1 | 1 | 5 | 3 | 5 | 2 | 9 | 8 | 4 | 5 | 2 | 5 | 4 | 3 | 1 | 9 | 8 | 106 |
| LSHADE44 | 1 | 1 | 2 | 2 | 1 | 3 | 2 | 2 | 2 | 2 | 2 | 3 | 4 | 4 | 6 | 6 | 1 | 3 | 7 | 1 | 3 | 4 | 4 | 3 | 4 | 2 | 8 | 9 | 92 |
| UDE | 1 | 1 | 1 | 4 | 1 | 1 | 1 | 4 | 6 | 4 | 6 | 5 | 3 | 2 | 1 | 2 | 3 | 7 | 9 | 6 | 1 | 1 | 1 | 1 | 1 | 3 | 5 | 7 | 88 |
| SPSO | 8 | 8 | 8 | 6 | 8 | 8 | 8 | 9 | 2 | 8 | 9 | 9 | 9 | 9 | 9 | 3 | 9 | 4 | 5 | 5 | 9 | 9 | 9 | 9 | 9 | 9 | 4 | 4 | 206 |
| EPSO | 1 | 1 | 9 | 9 | 6 | 9 | 9 | 6 | 9 | 6 | 8 | 7 | 7 | 7 | 8 | 1 | 7 | 2 | 4 | 9 | 7 | 7 | 6 | 8 | 8 | 7 | 3 | 5 | 176 |
| EPSOG | 1 | 1 | 7 | 7 | 7 | 5 | 6 | 7 | 1 | 7 | 5 | 6 | 6 | 6 | 4 | 7 | 5 | 5 | 6 | 8 | 6 | 5 | 7 | 5 | 6 | 6 | 2 | 3 | 147 |
| DE | 9 | 9 | 6 | 8 | 9 | 7 | 7 | 8 | 8 | 9 | 7 | 8 | 8 | 8 | 5 | 8 | 8 | 6 | 3 | 7 | 8 | 8 | 8 | 6 | 5 | 8 | 6 | 6 | 203 |
| **LSHADE44b** | 1 | 1 | 3 | 3 | 1 | 6 | 3 | 2 | 7 | 2 | 1 | 2 | 1 | 1 | 2 | 4 | 4 | 1 | 2 | 2 | 4 | 3 | 2 | 2 | 2 | 4 | 1 | 2 | **69** |

Tabela B.5: Pontuação sobre média dos resultados das 25 execuções dos algoritmos comparados sobre o *benchmark* CEC-2017. Execuções sobre 10 dimensões. Colunas representam funções 1 a 28

| 50 Dimensões – Colocação por Resultado Médio | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CAL-SHADE | 7 | 7 | 5 | 5 | 6 | 2 | 5 | 5 | 7 | 3 | 4 | 5 | 5 | 2 | 7 | 9 | 6 | 6 | 2 | 4 | 3 | 5 | 5 | 6 | 6 | 6 | 6 | 1 | 140 |
| LSHADE44IDE | 5 | 1 | 3 | 2 | 1 | 4 | 4 | 4 | 5 | 6 | 2 | 1 | 3 | 5 | 3 | 3 | 2 | 5 | 8 | 5 | 5 | 1 | 4 | 4 | 4 | 2 | 7 | 9 | 108 |
| LSHADE44 | 1 | 1 | 2 | 1 | 1 | 3 | 3 | 1 | 4 | 1 | 3 | 6 | 2 | 4 | 4 | 4 | 1 | 4 | 7 | 2 | 6 | 2 | 3 | 3 | 3 | 1 | 4 | 8 | 85 |
| UDE | 1 | 1 | 1 | 4 | 7 | 1 | 1 | 3 | 3 | 5 | 5 | 4 | 4 | 3 | 1 | 1 | 4 | 8 | 9 | 6 | 1 | 4 | 1 | 2 | 1 | 3 | 8 | 7 | 99 |
| SPSO | 8 | 8 | 6 | 6 | 5 | 7 | 8 | 9 | 2 | 9 | 9 | 9 | 8 | 8 | 9 | 7 | 9 | 9 | 4 | 3 | 9 | 9 | 9 | 9 | 9 | 9 | 5 | 4 | 206 |
| EPSO | 6 | 5 | 9 | 9 | 4 | 9 | 9 | 6 | 8 | 4 | 8 | 7 | 7 | 6 | 8 | 6 | 7 | 3 | 3 | 9 | 7 | 7 | 6 | 7 | 7 | 7 | 3 | 3 | 180 |
| EPSOG | 1 | 6 | 8 | 7 | 8 | 8 | 6 | 7 | 1 | 7 | 6 | 3 | 6 | 9 | 5 | 5 | 5 | 2 | 6 | 8 | 2 | 6 | 8 | 5 | 5 | 4 | 2 | 6 | 152 |
| DE | 9 | 9 | 7 | 8 | 9 | 6 | 7 | 8 | 9 | 8 | 7 | 8 | 9 | 7 | 6 | 8 | 8 | 7 | 5 | 7 | 8 | 8 | 7 | 8 | 8 | 8 | 9 | 5 | 213 |
| **LSHADE44b** | 1 | 1 | 4 | 3 | 1 | 5 | 2 | 2 | 6 | 1 | 1 | 2 | 1 | 1 | 2 | 2 | 3 | 1 | 1 | 1 | 4 | 3 | 2 | 1 | 2 | 5 | 1 | 2 | **61** |

Tabela B.6: Pontuação sobre resultado mediano entre as 25 execuções dos algoritmos comparados sobre o *benchmark* CEC-2017. Execuções sobre 50 dimensões.Colunas representam funções 1 a 28

| 50 Dimensões – Colocação por Resultado Mediano | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CAL-SHADE | 1 | 1 | 3 | 5 | 1 | 2 | 5 | 6 | 5 | 3 | 4 | 3 | 5 | 2 | 8 | 9 | 6 | 8 | 1 | 2 | 3 | 5 | 2 | 6 | 6 | 5 | 6 | 1 | 114 |
| LSHADE44IDE | 1 | 1 | 5 | 1 | 1 | 4 | 3 | 4 | 6 | 6 | 2 | 1 | 2 | 5 | 3 | 5 | 2 | 7 | 8 | 5 | 5 | 2 | 5 | 4 | 3 | 2 | 8 | 9 | 110 |
| **LSHADE44** | 1 | 1 | 2 | 2 | 1 | 3 | 2 | 1 | 4 | 1 | 1 | 6 | 1 | 4 | 4 | 4 | 1 | 4 | 7 | 1 | 6 | 3 | 4 | 3 | 2 | 1 | 5 | 8 | **83** |
| UDE | 1 | 1 | 1 | 4 | 5 | 1 | 1 | 3 | 3 | 5 | 6 | 2 | 4 | 3 | 2 | 2 | 3 | 6 | 9 | 6 | 1 | 1 | 1 | 2 | 1 | 3 | 7 | 7 | 91 |
| SPSO | 8 | 8 | 8 | 7 | 8 | 7 | 8 | 9 | 2 | 9 | 8 | 9 | 8 | 8 | 9 | 3 | 9 | 5 | 5 | 4 | 9 | 9 | 9 | 9 | 9 | 9 | 4 | 6 | 206 |
| EPSO | 7 | 6 | 9 | 9 | 6 | 9 | 9 | 5 | 9 | 4 | 9 | 7 | 7 | 6 | 7 | 1 | 7 | 3 | 3 | 8 | 7 | 7 | 6 | 7 | 7 | 7 | 3 | 3 | 178 |
| EPSOG | 1 | 7 | 7 | 6 | 7 | 8 | 6 | 7 | 1 | 7 | 5 | 4 | 6 | 9 | 5 | 8 | 5 | 2 | 6 | 9 | 2 | 6 | 8 | 5 | 5 | 6 | 2 | 4 | 154 |
| DE | 9 | 9 | 6 | 8 | 9 | 6 | 7 | 8 | 8 | 8 | 7 | 8 | 9 | 7 | 6 | 7 | 8 | 9 | 4 | 7 | 8 | 8 | 7 | 8 | 8 | 8 | 9 | 5 | 211 |
| LSHADE44b | 1 | 1 | 4 | 3 | 4 | 5 | 4 | 2 | 7 | 2 | 3 | 5 | 3 | 1 | 1 | 6 | 4 | 1 | 2 | 3 | 4 | 4 | 3 | 1 | 4 | 4 | 1 | 2 | 85 |

Tabela B.7: Pontuação sobre média dos resultados das 25 execuções dos algoritmos comparados sobre o *benchmark* CEC-2017. Execuções sobre 10 dimensões. Colunas representam funções 1 a 28

| 100 Dimensões – Colocação por Resultado Médio | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CAL-SHADE | 4 | 4 | 5 | 5 | 4 | 1 | 5 | 5 | 7 | 4 | 3 | 5 | 4 | 1 | 6 | 7 | 6 | 8 | 1 | 1 | 3 | 5 | 4 | 6 | 6 | 6 | 5 | 1 | 122 |
| LSHADE44IDE | 6 | 5 | 3 | 2 | 2 | 5 | 3 | 2 | 2 | 3 | 1 | 1 | 1 | 4 | 2 | 2 | 2 | 5 | 8 | 5 | 5 | 2 | 2 | 2 | 3 | 2 | 6 | 8 | 94 |
| **LSHADE44** | 1 | 1 | 2 | 1 | 1 | 4 | 4 | 1 | 1 | 1 | 2 | 6 | 2 | 3 | 4 | 3 | 1 | 4 | 7 | 3 | 6 | 1 | 3 | 1 | 2 | 1 | 4 | 9 | **79** |
| UDE | 2 | 2 | 1 | 4 | 6 | 2 | 1 | 3 | 8 | 5 | 4 | 2 | 6 | 2 | 3 | 1 | 3 | 6 | 9 | 6 | 2 | 6 | 1 | 3 | 1 | 3 | 7 | 7 | 106 |
| SPSO | 8 | 9 | 8 | 7 | 8 | 7 | 8 | 9 | 3 | 9 | 7 | 8 | 8 | 8 | 9 | 8 | 8 | 9 | 5 | 2 | 8 | 8 | 8 | 9 | 9 | 8 | 9 | 4 | 211 |
| EPSO | 7 | 6 | 9 | 9 | 3 | 9 | 9 | 7 | 3 | 6 | 9 | 7 | 7 | 5 | 7 | 6 | 7 | 3 | 3 | 9 | 7 | 7 | 5 | 7 | 7 | 7 | 3 | 3 | 177 |
| EPSOG | 5 | 7 | 7 | 6 | 7 | 8 | 6 | 6 | 3 | 7 | 6 | 4 | 5 | 9 | 5 | 5 | 5 | 2 | 4 | 8 | 4 | 3 | 9 | 5 | 4 | 5 | 2 | 6 | 153 |
| DE | 9 | 8 | 6 | 8 | 9 | 6 | 7 | 8 | 9 | 8 | 8 | 9 | 9 | 7 | 8 | 9 | 9 | 7 | 6 | 7 | 9 | 9 | 7 | 8 | 8 | 9 | 8 | 5 | 220 |
| LSHADE44b | 3 | 3 | 4 | 3 | 5 | 3 | 2 | 4 | 6 | 2 | 5 | 3 | 3 | 6 | 1 | 4 | 4 | 1 | 2 | 4 | 1 | 4 | 6 | 4 | 5 | 4 | 1 | 2 | 95 |

Tabela B.8: Pontuação sobre resultado mediano entre as 25 execuções dos algoritmos comparados sobre o *benchmark* CEC-2017. Execuções sobre 100 dimensões.Colunas representam funções 1 a 28

| 100 Dimensões – Colocação por Resultado Mediano | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CAL-SHADE | 4 | 4 | 6 | 5 | 3 | 2 | 5 | 5 | 6 | 5 | 4 | 3 | 4 | 1 | 6 | 7 | 5 | 7 | 2 | 1 | 1 | 5 | 2 | 7 | 5 | 5 | 7 | 1 | 118 |
| LSHADE44IDE | 5 | 5 | 4 | 1 | 1 | 4 | 2 | 2 | 2 | 4 | 2 | 2 | 1 | 4 | 1 | 3 | 2 | 4 | 8 | 3 | 4 | 2 | 3 | 3 | 3 | 2 | 4 | 8 | 89 |
| **LSHADE44** | 1 | 1 | 2 | 2 | 2 | 3 | 3 | 1 | 1 | 1 | 1 | 6 | 2 | 3 | 4 | 4 | 1 | 5 | 7 | 2 | 5 | 1 | 4 | 4 | 2 | 1 | 5 | 9 | **83** |
| UDE | 2 | 2 | 1 | 4 | 5 | 1 | 1 | 3 | 7 | 6 | 5 | 1 | 6 | 2 | 2 | 2 | 3 | 6 | 9 | 6 | 2 | 6 | 1 | 1 | 1 | 3 | 6 | 7 | 101 |
| SPSO | 8 | 8 | 8 | 7 | 8 | 7 | 8 | 9 | 3 | 9 | 7 | 8 | 8 | 9 | 9 | 9 | 8 | 9 | 4 | 5 | 8 | 8 | 9 | 9 | 9 | 8 | 9 | 4 | 215 |
| EPSO | 7 | 7 | 9 | 9 | 7 | 9 | 9 | 7 | 3 | 2 | 9 | 7 | 7 | 5 | 7 | 1 | 7 | 3 | 3 | 8 | 7 | 7 | 5 | 6 | 6 | 7 | 3 | 3 | 170 |
| EPSOG | 6 | 6 | 5 | 6 | 6 | 8 | 6 | 6 | 3 | 7 | 6 | 5 | 5 | 8 | 5 | 6 | 6 | 2 | 5 | 9 | 6 | 3 | 8 | 5 | 4 | 6 | 2 | 6 | 156 |
| DE | 9 | 9 | 7 | 8 | 9 | 5 | 7 | 8 | 9 | 8 | 8 | 9 | 9 | 7 | 8 | 8 | 9 | 8 | 6 | 7 | 9 | 9 | 7 | 8 | 8 | 9 | 8 | 5 | 221 |
| LSHADE44b | 3 | 3 | 3 | 3 | 4 | 6 | 4 | 4 | 8 | 3 | 3 | 4 | 3 | 6 | 3 | 5 | 4 | 1 | 1 | 4 | 3 | 4 | 6 | 2 | 7 | 4 | 1 | 2 | 104 |

A seguir serão apresentados os resultados dos algoritmos sobre o benchmark, que foram computados e deram origem as tabelas de pontuação são mostrados anteriormente. As tabelas são apresentadas na ordem em que a pontuação é calculada:

- Tabela de taxa de viabilidade: percentual das soluções válidas nas 25 execuções;
- Tabela de valor de restrição violada: apresenta os valores que foram violados, no resultado médio das 25 execuções e no resultado mediano;
- Tabela de valor de função objetivo: apresenta o valor da função objetivo, último critério de desempate entre os três critérios avaliados,

Serão apresentadas na sequência as 3 tabelas de valores médios (taxa de viabilidade, valores de restrições violadas e valores das funções objetivos) e as 2 tabelas de valores medianos (valores de restrições violadas e valores das funções objetivos,

Por conveniência serão apresentados somente os resultados de execuções em 10 dimensões, Os demais resultados são apresentados somente computados nas matrizes de pontuação,

Tabela B.9: Taxa de viabilidade para os algoritmos comparados sobre o *benchmark* CEC-2017 para 10 dimensões. Representa o percentual de soluções viáveis finais entre as 25 execuções, para as 29 funções.

| Taxa de Viabilidade – 10 Dimensões | | | | | | | |
|---|---|---|---|---|---|---|---|
| | C01 | C02 | C03 | C04 | C05 | C06 | C07 |
| CAL-SHADE | **100,0** | **100,0** | 44,0 | **100,0** | **100,0** | 96,0 | 68,0 |
| LSHADE44IDE | **100,0** | **100,0** | **100,0** | **100,0** | **100,0** | 0,0 | 80,0 |
| LSHADE44 | **100,0** | **100,0** | 92,0 | **100,0** | **100,0** | 0,0 | 88,0 |
| UDE | **100,0** | **100,0** | **100,0** | **100,0** | **100,0** | 96,0 | **96,0** |
| SPSO | **100,0** | **100,0** | 76,0 | **100,0** | **100,0** | 0,0 | 0,0 |
| EPSO | **100,0** | **100,0** | 92,0 | 60,0 | **100,0** | 0,0 | 0,0 |
| EPSOG | **100,0** | **100,0** | 57,1 | **100,0** | **100,0** | **100,0** | 14,3 |
| DE | **100,0** | **100,0** | 32,0 | **100,0** | **100,0** | 0,0 | 20,0 |
| LSHADE44b | **100,0** | **100,0** | **100,0** | **100,0** | **100,0** | 6,7 | 60,0 |
| | C08 | C09 | C10 | C11 | C12 | C13 | C14 |
| CAL-SHADE | **100,0** | **100,0** | **100,0** | **100,0** | **100,0** | **100,0** | **100,0** |
| LSHADE44IDE | **100,0** | **100,0** | **100,0** | **100,0** | **100,0** | **100,0** | **100,0** |
| LSHADE44 | **100,0** | **100,0** | **100,0** | **100,0** | **100,0** | **100,0** | **100,0** |
| UDE | **100,0** | **100,0** | **100,0** | 0,0 | **100,0** | **100,0** | **100,0** |
| SPSO | 0,0 | **100,0** | 0,0 | 0,0 | 84,0 | 96,0 | 8,0 |
| EPSO | **100,0** | **100,0** | **100,0** | 0,0 | 0,0 | **100,0** | 0,0 |
| EPSOG | **100,0** | **100,0** | **100,0** | 85,7 | **100,0** | **100,0** | **100,0** |
| DE | 4,0 | **100,0** | 68,0 | 0,0 | **100,0** | 92,0 | 80,0 |
| LSHADE44b | **100,0** | **100,0** | **100,0** | **100,0** | **100,0** | **100,0** | **100,0** |
| | C15 | C16 | C17 | C18 | C19 | C20 | C21 |
| CAL-SHADE | 68,0 | 60,0 | **0,0** | 0,0 | **0,0** | **100,0** | **100,0** |
| LSHADE44IDE | **100,0** | **100,0** | **0,0** | 0,0 | **0,0** | **100,0** | **100,0** |
| LSHADE44 | 48,0 | **100,0** | **0,0** | 0,0 | **0,0** | **100,0** | **100,0** |
| UDE | 88,0 | **100,0** | **0,0** | 0,0 | **0,0** | **100,0** | **100,0** |
| SPSO | **100,0** | **100,0** | **0,0** | 0,0 | **0,0** | **100,0** | 20,0 |
| EPSO | 0,0 | **100,0** | **0,0** | 0,0 | **0,0** | 52,0 | 0,0 |
| EPSOG | 71,4 | 42,9 | **0,0** | 85,7 | **0,0** | **100,0** | **100,0** |
| DE | 96,0 | **100,0** | **0,0** | 84,0 | **0,0** | **100,0** | **100,0** |
| LSHADE44b | **100,0** | **100,0** | **0,0** | **100,0** | **0,0** | **100,0** | **100,0** |
| | C22 | C23 | C24 | C25 | C26 | C27 | C28 |
| CAL-SHADE | **100,0** | **100,0** | 40,0 | 60,0 | **0,0** | 0,0 | **0,0** |
| LSHADE44IDE | **100,0** | **100,0** | **100,0** | **100,0** | **0,0** | 0,0 | **0,0** |
| LSHADE44 | **100,0** | **100,0** | **100,0** | **100,0** | **0,0** | 0,0 | **0,0** |
| UDE | **100,0** | **100,0** | **100,0** | **100,0** | **0,0** | 0,0 | **0,0** |
| SPSO | 96,0 | 0,0 | 32,0 | **100,0** | **0,0** | 0,0 | **0,0** |
| EPSO | 64,0 | 0,0 | 0,0 | **100,0** | **0,0** | 0,0 | **0,0** |
| EPSOG | **100,0** | **100,0** | **100,0** | **100,0** | **0,0** | 71,4 | **0,0** |
| DE | 40,0 | 60,0 | **100,0** | **100,0** | **0,0** | 40,0 | **0,0** |
| LSHADE44b | **100,0** | **100,0** | **100,0** | **100,0** | **0,0** | **100,0** | **0,0** |

Tabela B.10: Média do valor violado em restrições entre 25 execuções para os algoritmos comparados sobre o *benchmark* CEC-2017 para 10 dimensões. Representa o percentual de soluções viáveis finais entre as 25 execuções, para as 29 funções.

| Valor das Restrições Violadas – 10 Dimensões | | | | | | | |
|---|---|---|---|---|---|---|---|
| | C01 | C02 | C03 | C04 | C05 | C06 | C07 |
| CAL-SHADE | **0,0000** | **0,000000** | 0,000634 | **0,000000** | **0,000000** | 0,005366 | 0,003091 |
| LSHADE44IDE | **0,0000** | **0,000000** | **0,000000** | **0,000000** | **0,000000** | 0,037664 | 0,000032 |
| LSHADE44 | **0,0000** | **0,000000** | 0,000007 | **0,000000** | **0,000000** | 0,032309 | 0,000021 |
| UDE | **0,0000** | **0,000000** | **0,00000** | **0,000000** | **0,000000** | 0,006804 | 0,001216 |
| SPSO | **0,0000** | **0,000000** | 0,561068 | **0,000000** | **0,000000** | 0,913305 | 55,99834 |
| EPSO | **0,0000** | **0,000000** | 0,000005 | 0,201651 | **0,000000** | 2,644621 | 759,2003 |
| EPSOG | **0,0000** | **0,000000** | 0,000243 | **0,000000** | **0,000000** | 0,000008 | 0,028626 |
| DE | **0,0000** | **0,000000** | 0,000712 | **0,000000** | **0,000000** | 0,237333 | 0,003642 |
| LSHADE44b | **0,0000** | **0,000000** | **0,000000** | **0,000000** | **0,000000** | 0,167604 | 0,000019 |
| | C08 | C09 | C10 | C11 | C12 | C13 | C14 |
| CAL-SHADE | 0,00001 | 0,000004 | 0,000004 | 0,000020 | **0,000000** | **0,000000** | **0,000000** |
| LSHADE44IDE | **0,00000** | **0,000000** | **0,00000** | **0,000000** | **0,000000** | **0,000000** | **0,000000** |
| LSHADE44 | **0,00000** | **0,000000** | **0,00000** | **0,000000** | **0,000000** | **0,000000** | **0,000000** |
| UDE | **0,00000** | **0,000000** | **0,00000** | 0,002997 | **0,000000** | **0,000000** | **0,000000** |
| SPSO | 146,921 | **0,000000** | 152,0522 | 2308,922 | 0,657810 | 1,807834 | 1,400856 |
| EPSO | **0,00000** | **0,000000** | **0,00000** | 9311,556 | 0,629357 | **0,000000** | 2,500000 |
| EPSOG | **0,00000** | **0,000000** | **0,00000** | 0,005417 | **0,000000** | **0,000000** | **0,000000** |
| DE | 0,00265 | 0,000004 | 0,00003 | 1,725681 | **0,000000** | 4,516868 | 2,075582 |
| LSHADE44b | **0,00000** | **0,000000** | **0,00000** | **0,000000** | **0,000000** | **0,000000** | **0,000000** |
| | C15 | C16 | C17 | C18 | C19 | C20 | C21 |
| CAL-SHADE | 15,07050 | 0,012999 | 5,544800 | 140366000 | **4422,390** | **0,000000** | **0,000** |
| LSHADE44IDE | **0,000000** | **0,000000** | 5,220013 | 8226249,4 | 6633,592 | **0,000000** | **0,000** |
| LSHADE44 | 0,000092 | **0,000000** | 5,300000 | 2378710,0 | 6633,590 | **0,000000** | **0,000** |
| UDE | 0,000096 | **0,000000** | 5,380000 | 8288,5 | 6633,600 | **0,000000** | **0,000** |
| SPSO | 0,000050 | 0,000498 | 28,0085 | 214,3759 | 4430,730 | **0,000000** | 5,757 |
| EPSO | 0,497899 | **0,000000** | 25,5000 | 32,8340 | 4426,815 | 0,046135 | 0,185 |
| EPSOG | 0,000051 | 0,000082 | 5,509009 | 286,3399 | 4438,243 | **0,000000** | **0,000** |
| DE | 0,000007 | 0,000005 | 6,859114 | 5,123860 | 4422,395 | **0,000000** | **0,000** |
| LSHADE44b | **0,000000** | **0,000000** | **5,100000** | **0,000000** | 4422,395 | **0,000000** | **0,000** |
| | C22 | C23 | C24 | C25 | C26 | C27 | C28 |
| CAL-SHADE | **0,000000** | 0,000012 | 0,100907 | 0,012243 | 5,504990 | 12600000 | 4436,25 |
| LSHADE44IDE | **0,000000** | **0,000000** | **0,000000** | **0,000000** | 5,052675 | 40476062 | 6640,89 |
| LSHADE44 | **0,000000** | **0,000000** | **0,000000** | **0,000000** | 5,060070 | 412724 | 6650,23 |
| UDE | **0,000000** | **0,000000** | **0,000000** | **0,000000** | 5,420300 | 10457,0 | 6638,80 |
| SPSO | 1,337076 | 2,337397 | 2468,599 | 0,000049 | 272,4987 | 1755,1 | 4442,77 |
| EPSO | 0,056522 | 2,500000 | 0,492005 | **0,000000** | 25,5000 | 837,7 | 4440,12 |
| EPSOG | **0,000000** | **0,000000** | 0,000003 | 0,000007 | 5,507047 | 7140,3 | 4441,52 |
| DE | 18,82755 | 137,09212 | 0,000002 | 0,000007 | 14,7583 | 12586,3 | 4436,52 |
| LSHADE44b | **0,000000** | **0,000000** | **0,000000** | **0,000000** | **4,566667** | **0,0** | **4429,18** |

Tabela B.11: Média do Valor das funções objetivos entre 25 execuções para os algoritmos comparados sobre o *benchmark* CEC-2017 para 10 dimensões. Representa o percentual de soluções viáveis finais entre as 25 execuções, para as 29 funções.

| Valor da função objetivo – 10 Dimensões | | | | | | | |
|---|---|---|---|---|---|---|---|
| | C01 | C02 | C03 | C04 | C05 | C06 | C07 |
| CAL-SHADE | **0,0000** | **0,0000** | 110008,0 | 38,7380000 | 0,95677900 | 549,617000 | -48,73520 |
| LSHADE44I | **0,0000** | **0,0000** | 325939,3 | 14,4175115 | **0,00000000** | 808,360535 | -34,00317 |
| LSHADE44 | **0,0000** | **0,0000** | 31548,20 | 13,6147000 | **0,00000000** | 648,899000 | 3,743620 |
| UDE | **0,0000** | **0,0000** | 73,1230 | 24,4210000 | 0,15946000 | 86,668000 | -71,38700 |
| SPSO | **0,0000** | **0,0000** | 66,8165 | 15,9192438 | 0,00025356 | 21,380007 | -442,9027 |
| EPSO | **0,0000** | **0,0000** | **0,0000** | **0,0000000** | 0,00000191 | **1,989918** | -524,7416 |
| EPSOG | **0,0000** | **0,0000** | 24490,85 | 71,6365611 | 0,00000063 | 122,235970 | **-11225,74** |
| DE | 0,0003 | **0,0000** | 6868,558 | 13,5727872 | 0,05432304 | 201,616470 | -127,6013 |
| LSHADE44b | **0,0000** | **0,0000** | 21045,74 | 13,5727872 | **0,00000000** | 195,097090 | -90,99716 |
| | C08 | C09 | C10 | C11 | C12 | C13 | C14 |
| CAL-SHADE | -0,0013 | 0,1255 | -0,0005 | -0,15626600 | 3,98790000 | 1,11624000 | 2,640690 |
| LSHADE44I | -0,0013 | -0,0050 | -0,0005 | -0,16881943 | 3,98790246 | **0,00000000** | 3,000064 |
| LSHADE44 | -0,0013 | -0,0050 | -0,0005 | -0,16881900 | 3,99316000 | **0,00000000** | 2,881900 |
| UDE | -0,0013 | -0,0050 | -0,0005 | -5,99560000 | 3,98790000 | 0,31893000 | 2,470800 |
| SPSO | **-2,6112** | -0,6092 | **-1,3937** | -841,244544 | 3,98790248 | 0,00007278 | **0,000000** |
| EPSO | 0,0000 | -0,6092 | 0,0000 | **-895,025319** | **0,00000000** | 0,00020033 | **0,000000** |
| EPSOG | 0,0011 | **-50,0982** | 0,0005 | -0,090757 | 3,98790273 | 0,00000030 | 3,100151 |
| DE | 0,0051 | 2,7985 | 0,0005 | -37,468828 | 3,98798957 | 0,00033460 | 2,455433 |
| LSHADE44b | -0,0013 | -0,0050 | -0,0005 | -0,168819 | 3,98790246 | **0,00000000** | 2,376332 |
| | C15 | C16 | C17 | C18 | C19 | C20 | C21 |
| CAL-SHADE | 16,2454 | 65,4166 | 1,0001 | 1348,61000 | 0,00000344 | 0,86861400 | 6,862780 |
| LSHADE44I | 11,2783 | 40,4009 | 0,8824 | 3166,31106 | 0,00000000 | 0,41560643 | 3,987902 |
| LSHADE44 | 14,5456 | 40,7150 | 0,9111 | 2105,76000 | 0,00000145 | 0,19298000 | 3,988110 |
| UDE | 6,6288 | 6,0946 | 0,9491 | 99,6200000 | 0,00000000 | 1,49700000 | 4,412500 |
| SPSO | -7,0687 | 1,5707 | 0,0074 | 1,0000000 | **-9,02929766** | 0,51786445 | 3,987903 |
| EPSO | **-9,8706** | **0,0000** | **0,0000** | **0,0000000** | 0,00000000 | 0,85178808 | **2,984877** |
| EPSOG | 8,6394 | 56,5484 | 0,7945 | 36,6761144 | 16,4578980 | 2,29141600 | 3,987903 |
| DE | 5,4978 | 51,8363 | 0,9438 | 10,0000000 | 0,00000363 | 0,51870180 | 3,987921 |
| LSHADE44b | -3,9271 | 26,7034 | 0,2433 | 36,5976810 | 0,00000127 | **0,14544356** | 3,987902 |
| | C22 | C23 | C24 | C25 | C26 | C27 | C28 |
| CAL-SHADE | 6,6684 | 2,5279 | 15,7361 | 61,1432000 | 1,00191000 | 2438,80000 | 31,12620 |
| LSHADE44I | 0,1595 | 3,0212 | 8,7650 | 37,6990518 | 0,93737670 | 7944,15233 | 10,76297 |
| LSHADE44 | 0,6379 | 2,5411 | 8,6393 | 38,7043000 | 1,05620000 | 3546,97000 | 19,69530 |
| UDE | 5,3135 | 2,4682 | 5,4977 | 6,2830000 | 0,97031000 | 136,300000 | 4,90380 |
| SPSO | 0,2312 | **0,0000** | **-26,0720** | 6,2830461 | 0,02956203 | 5,00000102 | 2,08673 |
| EPSO | 1,4801 | **0,0000** | -9,9485 | **0,0000000** | **0,00000000** | **3,00000000** | 15,29428 |
| EPSOG | **0,0000** | 3,2103 | 8,6394 | 50,2654828 | 0,95153728 | 36,6803514 | 30,37095 |
| DE | 2,2477 | 3,4492 | 2,3562 | 58,1194534 | 0,94288147 | 31,9272313 | 11,83368 |
| LSHADE44b | **0,0000** | 2,3763 | -0,7855 | 26,7033983 | 0,17846381 | 36,5976810 | **0,000001** |

Tabela B.12: Valor violado em restrições da solução mediana entre 25 execuções para os algoritmos comparados sobre o *benchmark* CEC-2017 para 10 dimensões. Representa o percentual de soluções viáveis finais entre as 25 execuções, para as 29 funções.

| Valor Violado das Restrições – 10 Dimensões | | | | | | | |
|---|---|---|---|---|---|---|---|
| | C01 | C02 | C03 | C04 | C05 | C06 | C07 |
| CALSHADE | **0,0** | **0,0** | 0,00010300 | **0,0** | **0,0** | **0,0** | **0,0000** |
| LSHADE44I | **0,0** | **0,0** | **0,0** | **0,0** | **0,0** | 0,03831125 | **0,0000** |
| LSHADE44 | **0,0** | **0,0** | **0,0** | **0,0** | **0,0** | 0,02970210 | **0,0000** |
| UDE | **0,0** | **0,0** | **0,0** | **0,0** | **0,0** | **0,0** | **0,0000** |
| SPSO | **0,0** | **0,0** | 0,00004991 | **0,0** | **0,0** | 0,2034 | 51,7318 |
| EPSO | **0,0** | **0,0** | **0,0** | 0,32671217 | **0,0** | 4,66314355 | 734,003 |
| EPSOG | **0,0** | **0,0** | 0,00006333 | **0,0** | **0,0** | 0,00001041 | 0,0090 |
| DE | **0,0** | **0,0** | 0,00188721 | **0,0** | **0,0** | 0,36146438 | 0,0063 |
| LSHADE44b | **0,0** | **0,0** | **0,0** | **0,0** | **0,0** | 0,18391868 | **0,0000** |
| | C08 | C09 | C10 | C11 | C12 | C13 | C14 |
| CALSHADE | **0,0** | **0,0** | **0,0** | **0,0** | **0,0** | **0,0** | **0,0000** |
| LSHADE44I | **0,0** | **0,0** | **0,0** | **0,0** | **0,0** | **0,0** | **0,0000** |
| LSHADE44 | **0,0** | **0,0** | **0,0** | **0,0** | **0,0** | **0,0** | **0,0000** |
| UDE | **0,0** | **0,0** | **0,0** | 0,00299690 | **0,0** | **0,0** | **0,0000** |
| SPSO | 133,7882 | **0,0** | 110,379431 | 1638,93198 | **0,0** | **0,0** | 1,5000 |
| EPSO | **0,0** | **0,0** | **0,0** | 9235,36396 | 0,67002757 | **0,0** | 2,5000 |
| EPSOG | 0,000008 | **0,0** | 0,00000376 | 0,00000929 | **0,0** | **0,0** | **0,0000** |
| DE | 0,000727 | 0,000001 | 0,00001517 | 0,25207808 | **0,0** | **0,0** | **0,0000** |
| LSHADE44b | **0,0** | **0,0** | **0,0** | **0,0** | **0,0** | **0,0** | **0,0000** |
| | C15 | C16 | C17 | C18 | C19 | C20 | C21 |
| CALSHADE | **0,0** | 0,000051 | 5,50118500 | 971839,453 | **4422,39477** | **0,0** | **0,0000** |
| LSHADE44I | **0,0** | **0,0** | **5,50000000** | 82329,9075 | 6633,59216 | **0,0** | **0,0000** |
| LSHADE44 | 0,000057 | **0,0** | **5,50000000** | 3326,34000 | 6633,59000 | **0,0** | **0,0000** |
| UDE | **0,0** | **0,0** | **5,50000000** | 5453,10000 | 6633,60000 | **0,0** | **0,0000** |
| SPSO | 0,00005 | 0,000050 | 19,9201902 | 114,242113 | 4430,60613 | **0,0** | 5,5962 |
| EPSO | 0,670724 | **0,0** | 25,5000000 | 0,33333333 | 4425,72791 | **0,0** | 0,0067 |
| EPSOG | 0,000161 | 0,000001 | 5,50029637 | 0,00000357 | 4435,72811 | **0,0** | **0,0000** |
| DE | 0,000001 | 0,000007 | 5,50001879 | 0,00002452 | **4422,39477** | **0,0** | **0,0000** |
| LSHADE44b | **0,0** | **0,0** | **5,50000000** | **0,0** | **4422,39477** | **0,0** | **0,0000** |
| | C22 | C23 | C24 | C25 | C26 | C27 | C28 |
| CALSHADE | **0,0** | **0,0** | 0,00468000 | **0,0** | 5,50025900 | 1959116,58 | 4438,936 |
| LSHADE44I | **0,0** | **0,0** | **0,0** | **0,0** | 5,50000000 | 368426,738 | 6633,592 |
| LSHADE44 | **0,0** | **0,0** | **0,0** | **0,0** | 5,50000000 | 10300,5000 | 6653,730 |
| UDE | **0,0** | **0,0** | **0,0** | **0,0** | 5,50000000 | 8047,60000 | 6633,600 |
| SPSO | **0,0** | 2,499917 | 0,00005000 | 0,00005000 | 249,808571 | 1484,10373 | 4441,958 |
| EPSO | **0,0** | 2,500000 | 0,24317222 | **0,0** | 25,5000000 | 201,078280 | 4442,379 |
| EPSOG | **0,0** | 0,001 | 0,001 | 0,005 | 5,50048692 | 0,00000478 | 4442,394 |
| DE | 10,3206 | 0,006 | 0,00000134 | 0,00000294 | 5,50931970 | 0,50064815 | 4435,729 |
| LSHADE44b | **0,0** | **0,0** | **0,0** | **0,0** | **4,50000000** | **0,0** | **4429,061** |

Tabela B.13: Valor da função objetivo da solução mediana entre 25 execuções para os algoritmos comparados sobre o *benchmark* CEC-2017 para 10 dimensões. Representa o percentual de soluções viáveis finais entre as 25 execuções, para as 29 funções.

| Valor da função objetivo – 10 Dimensões | | | | | | | |
|---|---|---|---|---|---|---|---|
| | C01 | C02 | C03 | C04 | C05 | C06 | C07 |
| CALSHADE | **0,0** | **0,0** | 40103,1993 | 35,818324 | **0,0** | 307,643490 | -65,209283 |
| LSHADE44I | **0,0** | **0,0** | 225970,278 | **13,572787** | **0,0** | 1750,59508 | -26,777582 |
| LSHADE44 | **0,0** | **0,0** | 21144,4000 | 13,585300 | **0,0** | 1368,85000 | 12,781500 |
| UDE | **0,0** | **0,0** | 74,6840 | 18,904000 | **0,0** | 76,915000 | -69,855000 |
| SPSO | **0,0** | **0,0** | 77,1970 | 26,863763 | 2,486985 | **75,616147** | -399,13783 |
| EPSO | **0,0** | **0,0** | **0,0000** | 96,612287 | 0,679712 | 128,418969 | -422,67507 |
| EPSOG | **0,0** | **0,0** | 49028,6412 | 137,48189 | 0,094234 | 136,650937 | **-454,46000** |
| DE | 0,0284 | 0,013610 | 57726,9853 | 15,919244 | 4,478763 | 475,711863 | 23,75268 |
| LSHADE44b | **0,0** | **0,0** | 58912,3527 | **13,572787** | **0,0** | 632,683281 | 27,15918 |
| | C08 | C09 | C10 | C11 | C12 | C13 | C14 |
| CALSHADE | -0,0013 | -0,004975 | -0,00051000 | -0,168819 | 3,987902 | **0,00000000** | 2,62881600 |
| LSHADE44I | -0,0013 | -0,004975 | -0,00050965 | -0,168819 | 3,987902 | **0,00000000** | 3,01258627 |
| LSHADE44 | -0,0013 | -0,004975 | -0,00050965 | -0,168819 | 3,993570 | **0,00000000** | 2,90533000 |
| UDE | -0,0013 | -0,004975 | -0,00050965 | -5,995600 | 3,987900 | **0,00000000** | 2,37630000 |
| SPSO | **-1,5782** | -0,609172 | **-0,68550023** | -763,44052 | 3,987903 | 0,00612783 | 1,22574117 |
| EPSO | 0,0 | -0,609172 | 0,00000000 | **-893,70129** | **1,989918** | 1,63470508 | **0,00000000** |
| EPSOG | 0,002 | **-21,09611** | 0,00103371 | -0,013836 | 3,987903 | 0,02612843 | 3,51723263 |
| DE | 0,0314 | 11,56778 | 0,00175425 | 0,463970 | 3,988604 | 0,18933101 | 3,61262169 |
| LSHADE44b | -0,0013 | -0,004975 | -0,00050965 | -0,168819 | 3,987902 | **0,00000000** | 2,37633241 |
| | C15 | C16 | C17 | C18 | C19 | C20 | C21 |
| CALSHADE | 14,922 | 64,40275 | 1,00991800 | 704,5000 | 0,000003 | 0,75764600 | 3,98790200 |
| LSHADE44I | 11,780 | 39,27004 | 0,78195307 | 3338,985 | 0,000000 | 0,40117271 | 3,98790246 |
| LSHADE44 | 14,922 | 39,27000 | 0,83544900 | 2925,750 | 0,000001 | **0,19079100** | 3,98790000 |
| UDE | 5,4977 | 6,283000 | 1,00300000 | 53,53300 | 0,000000 | 1,37590000 | 3,98790000 |
| SPSO | **-3,9270** | 6,283046 | 0,01724107 | 2,000000 | **-8,107387** | 0,72679809 | 20,894120 |
| EPSO | -2,0340 | **0,0** | **0,00000000** | **0,0** | 7,826896 | 1,60556340 | **3,97983623** |
| EPSOG | 11,781 | 64,40265 | 1,00913002 | 40,165325 | 32,000875 | 2,32210082 | 3,98790345 |
| DE | 11,780 | 70,68581 | 1,00975355 | 53,727292 | 0,000007 | 0,73267496 | 3,98880263 |
| LSHADE44b | 5,4977 | 43,98216 | 1,02304569 | 36,597703 | 0,000002 | 0,20833456 | 3,98790246 |
| | C22 | C23 | C24 | C25 | C26 | C27 | C28 |
| CALSHADE | **0,0** | 2,628819 | 14,9224940 | 62,831714 | 1,008974 | 1020,3896 | 31,511898 |
| LSHADE44I | **0,0** | 3,061671 | 8,63934578 | 39,269850 | 0,943628 | 3763,2500 | **0,00000000** |
| LSHADE44 | **0,0** | 2,402770 | 8,63931000 | 39,269800 | 0,584145 | 1844,5000 | 27,566700 |
| UDE | **0,0** | 2,376300 | 5,49770000 | 6,283000 | 1,010000 | 77,500000 | **0,00000000** |
| SPSO | 4,1551 | 1,155154 | **-3,92706153** | 12,566231 | 0,151445 | **14,000000** | 11,62032857 |
| EPSO | 2,9253 | **0,0** | -3,57592645 | **0,0** | **0,0** | 30,000000 | 38,28268255 |
| EPSOG | 0,1528 | 3,481121 | 14,9225651 | 62,831853 | 1,008708 | 38,121239 | 55,12769010 |
| DE | 471719,5 | 3,962233 | 11,7809743 | 70,685843 | 1,011060 | 43,353975 | 28,45428111 |
| LSHADE44b | **0,0** | 2,376332 | 2,35612378 | 37,698973 | 1,309860 | 36,597697 | 19,85369409 |