UNIVERSIDADE FEDERAL DO PARANÁ

HELLEN CRISTINA SPENGLER

ALGORITMO DE EVOLUÇÃO DIFERENCIAL COM PARÂMETROS AUTO-ADAPTATIVOS APLICADO AO PROBLEMA DE ALOCAÇÃO DE CONFIABILIDADE-REDUNDÂNCIA

CURITIBA

2017

HELLEN CRISTINA SPENGLER

ALGORITMO DE EVOLUÇÃO DIFERENCIAL COM PARÂMETROS AUTO-ADAPTATIVOS APLICADO AO PROBLEMA DE ALOCAÇÃO DE CONFIABILIDADE-REDUNDÂNCIA

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Ciências, no Curso de Pós-Graduação em Métodos Numéricos em Engenharia, Setores de Tecnologia e Ciências Exatas, da Universidade Federal do Paraná.

Orientador: Prof. Dr. Gustavo Valentim Loch.

CURITIBA

2017

O48 Spengler, Hellen Cristina

Algoritmo de evolução diferencial com parâmetros auto-adaptativos aplicado ao problema de alocação de confiabilidade-redundância / Hellen Cristina. – Curitiba, 2017.
65 f.: il.; tabs. : color. : 30 cm.

Dissertação (mestrado) - Setores de Tecnologia e Ciências Exatas, da Universidade Federal do Paraná, Programa de Pós-Graduação em Métodos Numéricos em Engenharia.

Orientador: Prof. Dr. Gustavo Valentim Loch
Bibliografia: p.61-65.

1. Método numérico. 2. Engenharia. I. Universidade Federal do Paraná. II. Loch, Gustavo Valentim. III. Título.

CDD 519.4

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DO PARANÁ
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
Setor TECNOLOGIA
Programa de Pós-Graduação MÉTODOS NUMÉRICOS EM ENGENHARIA

ATA Nº

ATA DE SESSÃO PÚBLICA DE DEFESA DE MESTRADO PARA A OBTENÇÃO DO GRAU DE MESTRE EM MÉTODOS NUMÉRICOS EM ENGENHARIA

No dia quatorze de Fevereiro de dois mil e dezessete às 09:00 horas, na sala Auditório, Centro de Estudos em Engenharia Civil - CESEC/UFPR, campus Centro Politécnico, foram instalados os trabalhos de arguição da mestranda **HELLEN CRISTINA SPENGLER** para a Defesa Pública de sua dissertação intitulada **Algoritmo de Evolução Diferencial Com Parâmetros Auto-Adaptativos Aplicado Ao Problema de Alocação de Confiabilidade-Redundância**. A Banca Examinadora, designada pelo Colegiado do Programa de Pós-Graduação em MÉTODOS NUMÉRICOS EM ENGENHARIA da Universidade Federal do Paraná, foi constituída pelos seguintes Membros: GUSTAVO VALENTIM LOCH (UFPR), ROBERTO ZANETTI FREIRE (PUC/PR), ROY WILHELM PROBST (UTFPR), JOSÉ EDUARDO PÉCORA JUNIOR (UFPR). Dando início à sessão, a presidência passou a palavra a discente, para que a mesma expusesse seu trabalho aos presentes. Em seguida, a presidência passou a palavra a cada um dos Examinadores, para suas respectivas arguições. A aluna respondeu a cada um dos arguidores. A presidência retomou a palavra para suas considerações finais e, depois, solicitou que os presentes e a mestranda deixassem a sala. A Banca Examinadora, então, reuniu-se sigilosamente e, após a discussão de suas avaliações, decidiu-se pela APROVAÇÃO da aluna. A mestranda foi convidada a ingressar novamente na sala, bem como os demais assistentes, após o que a presidência fez a leitura do Parecer da Banca Examinadora. Nada mais havendo a tratar a presidência deu por encerrada a sessão, da qual eu, GUSTAVO VALENTIM LOCH, lavrei a presente ata, que vai assinada por mim e pelos membros da Comissão Examinadora.

Curitiba, 14 de Fevereiro de 2017.

GUSTAVO VALENTIM LOCH
Presidente da Banca Examinadora (UFPR)

ROBERTO ZANETTI FREIRE (Via Skype)
Avaliador Externo (PUC/PR)

ROY WILHELM PROBST
Avaliador Externo (UTFPR)

JOSÉ EDUARDO PÉCORA JUNIOR
Avaliador Externo (UFPR)

A Deus, meus pais, familiares e amigos, por todo incentivo e carinho dedicados a mim.

AGRADECIMENTOS

Agradeço primeiramente a Deus e todos os céus, pela vida repleta de oportunidades e grande animo que me ofertaram.

Aos meus pais, Marcos e Sandra, por toda preocupação, tempo e dedicação doados com tanto amor.

Aos meus familiares, pelo apoio e conselhos vindos sempre em boa hora. Em especial ao meu irmão Alan.

Ao meu orientador, professor Gustavo Valentim Loch, por toda paciência em me guiar nesse caminho.

Ao Grupo de tecnologia Aplicada à Otimização - GtAO - pelo acolhimento, estudos e ambiente feliz que me proporcionaram.

Aos professores do PPGMNE, que contribuíram direta ou indiretamente com seus conhecimentos e experiências. Também ao secretário Jair, pela disposição em esclarecer tantas dúvidas.

Aos meus amigos e colegas, por compartilharem tempo, sabedoria e alegrias comigo. Em especial, a Tulipa e ao Rômulo, pelas conversas e estudos divididos.

A Capes, pelo apoio financeiro durante todo o mestrado.

E a todos que participaram, cada um a sua maneira, nessa jornada.

Lavoisier

"Nada se perde, tudo muda de dono" - tardia reflexão de Lavoisier ao descobrir que lhe haviam roubado a carteira.

(MÁRIO QUINTANA, 1987, p.728)

RESUMO

Os problemas de otimização combinatória, atualmente, recebem grande atenção nas pesquisas acadêmicas e aplicações. As primeiras pesquisas, a partir da década de 1940, limitavam-se a resolução de problemas de programação linear, mas a evolução computacional nas décadas seguintes expandiu os estudos para problemas inteiros e também para não-lineares. Dentre os problemas não-lineares inteiro-misto, destaca-se o problema de alocação de confiabilidade-redundância. As características deste problematornam sua resolução difícil, mesmo para pequenas instâncias. Logo, durante as últimas décadas foram propostas diversas técnicas para sua resolução, muitas dessas utilizando e adaptando abordagens de meta-heurísticas evolutivas. Nesse grupo, destaca-se o algoritmo de Evolução Diferencial, por sua simplicidade e bom desempenho. Porém, ainda que eficaz, apresenta algumas deficiências, em relação a existência de parâmetros fixados inicialmente em seus operadores. Assim, se tornando uma tarefa extra a escolha desses parâmetros, sendo diferente a cada problema. Nesse âmbito, foram apresentadas propostas de usar algoritmos que auto-adaptassem os parâmetros. Neste trabalho toma-se como foco o algoritmo de Evolução Diferencial com Mutação Auto-adaptativa (SaMDE). O algoritmo SaMDE foi proposto recentemente em (SILVA, 2010), e até então não tem muitas aplicações divulgadas na literatura. É de conhecimento que sua convergência embora eficaz, precisa ser melhorada. Logo, a presente dissertação propõe uma versão híbrida com o algoritmo de Busca por Enxame de Partículas (PSO), para agregar vantagens ao processo de busca do SaMDE, com o objetivo de melhorar o desempenho e a convergência do algoritmo SaMDE para o problema de alocação de confiabilidade-redundância. Tanto a versão híbrida quanto o SaMDE foram aplicados a exemplos clássicos da literatura para o problema de alocação de confiabilidade-redundância, sendo para o sistema em série, série-paralelo e complexo. A versão híbrida proposta apresentou melhora, comparada ao algoritmo SaMDE, no caso de problemas em série e complexo. Ambas obtiveram convergência próxima aos melhores resultados conhecidos na literatura.

Palavras-chave: Evolução Diferencial, Auto-adaptação, Alocação de redundância-confiabilidade.

ABSTRACT

Combinatorial optimization problems are currently receiving great attention in academic research and applications. The first researches, from the 1940s, were limited to solving linear programming problems, but computational evolution in the following decades expanded the studies to whole problems as well as to non-linear ones. Among the non-linear integer-mixed problems, the problem of reliability-redundancy allocation is highlighted. The characteristics of this problem make its resolution difficult, even for small instances. Therefore, during the last decades several techniques have been proposed for their resolution, many of them using and adapting approaches of evolutionary metaheuristics. In this group, the Differential Evolution algorithm stands out for its simplicity and good performance. However, although effective, it presents some deficiencies, in relation to the existence of parameters to be initially set in its operators. Thus, it becomes an extra task to choose these parameters, being different to each problem. In this context, the proposal came up to use algorithms that would auto-adapt the parameters. This work focuses on the algorithm of Differential Evolution with Auto-adaptive Mutation (SaMDE). The SaMDE algorithm was recently proposed in (Silva, 2010), and until then has not many applications published in the literature. It is well known that its convergence, although effective, needs to be improvedTherefore, this work proposes a hybrid version with the Particle Swarm Search algorithm (PSO), to add advantages to the search process of SaMDE, with the objective of improving the performance and convergence of the SaMDE algorithm for the allocation problem of Reliability-redundancy. Both the hybrid version and the SaMDE were applied to classic examples of the literature for the problem of reliability-redundancy allocation, being for serial, parallel-series and complex systems. The proposed hybrid version presented improvement, compared to the SaMDE algorithm, in the case in series and complex. Both obtained convergence close to the best results known in the literature.

Key-words: Differential Evolution, Self-adaptation, Allocation of redundancy-reliability.

LISTA DE SÍMBOLOS E ABREVIAÇÕES

RBD - Método de Diagrama de Bloco de Confiabilidade
DE - Algoritmo de Evolução Difencial.
jDE - Auto-adaptação para o Ajuste de Parâmetros na Evolução Diferencial.
SADE - Evolução Diferencial Auto-adaptativa.
DESAP - Evolução Diferencial com População Auto-adaptativa.
SaMDE - Evolução Diferencial com Mutação Auto-adaptativa.
PSO - Otimização por Enxame de Partículas.
GA - Algoritmo Genético.
IA - Algoritmo Imune.
IAs- Algoritmo Imune Artificial.
ES - Estratégia da Águia
HS - Busca Harmônica.
CS - Busca Cuckoo.
TS - Busca Tabu.
ABC - Colônia de Abelhas Artificial.
BBO - Algoritmo de otimização biogeográfica.
SSO - Otimização Simplificada de Partículas.
PSSO - Otimização Simplificada Baseada em Partículas.
NAFSA - Novo Algoritmo Artificial de Enxame de Peixes.
PSFS - Pesquisa fractal estocástica guiada por penalidade.

LISTA DE NOTAÇÕES

R_s	–	Confiabilidade do sistema
g	–	conjunto de restrições do problema
r	–	vetor da confiabilidade dos componentes do sistema
r_i	–	confiabilidade de cada componente do subsistema i
n	–	vetor dos componentes redundantes do sistema
n_i	–	número de componentes redundantes no subsistema i
f	–	função de otimização do problema
l	–	vetor de recursos do problema
m	–	número de subsistemas
$x(t)$	–	função de entrada
$y(t)$	–	função de saída
t_0	–	Deslocamento
R_i	–	confiabilidade de cada subsistema i
R_{ij}	–	confiabilidade do j-ésimo componente no i-ésimo subsistema, com $i = 1, \dots, m$ e $j = 1, \dots, n_i$
R_{ij}	–	confiabilidade do i-ésimo componente no j-ésimo subsistema, com $j = 1, \dots, n$ e $i = 1, \dots, m_j$
$P(S\|A)$	–	probabilidade do evento S ocorrer condicionado a ocorrencia do evento A
$P(A)$	–	probabilidade do evento A ocorrer
t	–	geração da população nos algoritmos
X_t	–	população na geração t
$x_{t,i}$	–	$x_{t,i} = (x_{t,i,1}, x_{t,i,2}, \dots, x_{t,i,n})$: i-ésima solução na t-ésima geração, em que n é uma terceira variável do problema, do algoritmo DE.
$x_{t,i,j}$	–	Variáveis do problema a serem otimizadas, com $i = 1,2, \dots, np$ e $j = 1,2, \dots, n$.
NP	–	número de soluções (indivíduos) da população
$\bar{r}_k$	–	soluções escolhidas aleatoriamente da população X_t, com $k = 1,2,3$ (DE) ou $k = 1,2,3,4,5$ (SaMDE)
F ou F'	–	parâmetro de fator de escala
V_t	–	população mutante na geração t.
$v_{t,i}$	–	i-ésima solução mutante da t-ésima geração.
U_t	–	população de descendentes na geração t.

$u_{t,i}$	–	i-ésima solução descendente na t-ésima geração
CR	–	parâmetro de recombinação discreta.
δ_i	–	parâmetro que é sorteado aleatoriamente entre os valores de $1,2,\dots,n$.
ρ	–	estratégia escolhida pela roleta.
$x_{t,i}$	–	i-ésima solução na t-ésima geração, do algoritmo SaMDE.
V_i^{ρ}	–	valor de chance de escolha da estratégia ρ, com $\rho = 1,2,3,4$.
F_i^{ρ}, CR_i^{ρ}	–	par de parâmetros F e CR da i-ésima solução da população da estratégia ρ.
V	–	volume máximo.
C	–	custo máximo.
W	–	peso máximo.
$\begin{cases} \alpha_i \\ \beta_i \\ w_i \\ w_i v_i^2 \end{cases}$	–	parâmetros da modelagem das restrições do problema.
v_t^i	–	velocidade da i-ésima partícula na t-ésima geração do algoritmo PSO.
θ	–	parâmetro de inércia.
a_1, a_2	–	valores aleatórios entre 0 e 1.
p^i	–	melhor posição encontrada pela i-ésima partícula.
p_t^s	–	melhor posição dentre todas as partículas na iteração t.
c_1	–	parâmetro de confiança em si mesmo.
c_2	–	parâmetro de confiança na população.
λ	–	variável de valor aleatório 0 ou 1.
$rand$	–	valor aleatório, com distribuição uniforme, no intervalo $[0,1]$
$\boldsymbol{vizinho_{u,k}}$	–	u-ésimo vizinho em relação a solução $\bar{r}_k$, em que $u = 1,2,\dots,10$ e $k = 1,2,\dots,5$

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

SUMÁRIO

INTRODUÇÃO

A crescente aplicação da otimização matemática está associada à necessidade de melhores respostas para diversos problemas reais apresentados pelas empresas. Um dos problemas é a maximização da confiabilidade de um produto ou processo, sujeito às restrições como custo, peso, volume, quantidade de pessoas envolvidas, entre outras.

A formalização e valorização da confiabilidade acompanham o desenvolvimento da tecnologia. Em especial, da Segunda Guerra Mundial em diante foi dada maior atenção ao problema, consequentemente gerando demanda pelo desenvolvimento e aprimoramentode técnicas de avaliação e compreensão (KUO, PRASAD, 2000).

Uma definição para a confiabilidade, amplamente aceita e utilizada, foi proposta por (HALPERN, 1978): confiabilidade é a probabilidade de um sistema, equipamento ou componente, submetido a condições previamente estabelecidas, desempenhe as funções especificadas no projeto, durante um período de tempo também especificado.

Na produção, seja de produtos, sistemas ou estruturas, a confiabilidade influencia diretamente na rentabilidade e segurança. Projetos de itens confiáveis levam a processos mais robustos e necessitam de menos intervenções futuras do fabricante, de vultosos custos.

Houveram grandes tragédias provocadas por ignorar tais fatores. Em 1963, a explosão do submarino nuclear Thresher matou 129 pessoas. Testes apontaram que os tripulantes ultrapassaram em mais de trinta por cento da profundidade máxima, provocando o colapso da estrutura. Em 1986, os quatro novos reatores nucleares em Chernobyl explodiram, causandotrinta mortes, duzentos casos de contágio radioativo crônico, grandes áreas contaminadas ainda atualmente e perdas na ordem de três bilhões de dólares, sendo considerado um dos piores acidentes comerciais da história (PEREIRA, 2004). Portanto, a confiabilidade deve ser considerada necessária e importante para a indústria, tornando-se um fator a ser considerado no planejamento e fabricação.

No estudo de sistemas, a modelagem de seus designs serelaciona diretamente a confiabilidade do sistema. Conjuntamente, outras variáveis também influem sobre o resultado, tais como o custo, o peso e o volume.

Por exemplo, considerando um sistema como na Figura 1. Tal sistema é composto por três subsistemas, cada um com um número de componentes redundantes (n_i), e cada componente com sua confiabilidade por subsistema (r_i).

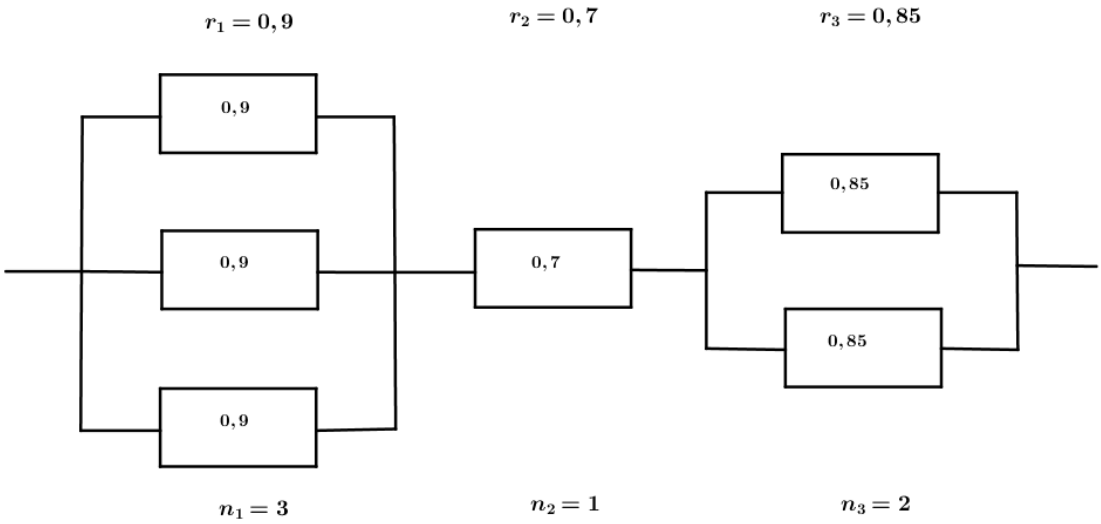

Figura 1: Exemplo de um sistema.

Logo, existem duas variáveis vindas do design do sistema: a confiabilidade de cada componente e o número de componentes redundantes por subsistema. Como notação tem-se $r = (r_1, r_2, \dots, r_m)$o vetor da confiabilidade dos componentes e$n = (n_1, n_2, \dots, n_m)$o vetor dos componentes redundantes, por subsistema (no caso com m subsistemas).

Usualmente maximiza-se a confiabilidade pela função objetivo e limitam-se o custo, o peso e o volume pelas funções de restrições (DINGRA, 1992). A seguir está a formulação do problema.

Maximize $R_s = f(r, n)$
sujeito a $g(r, n) \leq l$
$0 \leq r_i \leq 1, r_i \in \mathbb{R}, n_i \in \mathbb{Z}^+, 1 \leq i \leq m$

Em que R_s é a confiabilidade do sistema, g representa o conjunto de restrições(consideradas usualmente na literatura como peso, volume e custo).

O vetor l representa os recursos do problema e m o número de subsistemas.Ressalta-se ainda que a confiabilidade r_i de cada componente depende do custo c_i.

Buscando pela maximização da confiabilidade desses sistemas, pode-se tomá-los por duas abordagens: acrescer as confiabilidades individuais dos componentes do sistema ou usar componentes redundantes nos subsistemas. Estas abordagens já foram empregadas anteriormente para resolução deste e outros problemas, e apresentaram soluções satisfatórias (CHEN, YOU, 2005), (COIT, SMITH, 1996). Apesar disso ainda possuem limitações: a primeira abordagem leva a poucas opções de solução e ao esgotamento da tecnologia perante o auto nível de confiabilidade exigido pelos componentes;já a segunda abordagem, traz grandes acréscimos na utilização de recursos nas restrições, podendo levar ainfactibilidade.

Na tentativa de contornar tais limitações, unem-se ambas as abordagens, isto é, acrescer a confiabilidade dos componentes do sistema, e usar componentes redundantes nos subsistemas. Assim, têm-se elementos intercambiáveis que trazem novas formas factíveis de melhoria. Tal abordagem chama-se, na literatura, de Problema de Alocação de Confiabilidade-Redundância (ZOULFAGHARI, 2014).

Pela estrutura da abordagem tem-se que existem variáveis inteiras (quantidade de componentes) e reais (confiabilidade dos componentes). Além disso, sua função objetivo é não linear. Logo, pode-se entender por um problema de otimização não linear inteiro-misto.

Para sua resolução, no decorrer de algumas décadas, foram empregados diferentes métodos. Inicialmente utilizaram-se métodos exatos, que apresentaram dificuldades na resolução, pela complexidade do problema, mesmo em instâncias pequenas. Tillman et al. e (TILLMAN et al, 1997) e Gopal et al. (GOPAL et al, 1978) desenvolveram métodos heurísticos para resolver o problema de alocação de confiabilidade-redundância. Kuo et al. (KUO et al, 1987) demonstrou, para o sistema em série de cincosubsistemas, um método otimizador utilizando os métodos de multiplicadores de Lagrange e branch-and-bound. Hikita et al. (HIKITA et al, 1992) elaborou um algoritmo baseado em restrições substitutivas. Porém, essas abordagens exigem derivadas de todas

as funções de restrição, que podem ser não-lineares e não facilmente derivadasem relação a algumas variáveis.

Foi provado por Chern que o problemade alocação de confiabilidade-redundância tem complexidade NP-hard (CHERN, 1992). Diante disso, foram propostas diversas abordagens baseadasna aplicação demeta-heurísticas para resolução, como Algoritmo Genético (TAVAKKOLI-MOGHADDAN et al, 2008) e (MARTORELL et al, 2004), Colônia Artificial de Abelhas (SADJADI, SOLTANI, 2012) e (YEH, HSIEH, 2011), Busca Tabu (OUZINEB et al, 2008), Algoritmo de Sistema Imune Artificial (CHEN, 2006), Otimização por Enxame de Partículas (COELHO, 2009), Algoritmo de Evolução Diferencial (LIU, QIN, 2015), BBO (GARG, 2015) e etc.

Além disso, também existem trabalhos que apresentam combinações de meta-heurísticas já utilizadas, ou seja, versões híbridas: Busca Tabu e Evolução Diferencial (LIU, QIN, 2014), Busca Harmônica e Otimização por Enxame de Partículas (ZOU et al, 2011), Algoritmo Genético e Otimização por Enxame de Partículas (SHEIKHALISHAHI et al, 2013), etc.

Diante do descrito, constata-se que hágrandes esforços para conseguir melhorias na resolução no problema de alocação de confiabilidade-redundância, criando-se novas meta-heurísticas e aprimorando as já existentes.

Dentre os diversos métodos possíveis para resolução de problemas gerais de otimização, o algoritmo de Evolução Diferencial,criado em (STORN, PRICE, 1995), destaca-se entre os algoritmos evolutivos, superando-os pelas suas boas propriedades de convergência (SWAGATAM et al, 2016).

Os Algoritmos Evolutivos formam uma família de meta-heurísticas que imitamo processo de evolução natural (LIU, QIN, 2014), assim como o Algoritmo de Sistema Imune Artificial (CHEN, YOU, 2005), o Algoritmo de Otimização por Enxame de Partículas (COELHO, 2009), o Algoritmo Genético (COIT, 1996) e o Algoritmo de Evolução Diferencial (SILVA, 2010). Uma das vantagens de suas aplicações a diversos problemas está no fato de não necessitarem de informações como diferenciabilidade ou continuidade da função objetivo.

O Algoritmo de Evolução Diferencial tem uma estrutura simples, dado que osoperadores possuem uma formulação matemática baseada na soma e subtração de vetores, mas mantendo a estrutura de um algoritmo evolutivo.

Destaca-se ainda que desde sua criação é largamente aplicado a variados problemas, e tem se mostrado eficaz (SWAGATAM et al, 2016). Porém, ainda que eficaz, possui deficiências. Existem parâmetros a serem fixados inicialmente em seus operadores, que tem influência direta no resultado obtido. Tornando assim, a escolha dos parâmetros uma tarefa extra, podendo ser diferente para cada problema aplicado.

Logo, houve-se a proposta de automatizara escolha dos parâmetros para os algoritmos evolutivos, levando a ideia de auto-adaptação também para a evolução diferencial (SCHWEFEL, 1981). Alguns métodos já foram desenvolvidos, por exemplo: Auto-adaptação para o Ajuste de Parâmetros na Evolução Diferencial – jDE (BREST, MAUCED, 2006); Evolução Diferencial Auto-adaptativa - SaDE (QIN, SUCANTHAN, 2005); Evolução Diferencial com População Auto-adaptativa - DESAP (TEO, 2006); Evolução Diferencial com Mutação Auto-adaptativa - SaMDE (SILVA, 2010).

O método SaMDE ainda é recente e por isso não possui muitas aplicações divulgadas na literatura. Além disso, sabe-se que sua convergência precisa ser melhorada (SILVA, 2010). Nesse intuito, este trabalho pretende implementar o algoritmo SaMDE ao problema de alocação de confiabilidade-redundância, e estudar formas de melhorias da convergência global. Dentre as estratégias de melhoria, optou-se pela hibridação com algoritmos que agreguem vantagens aos processos de busca do SaMDE, em especial optou-se pelo uso do algoritmo de Busca por Enxame de Partículas (PSO, do inglês *Particle Swarm Optimization*).

1.1 OBJETIVO GERAL

O objetivo geral desta dissertação é a melhoria do desempenho de convergência do algoritmo SaMDE para o problema de alocação de confiabilidade-redundância.

1.2 OBJETIVOS ESPECÍFICOS

A fim de alcançar o objetivo geral, a seguir são listados os objetivos específicos do trabalho:

- Realizar levantamento na literatura das diferentes abordagens do algoritmo de Evolução Diferencial e melhorias propostas.
- Identificar os métodos de auto-adaptação para o algoritmo de Evolução Diferencial propostos na literatura.
- Propor uma versão híbrida para o algoritmo SaMDE, utilizando o algoritmo PSO.
- Implementar o método SaMDE e sua versão híbrida para o problema de alocação de confiabilidade-redundância.
- Verificar os desempenhos dos algoritmosSaMDE e a versão hibrida proposta, para o problema de alocação de confiabilidade-redundância.
- Comparar as soluções obtidas pelo algoritmo proposto com as já conhecidas pela literatura.

1.3 ESTRUTURA DO TRABALHO

O presente trabalho está estruturado em cinco capítulos. O primeiro capítulo introduz o tema do trabalho, e traz seus objetivos. No segundo capítulo faz-se uma breve revisão de literatura sobre os principais conceitos do trabalho: sistemas, Algoritmo de Evolução Diferencial (DE) e Algoritmo de Evolução Diferencial com Mutação Auto-adaptativa (SaMDE).

No capítulo três, apresenta-se a heurística híbrida proposta para o algoritmo SaMDE e PSO. No quarto capítulo, são especificados os exemplos de problema de alocação de confiabilidade-redundância a serem aplicados, bem como são exibidos os resultados para a implementação do algoritmo SaMDE e de sua versão híbrida proposta. Além disso, são comparados aos resultados já conhecidos na literatura.

Ao final, no quinto capítulo são realizadas as considerações finais e sugestões para trabalhos futuros.

2 REVISÃO DE LITERATURA

No presentecapítulo éapresentada uma breve revisão de literatura do Problema de Alocação de Confiabilidade-Redundância, assim como das meta-heurísticas de Evolução Diferencial (DE, do inglês *DifferentialEvolution*) e Evolução Diferencial com Mutação Auto-adaptativa (SaMDE, do inglês *Self AdaptationMutationDifferentialEvolution*). Tais conceitos constituema base para o entendimento e desenvolvimento destadissertação.

2.1 O PROBLEMA DE ALOCAÇÃO DE CONFIABILIDADE-REDUNDÂNCIA

No estudo de sistemas, a modelagem de seus designs serelaciona diretamente com a confiabilidade do sistema. Conjuntamente, outras variáveis também influem sobre o resultado, tais como o custo, o peso e o volume. Usualmente maximiza-se a confiabilidade pela função objetivo e limitam-seo custo, o peso e o volume pelas funções de restrições(DINGRA, 1992).

Buscando pela maximização da confiabilidade desses sistemas, pode-se tomá-los por duas abordagens: acrescer a confiabilidade dos componentes do sistema ou usar componentes redundantes nos subsistemas. Uma abordagem eficiente, por trazer elementos intercambiáveis, é a união das duas abordagens anteriores, que se chama Problema de Alocação de Confiabilidade-Redundância (COELHO, 2009).

Nesta estrutura há variáveis inteiras (componentes redundantes) e reais (confiabilidade dos componentes). Além disso, suas restrições são não lineares. Logo, pode-se entender por um problema de otimização não linear inteiro-misto. Sua formulação está a seguir.

Maximize $R_s = f(r, n)$
sujeito a $g(r, n) \leq l$
$0 \leq r_i \leq 1, r_i \in \mathbb{R}, n_i \in \mathbb{Z}^+, 1 \leq i \leq m.$

Em que R_s é a confiabilidade do sistema, g representa o conjunto de restrições(usualmente interpretadas como peso, volume e custo),$r = (r_1, r_2, \ldots, r_m)$o vetor da confiabilidade dos componentes do sistema e$n =$

$(n_1, n_2, \dots, n_m)$o vetor dos componentes redundantes do sistema. Logo, r_i é a confiabilidade de cada componente no subsistema i e n_i é o número de componentes redundantes no subsistema i. O vetor l representa os recursos do problema e m o número de subsistemas.

2.2 SISTEMAS E SUAS CONFIGURAÇÕES

SegundoINCOSE (2006), um sistema é um conjunto integrado de componentes projetado para cumprir um objetivo (INCOSE, 2006). Existem numerosas variáveis associadas na análise de um sistema: reparável ou não reparável, variante com o tempo, e outros.

As seguintes definições desta seção são referentes ao trabalho de (SALGADO, 2008).

Nesse trabalho, a aplicação se dará a sistemas que não integram atributos dinâmicos. Isto é, são sistemas não reparáveis e invariantes com o tempo. Issofacilita a modelagem matemática da função de confiabilidade do sistema, fazendo-se pelo método de Diagrama de Bloco de Confiabilidade (RBD, do inglês *ReabilityBlockDiagram*). Sendo, dentre os métodos conhecidos, o de manejo mais simples.

Um sistema reparável é aquele apto a manutenção caso ocorra falha. Já um sistema não-reparável, não pode ser reparado por manutenção, a não ser por substituição ou descarte do componente que apresentou falha. A maioria dos sistemas reais são reparáveis, tais como maquinários, aparelhos eletrônicos, automóveis, e etc.

Já um sistema é invariante com o tempo se um deslocamento $t_0$na entrada $x(t)$, provoca o mesmo deslocamento na saída$y(t)$, não importando quando é aplicado este deslocamento. Ou melhor:

$$x(t) \rightarrow y(t) \implies x(t - t_0) \longrightarrow y(t - t_0).$$

Caso contrário, considera-se o sistema variante com o tempo. Porém, na realidade nenhum sistema atende este critério, considerando-se apenas uma aproximação desta característica.

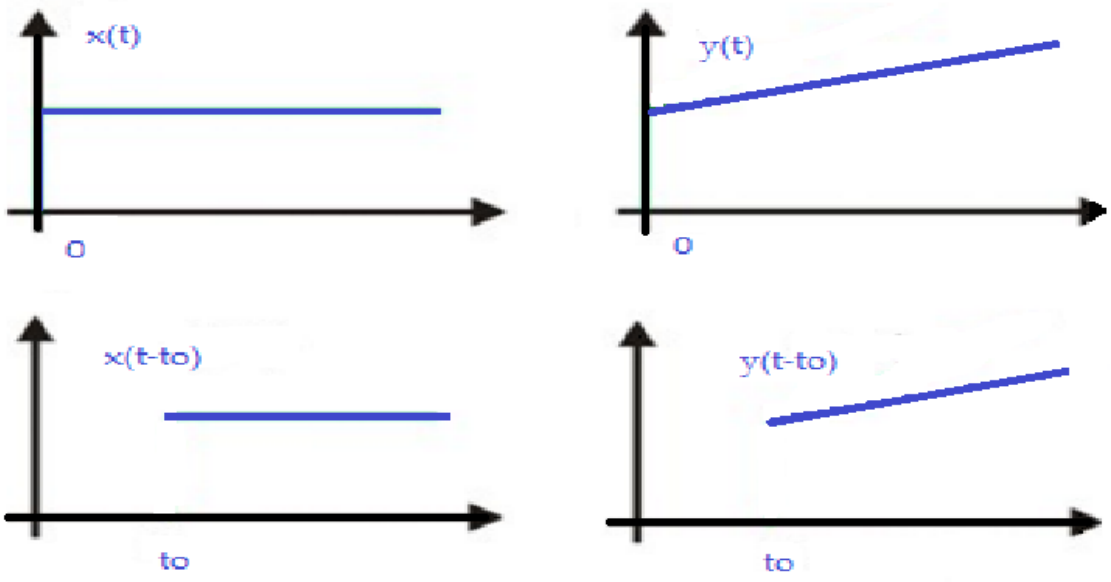

Figura 2: Representação de invariância no sistema, em relação ao deslocamento t_0.

As configurações mais comuns para um sistema são em série, paralelo, paralelo-série, série-paralelo e complexo, conforme explicações a seguir.

Sistema em série: Um sistema é definido como *série*, se a falha de ao menos um componente leva a falha do sistema inteiro, isto é, todos os componentes devem estar funcionando para que o sistema também funcione.

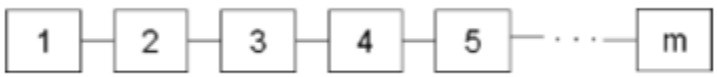

Figura 3: Representação de um sistema em série.

Sobre tais hipóteses, pode-se calcular a confiabilidade do sistema. Inicialmente considera-se que os $m$componentes são mutuamente independentes. Logo, a função é dada pelo produto da confiabilidade de cada componente, ou similarmente de cada subsistema, representando-os por R_i.

$$R_s = \prod_{i=1}^{m} R_i \qquad (1)$$

Sistemas em paralelo: Um sistema é definido como *paralelo* se, e somente se, a falha de todos os componentes ocasiona a falha do sistema.

Assim, caso ao menos um componente esteja funcionando, então o sistema também estará.

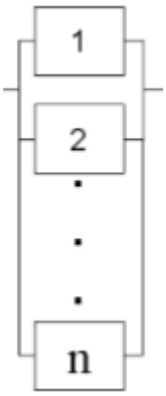

Figura 4: Representação de um sistema em paralelo.

Supondo n componentes mutuamente independentes, a confiabilidade pode ser calculada pelo complemento da probabilidade de falha do sistema, que por consequência é o produto da probabilidade de falha de todos os componentes, ou subsistemas.

$$R_s = 1 - \prod_{i=1}^{n}(1 - R_i) \tag{2}$$

Sistema em série-paralelo:um sistema *série-paralelo*é composto por m subsistemas em série e $n_i(i = 1, \dots, n)$ componentes em paralelo nos m subsistemas, conforme a figura abaixo.

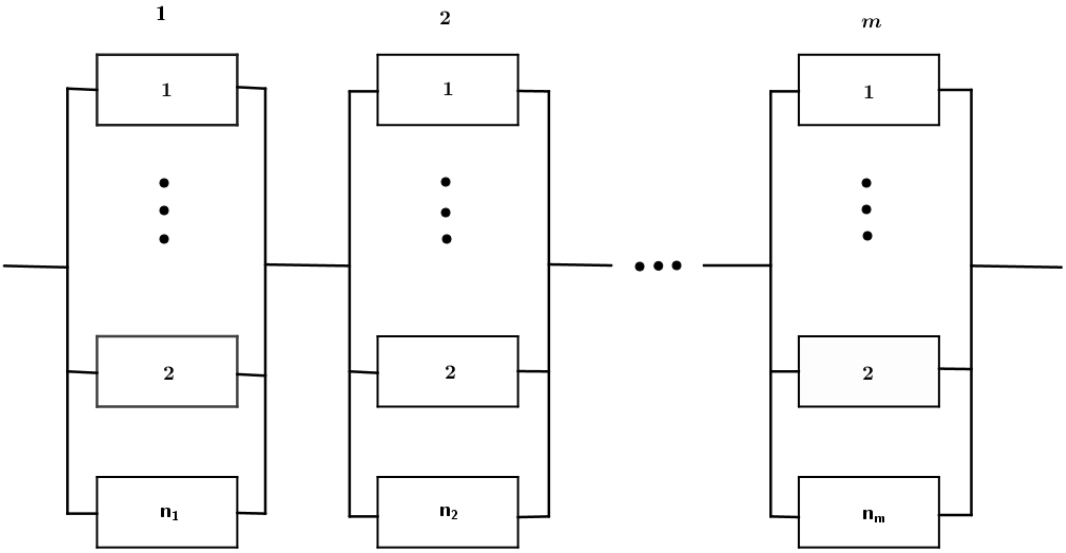

Figura 5: Representação de um sistema em série-paralelo.

Como visto anteriormente, a confiabilidade de cada subsistema i pode ser calculada pela confiabilidade de um sistema em paralelo com n_i componentes.

$$R_i = 1 - \prod_{j=1}^{n_i} \left(1 - R_{ij}\right) \tag{3}$$

Em que R_{ij} é a confiabilidade do j-ésimo componente no i-ésimo subsistema, com$i = 1, \dots, m$ e $j = 1, \dots, n_i$. Após a redução dos blocos dos subsistemas, pode-se avaliar o sistema como em série e calcular sua confiabilidade.

$$R_s = \prod_{i=1}^{m} \left(1 - \prod_{j=1}^{n_i} R_{ij}\right) \tag{4}$$

De mesmo modo, outra possível combinação dessas estruturas é o sistema em *paralelo-série*. Composto por n subsistemas em paralelo, com $m_j (j = 1, \dots, n)$ componentes para cada sistema j.

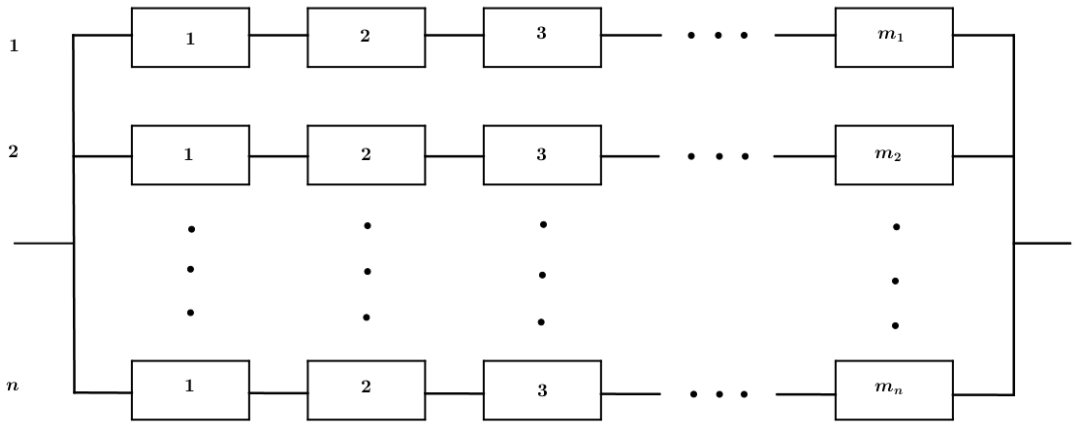

Figura 6: Representação de um sistema em paralelo-série.

Para calcular a confiabilidade do sistema, primeiramente deduz-se a confiabilidade de cada j-ésimo subsistema em série, assumindo que as m_j componentes sejam mutuamente independentes.

$$R_j = \prod_{i=1}^{m_j} \left(1 - R_{ij}\right) \tag{5}$$

Em que R_{ij} é a confiabilidade do i-ésimo componente no j-ésimo subsistema, com $j = 1, \dots, n$ e $i = 1, \dots, m_j$. Agora, tomando os n subsistemas em paralelo, faz-se o cálculo da confiabilidade do sistema.

$$R_s = 1 - \prod_{j=1}^{n} \left(1 - \prod_{i=1}^{m_j} R_{ij} \right) \tag{6}$$

Sistema complexo: um sistema *complexo* é formado por componentes e subsistemas que interagem e possibilitam certas características, que não são explicadas pelos seus componentes.

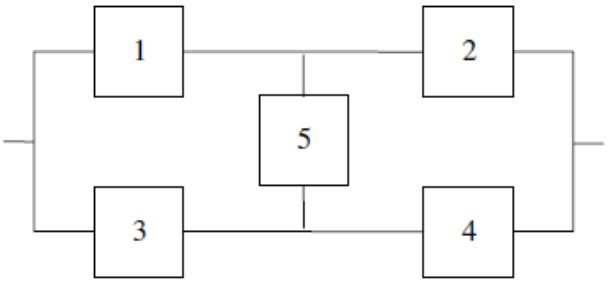

Figura 7: Representação de um sistema complexo.

Seu formato e cálculo de sua confiabilidade variam de sistema para sistema. Por exemplo, para a configuração apresentada na Figura 7, é usadoa redução por diagrama de bloco. Primeiramente devemos escolher um componente para supor 100% confiável, e depois reconsiderar o caso de falha desse componente. Assim, escolhe-se a componente 5.

Quando o evento A ocorre, tem-se que a componente 5 é suposta 100% confiável,então levando em consideração o uso dessa componente e sua posição, obtendo a configuração do sistema conforme a Figura 8. Caso contrário, ocorrendo o evento $\overline{A}$, em que a componente 5 apresenta falha, tem-se que não é considerado seu uso, necessitando percorrer os caminhos alternativos, conforme o sistema da Figura 9.

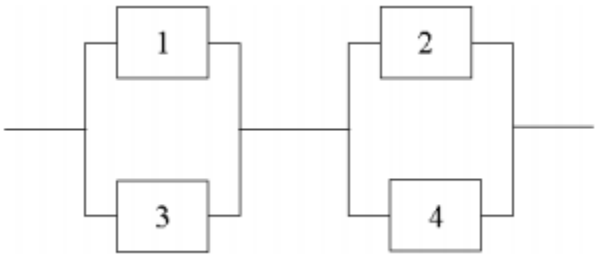

Figura 8: Sistema complexo, quando a componente 5 não tem falha.

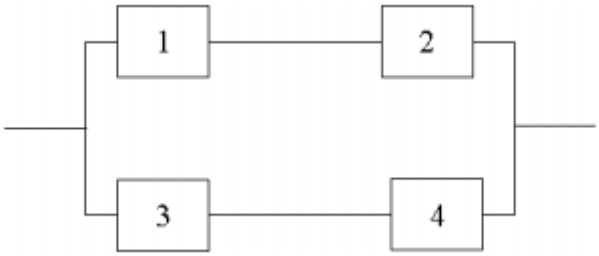

Figura 9: Sistema complexo, quando a componente 5 tem falha.

Logo, sendo S o evento de sucesso do sistema, pelo Teorema de Bayesa confiabilidade do sistema ao todo é dada por:

$$R_s = P(S|A)P(A) + P\left(S|\overline{A}\right)P\left(\overline{A}\right) \tag{7}$$

E os cálculos para cada parte:

$$P(S|A)P(A) = \left\{\left[\left(1-(1-R_1)(1-R_3)\right)\left(1-(1-R_2)(1-R_4)\right)\right]R_5\right\} \tag{8}$$

$$P\left(S|\overline{A}\right)P\left(\overline{A}\right) = \{[1-(1-R_1R_3)(1-R_2R_4)](1-R_5)\} \tag{9}$$

Finalmente, substituindo as equações 8 e 9 na equação 7, e fazendo as operações necessárias obtém-se a confiabilidade geral do sistema:

$$\begin{aligned} R_s = {} & R_1R_2 + R_3R_4 + R_1R_4R_5 + R_2R_3R_5 - R_1R_2R_3R_4 - R_1R_2R_3R_5 \\ & - R_1R_2R_4R_5 - R_1R_3R_4R_5 - R_2R_3R_4R_5 + 2R_1R_2R_3R_4R_5 \end{aligned} \tag{10}$$

2.3 A EVOLUÇÃO DIFERENCIAL

O algoritmo de Evolução Diferencial (DE, do inglês *DifferentialEvolution*), proposto em 1995 por Storn e Price, faz parte do

conjunto de Algoritmos Evolutivos (EA, do inglês EvolutionaryAlgorithms). Tal conjunto possui o aspecto da adaptação baseada nas leis de evolução de Darwin (SILVA, 2010), em que os indivíduos mais aptos (portadores de melhores características) têm maior probabilidade de sobreviver na população para a próxima geração.

Ele dispõe de importantes atributos, por exemplo, não requerer que a função objetivo seja contínua ou diferenciável, ser adaptável a variáveis contínuas e discretas, e ainda as suas combinações. No entanto, o que o destaca mesmo entre os algoritmos evolutivos, é a mutação por diferenças de vetores (os indivíduos da população). Sendo um processo simples, que se baseia apenas em um parâmetro e que é responsável pelas mudanças de toda a população a cada geração (EIBEN et al, 1999).

Inicialmente o algoritmo de evolução diferencial foi aplicado para resolução de sistemas de variáveis contínuas (STORN, PRICE, 1997) e logo se mostrou eficiente a uma grande gama de problemas, tanto na otimização mono objetivo (MEZURA-MONTES et al, 2006), quanto na otimização multi-objetivo, formulada primeiramente por (CHANG et al, 1999).

Até hoje as variações do DE mostram bons resultados nas competições sobre os EAs organizadas pela IEEE CongressonEvolutionaryComputation (CEC), assim se demonstrando um algoritmo competitivo atualmente (SWAGATAM et al, 2016).

2.3.1 O algoritmo de Evolução Diferencial

De acordo com a versão original, considere uma população de soluções iniciais, geradas aleatoriamente do espaço de busca, $X_t = \{x_{t,i}; i = 1, \dots, NP\}$, em que$x_{t,i}$ representa o i-ésimo indivíduo da t-ésima geração eNP o número de indivíduos da população. Logo, um indivíduo será descrito como:

$$x_{t,i} = (x_{t,i,1}, x_{t,i,2}, \dots, x_{t,i,n}) \tag{11}$$

Em que n se refere a uma terceira variável que representa o número de variáveis do problema.

Como mencionado previamente, o algoritmo utiliza um processo de busca constituído na diferença entre duas ($x_{t,\bar{r}_2}$ e $x_{t,\bar{r}_3}$, para notação adotada) soluções escolhidas aleatoriamente dentre as presentes na população. O resultado é adicionado a uma terceira solução ($x_{t,\bar{r}_1}$).

$$v_{t,i} = x_{t,\bar{r}_1} + F\left(x_{t,\bar{r}_2} - x_{t,\bar{r}_2}\right) \tag{12}$$

Em que $\bar{r}_1 \neq \bar{r}_2 \neq \bar{r}_3$, e $F \in [0{,}8; 1]$ é o fator de escala multiplicado pelo vetor diferença, $v_{t,i}$ é a i-ésima solução mutante da t-ésima geração e $x_{t,\bar{r}_1}$ é o vetor base no qual se aplica a mutação diferencial.

Assim, obtém-se uma população mutante $V_t = \{v_{t,i}; i = 1{,}2, \dots, NP\}$, que será recombinada com a população corrente X_t, formando a população de descendentes, chamada de U_t.

$$u_{t,i,j} = \begin{cases} v_{t,i,j}, & se\ U_{[0,1]} \leq CR\ \vee j = \delta_i \\ x_{t,i,j}, & caso\ contrário. \end{cases} \tag{13}$$

Em que CR é o parâmetro de recombinação discreta, tal que $CR \in [0{,}1]$. E $\delta_i \in 1, \dots, n$ é sorteado aleatoriamente, garantindo que ao menos uma variável da solução teste passe para a solução descendente. Observa-se que o parâmetro CR é capaz de administrar a porcentagem de $v_{t,i}$ herdada por $u_{t,i}$. Além disso, destaca-se que há um vetor mutante, $u_{t,i,j}$, para cada indivíduo $x_{t,i,j}$ da população corrente. Logo ambas as populações, X_t e U_t, possuem o mesmo número de indivíduos.

Após gerada a população U_t,é realizada uma seleção entre as soluções das populações X_t e U_t, avaliando o valor da função objetivo de $u_{t,i}$ e sua correspondente $x_{t,i}$,sendo aprovada a com melhor valor. A solução aprovada é classificada para compor a população corrente da próxima geração, X_{t+1}. Neste caso, representa-se por um problema de maximização.

$$x_{t+1,i} = \begin{cases} u_{t,i}, & se\ f(u_{t,i}) > f(x_{t,i}), \\ x_{t,i}, & caso\ contrário. \end{cases} \quad (14)$$

	Pseudocódigo – Algoritmo DE
1:	$t \leftarrow 1$
2:	Inicializar a população $X_t = (x_{t,i}; i = 1,2, \dots, NP)$
3:	Avaliar X_t
4:	**While**condição_de_parada**do**
5:	**For** $i = 1:NP$**do**
6:	Selecione aleatoriamente $\bar{r}_1, \bar{r}_2, \bar{r}_3 \in 1, \dots, NP$
7:	Selecione aleatoriamente $\delta_i \in 1, \dots, n$
8:	Selecione aleatoriamente $F \in [0.8, 1]$
9:	**For** $j = 1:n$**do**
10:	**If**$U_{[0,1]} \leq CR \vee j = \delta_i$**then**
11:	$u_{t,i,j} \leftarrow x_{t,\bar{r}_1,j} + F(x_{t,\bar{r}_2,j} - x_{t,\bar{r}_3,j})$
12:	**Else**
13:	$u_{t,i,j} \leftarrow x_{t,i,j}$
14:	**EndIf**
15:	**End For**
16:	**End For**
17:	**For**$i = 1:NP$**do**
18:	**If**$f(u_{t,i}) > f(x_{t,i})$**then**
19:	$x_{t+1,i} \leftarrow u_{t,i}$
20:	**Else**
21:	$x_{t+1,i} \leftarrow x_{t,i}$
22:	**EndIf**
23:	**End For**
24:	$t \leftarrow t + 1$
25:	**EndWhile**

Quadro 1: Pseudocódigo do algoritmo de evolução diferencial.

Note que, as soluções da população inicial devem estar espalhadas de tal forma a cobrirem o espaço de busca a fim de evitar que grandes áreas do espaço de busca não possuam nenhuma solução na população. Existem estudos e métodos específicos para resolução mais aprimorada dessa característica, como em (TEO, 2006).

Ressalta-se ainda que as perturbações causadas pela diferença entre x_{t,r_2} e x_{t,r_3} são proporcionais as posições destas soluções no espaço de busca. Além disso, nas primeiras gerações estas diferenças serão maiores do que nas gerações futuras, que já apresentam elevada convergência. Assim, quanto maior a convergência, menor será a distância entre as soluções da população,

dado que se concentraram em torno das vizinhanças da melhor solução encontrada.

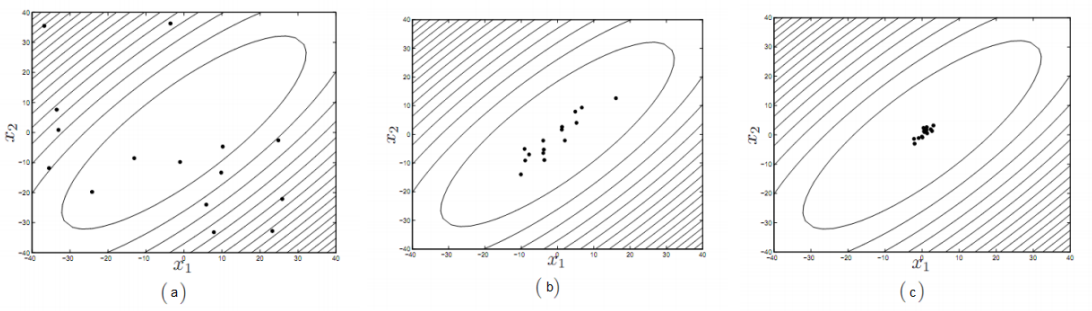

Figura 10: Função quadrática (GUIMARÃES, 2009). Distribuição espacial da população nas gerações $t = 1$ (a), $t = 10$ (b) e $t = 20$ (c).

Como citado anteriormente, o algoritmo de evolução diferencial possui algumas deficiências, seja pelo número limitado de combinações do espaço de busca ou pelosparâmetros fixos(EIBEN, 1999).Assim, a auto-adaptação dos parâmetros traz uma solução a estas deficiências, proporcionando maior número de combinações do espaço de busca. Além disso, também facilita a adaptação dos parâmetros a cada diferente problema, evitando o desperdício de tempo e trabalho nas atividades de determinar quais os melhores parâmetros para cada problema a ser tratado. Diversos métodos já foram desenvolvidos utilizando estes parâmetros como base, tais como o jDE, SaDE, DESAP e SaMDE (SILVA, 2010).

2.4 O ALGORITMO DE EVOLUÇÃO DIFERENCIAL COM MUTAÇÃO AUTO-ADAPTATIVA

O algoritmo de Evolução Diferencial com Mutação Auto-Adaptativa (SaMDE,do inglês Self-adaptiveMutationDiferentialEvolution) proposto primeiramente por Silva, faz a auto-adaptação dos parâmetros para múltiplas estratégias de mutação. Issopermite o uso dessas estratégias em diferentes momentos, tornando-o um algoritmo que maximiza seu desempenho comparado aos algoritmosde uma única estratégia (SILVA, 2010).

Os parâmetros auto-adaptados, no algoritmo SaMDE, são F e CR, para as múltiplas estratégias de mutação, utilizadasjá no algoritmo SaDE e também no SaMDE:

- **rand/1** – Estratégia original do algoritmo DE, possui convergência lente e alta capacidade de exploração.

$$u_{t,i,j} \leftarrow x_{t,\bar{r}_1,j} + F_i^{\rho}\left(x_{t,\bar{r}_2,j} - x_{t,\bar{r}_3,j}\right) \quad (15)$$

- **best/1** – Estratégia de convergência rápida e bom comportamento com funções unimodais.

$$u_{t,i,j} \leftarrow x_{t,best,j} + F_i^{\rho}\left(x_{t,\bar{r}_1,j} - x_{t,\bar{r}_2,j}\right) \quad (16)$$

- **rand/2** – Estratégia que possibilita uma perturbação diferente, dado que a distribuição estatística da soma de todos os pares de vetores possui a forma de sino.

$$u_{t,i,j} \leftarrow x_{t,\bar{r}_1,j} + F_i^{\rho}\left(x_{t,\bar{r}_2,j} - x_{t,\bar{r}_3,j}\right) + F_i^{\rho}\left(x_{t,\bar{r}_4,j} - x_{t,\bar{r}_5,j}\right) \quad (17)$$

- **current-to-rand/1** – Estratégia invariante a rotação e que apresentou bom desempenho em problemas multi-objetivos.

$$u_{t,i,j} \leftarrow x_{t,i,j} + F_i^{\rho}\left(x_{t,\bar{r}_1,j} - x_{t,i,j}\right) + F_i^{\rho}\left(x_{t,\bar{r}_2,j} - x_{t,\bar{r}_3,j}\right) \quad (18)$$

Assim, uma solução no algoritmo SaMDE terá a seguinte forma:

$$x_{t,i} = \left(x_{t,i,1}, \dots, x_{t,i,n}, V_i^1, V_i^2, V_i^3, V_i^4, F_i^1, CR_i^1, F_i^2, CR_i^2, F_i^3, CR_i^3, F_i^4, CR_i^4\right) \quad (19)$$

Em que $x_{t,i,j}$ $(i = 1,2, \dots, np$ e $j = 1, \dots, n)$ são as variáveis a serem otimizadas, $V_i^{\rho} \in [0,1]$ é o valor de chance de uso da estratégia ρ,com $\rho = 1,2,3,4$. E $F_i^{\rho} \in [0,1], CR_i^{\rho} \in [0,1]$ são os pares de parâmetros da estratégia ρ.

As perturbações dos parâmetros são realizadas também pela mutação diferencial. Os parâmetros são inicializados aleatoriamente, levando em consideração os intervalos descritos anteriormente. Assim, cada um dos parâmetros V_i^{ρ} são alterados a cada solução da população.

$$V_i^{\rho} = V_{\bar{r}_1}^{\rho} + F'\left(V_{\bar{r}_2}^{\rho} - V_{\bar{r}_3}^{\rho}\right) \quad (20)$$

Por meio do algoritmo de seleção por roleta, é escolhido um dos valores de V_i^{ρ} atualizados, que representará a estratégia vencedora. Por conseqüência, apenas o par F_i^{ρ} e CR_i^{ρ} referente à estratégia vencedora será atualizado, conforme a seguir:

$$F_i^{\rho} = F_{\bar{r}_1}^{\rho} + F'\left(F_{\bar{r}_2}^{\rho} - F_{\bar{r}_3}^{\rho}\right) \quad (21)$$

$$CR_i^{\rho} = CR_{\bar{r}_1}^{\rho} + F'\left(CR_{\bar{r}_2}^{\rho} - CR_{\bar{r}_3}^{\rho}\right) \quad (22)$$

Em que ρ representa a estratégia vencedora, e F' é escolhido aleatoriamente no intervalo $[0{,}8;\ 1]$. Após atualização, segue a mutação diferencial referente a estratégia escolhida e a seleção, para todos as soluções da população a cada geração, até que o critério de parada estabelecido seja alcançado.

Alguns casos podem apresentar variáveis de otimização ou os parâmetros fora do intervalo estabelecido. Uma abordagem recomendável foi proposta por (ROKKONEN et al.,2005):

$$x = \begin{cases} 2x_{inferior} - x, & se\ x < x_{inferior}\ , \\ 2x_{superior} - x, & se\ x > x_{superior}\ . \end{cases} \quad (23)$$

Em que x é a variável de otimização ou parâmetro, e $x_{inferior}$ e $x_{superior}$ são os limites inferior e superior do intervalo estabelecido.

	Pseudocódigo – Algoritmo SaMDE
1:	$t \leftarrow 1$
2:	Inicializar a população $X_t = \left(x_{t,i}; i = 1{,}2, \dots, NP\right)$
3:	Avaliar X_t
4:	**While**condição_de_parada**do**
5:	**For** $i = 1{:}\,NP$**do**
6:	Selecione aleatoriamente $\bar{r}_1, \bar{r}_3, \dots, \bar{r}_5 \in 1, \dots, NP$
7:	Selecione aleatoriamente $\delta_i \in 1, \dots, n$
8:	Selecione aleatoriamente $F' \in [0.8, 1]$
9:	**For** $\rho = 1{:}\,Número\ de\ estratégias$**do**
10:	$V_i^{\rho} = V_{\bar{r}_1}^{\rho} + F'\left(V_{\bar{r}_2}^{\rho} - V_{\bar{r}_3}^{\rho}\right)$
11:	**End For**
12:	$\rho \leftarrow$Estratégia escolhida pela roleta
13:	$F_i^{\rho} = F_{\bar{r}_1}^{\rho} + F'\left(F_{\bar{r}_2}^{\rho} - F_{\bar{r}_3}^{\rho}\right)$
14:	$CR_i^{\rho} = CR_{\bar{r}_1}^{\rho} + F'\left(CR_{\bar{r}_2}^{\rho} - CR_{\bar{r}_3}^{\rho}\right)$
15:	**For** $j = 1{:}\,n$**do**
16:	**If**$U_{[0,1]} \leq CR_i^{\rho}\ \vee\ j = \delta_i$**then**
17:	**If**$\rho = 1$**then**
18:	$u_{t,i,j} \leftarrow x_{t,\bar{r}_1,j} + F_i^{\rho}\left(x_{t,\bar{r}_2,j} - x_{t,\bar{r}_3,j}\right)$
19:	**ElseIf**$\rho = 2$**then**
20:	$u_{t,i,j} \leftarrow x_{t,best\ ,j} + F_i^{\rho}\left(x_{t,\bar{r}_1,j} - x_{t,\bar{r}_2,j}\right)$

21: **ElseIf** $\rho = 3$ **then**
22: $u_{t,i,j} \leftarrow x_{t,\bar{r}_1,j} + F_i^{\rho}\left(x_{t,\bar{r}_2,j} - x_{t,\bar{r}_3,j}\right) + F_i^{\rho}\left(x_{t,\bar{r}_4,j} - x_{t,\bar{r}_5,j}\right)$
23: **ElseIf** $\rho = 4$ **then**
24: $u_{t,i,j} \leftarrow x_{t,i,j} + F_i^{\rho}\left(x_{t,\bar{r}_1,j} - x_{t,i,j}\right) + F_i^{\rho}\left(x_{t,\bar{r}_2,j} - x_{t,\bar{r}_3,j}\right)$
25: **EndIf**
26: **Else**
27: $u_{t,i,j} \leftarrow x_{t,i,j}$
28: **EndIf**
29: **EndFor**
30: **For** $i = 1{:}\,NP$ **do**
31: **If** $f\left(u_{t,i}\right) \leq f\left(x_{t,i}\right)$ **then**
32: $x_{t+1,i} \leftarrow u_{t,i}$
33: **Else**
34: $x_{t+1,i} \leftarrow x_{t,i}$
35: **EndIf**
36: **End For**
37: **End For**
38: $t \leftarrow t + 1$
39: **EndWhile**

Quadro 2: Pseudocódigo do algoritmo SaMDE.

No entanto, mesmo que apresente rápida convergência, por vezes torna-se prematura, diminuindo sua capacidade de diversificação, e tendo soluções com convergência para ótimos locais. Desse modo, o processo de hibridação pretende trazer maior diversificação, e por consequência melhorar seu desempenho de convergência, permitindo uma busca em âmbito global.

3 HEURÍSTICA HÍBRIDA

No presente capítuloéapresentada a heurística híbrida proposta, bem como detalhes da implementação para o problema de alocação de confiabilidade-redundância.

3.1 SOLUÇÕES NO ALGORITMO SAMDE

O problema de alocação de confiabilidade redundância é um problema classificado como não linear inteiro-misto. A luz disso tem-se que as variáveis que são tratadas no problema possuem forma inteira e real (CHEN, 2006). Mais especificamente, a quantidade de componentes redundantes do sistema compõe as variáveis inteiras, já a confiabilidade dessas componentes integra as variáveis reais. Assim, os componentes redundantes variam no intervalo dos números inteiros, limitados pelo máximo de alocações permitidas por subsistema e o número um. Observa-se que é excluído o número zero, porque que um componente negativo ou nulo não pode ser considerado, dado que não constitui uma solução aceitável, pois caso contrário haveriam subsistemas sem componente e, portanto, tornando o sistema inoperante. Já as variáveis reais, por serem probabilidades (confiabilidades), estão delimitadas entre o intervalo real $[0,1]$. Logo, uma solução no algoritmo SaMDE terá a seguinte forma:

$$x_{t,i} = \left(x_{t,i,1}, \dots, x_{t,i,n}, V_i^1, V_i^2, V_i^3, V_i^4, F_i^1, CR_i^1, F_i^2, CR_i^2, F_i^3, CR_i^3, F_i^4, CR_i^4\right) \quad (24)$$

Em que $x_{t,i,j} (i = 1, 2, \dots, NP$ e $j = 1,2, \dots, n)$ são as variáveis a serem otimizadas (sendo reais e inteiras), $V_i^\rho \in [0,1]$ é o valor de chance de uso da estratégia ρ,com $\rho = 1,2,3,4$. E $F_i^\rho \in [0,1]$, $CR_i^\rho \in [0,1]$ são os pares de parâmetros da estratégia ρ.

Os parâmetros são tratados pelo processo de auto-adaptação do próprio algoritmo. Desse modo, é suficiente apenas inicializá-los aleatoriamente no intervalo condizente.

Além disso, observa-se que o algoritmo SaMDE obtém convergência satisfatória das variáveis inteiras, levando o foco de atenção para as variáveis reais. Lembrando que o intervalo de variação das variáveis

inteiras(especialmente considerando os exemplos clássicos da literatura) é consideravelmente pequeno, comparado com as variáveis reais.

Portanto, as variáveis reais concentram maior desafio e esforço de resolução. Dado que as soluções se apresentam sensíveis a pequenas mudanças nos valores das variáveis, podendolevar a infactibilidade ou também a rápidas mudanças no valor otimizado.

Para exemplificar a sensibilidade das soluções, apresenta-se um exemplo de um sistema em série para o problema de alocação de confiabilidade-redundância. Este problema com esta configuração e restrições foi apresentado inicialmente em (HIKITA et al, 1992), e se tornou um exemplo clássico de aplicação e comparação na literatura. A seguir está a sua função objetivo descrita na Equação 25, as restrições referentes às variáveis de custo, peso e volume, conforme as Equações 26, 27 e 28. E também a estrutura do sistema considerado, na Figura 11.

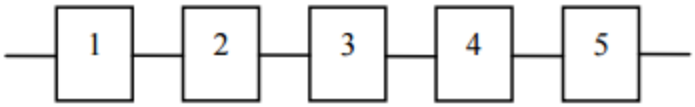

Figura 11: Estrutura do sistema em série do exemplo.

$$f(r,n) = \prod_{i=1}^{5} R_i(n_i) \tag{25}$$

$$g_1(r,n) = \sum_{i=1}^{5} w_i v_i^2 n_i^2 \leq V \tag{26}$$

$$g_2(r,n) = \sum_{i=1}^{5} \alpha_i (-1000/\ln r_i)^{\beta_i} \left(n_i + exp(n_i/4)\right) \leq C \tag{27}$$

$$g_1(r,n) = \sum_{i=1}^{5} w_i n_i exp(n_i/4) \leq W \tag{28}$$

$$0 \leq r_i \leq 1, n_i \in \mathbb{Z}^+$$

Em que R_s é a confiabilidade do sistema, g representa o conjunto de restrições(usualmente interpretadas como peso, volume e custo),$r = (r_1, r_2, \dots, r_m)$o vetor da confiabilidade dos componentes do sistema e$n =$

$(n_1, n_2, \ldots, n_m)$o vetor dos componentes redundantes do sistema. Logo, r_i é a confiabilidade de cada componente no subsistema i e n_i é o número de componentes redundantes no subsistema i. O vetor l representa os recursos do problema e m o número de subsistemas. Os parâmetros α_i, β_i, w_i e $w_i v_i^2$ são valores dados pela modelagem do problema.

A Tabela 1 traz a solução apresentada em (CHEN, 2005) e versões modificadas (destaque em negrito) e o impacto de mudança nos valores de folga e confiabilidade do sistema. Observa-se que em todas as soluções o número de componentesn_i em cada subsistema foi o mesmo. No entanto, uma pequena modificação em apenas uma das confiabilidades r_i, gera diferenças nas folgas das restrições, levando facilmente a infactibilidade como na terceira solução modificada, e também a piora de desempenho da confiabilidade ou melhora, como na primeira e segunda solução modificada, respectivamente. Logo o estudo da presente dissertação se concentrou principlamente em métodos que melhorassem o desempenho de busca das variáveis reais pelo método SaMDE. Dentre os métodos, destacou-se o algoritmo de busca por enxame de partículas, que poderia trazer novos "movimentos" à busca e um processo de diversificação orientada.

	Solução (Chen, 2005)		Soluções modificadas					
	n_i	r_i	n_i	r_i	n_i	r_i	n_i	r_i
	3	0,779266	3	**0,779268**	3	0,779266	3	**0,779268**
	2	0,872513	2	0,872513	2	**0,872512**	2	**0,872516**
	2	0,902634	2	0,902634	2	0,902634	2	0,902634
	3	0,710648	3	0,710648	3	0,710648	3	0,710648
	3	0,788406	3	0,788406	3	0,788406	3	0,788406
Folga g_1	27		27		27		27	
Folga g_2	0,001375968		**0,000908708**		**0,001794727**		**-0,000347622**	
Folga g_3	7,518918241		7,518918241		7,518918241		7,518918241	
R_s	0,931678196		**0,931678472**		**0,931677955**		**0,931679196**	

Tabela 1: Soluções, com suas modificações, referentes a um sistema em série com cinco subsistemas e no máximo cinco alocações.

3.2 MÉTODO DE BUSCA NO ALGORITMO PSO

O Algoritmo de otimização por Enxame de Partículas (PSO, do inglês *Particle Swarm Optimization*) tem uma abordagem populacional baseada na inteligência coletiva. De maneira diferente dos algoritmos evolutivos, a cada partícula é atribuída uma velocidade. Tal velocidade é utilizada para atualizar as posições, levando em conta o melhor valor da função aptidão encontrada pela partícula (conhecimento local) e o melhor valor da função aptidão encontrado pela população (conhecimento global). Tornando-se uma modelagem matemática de quanto a partícula confia em si mesma e quanto ela confia na população, para a tomada de decisão de que direção seguir. Proporciona-se assim uma ponderação dessas variáveis para obter um conhecimento prévio para o próximo passo. A seguir é descrito o procedimento:

$$\boldsymbol{v}_{t+1}^{i} = \theta \boldsymbol{v}_{t}^{i} + c_1 a_1 \left(p^{i} - x_{t}^{i}\right) + c_2 a_2 \left(p_{t}^{s} - x_{t}^{i}\right) \tag{29}$$

Em que $\boldsymbol{v}_{t+1}^{i}$ é a velocidade da i-ésima partícula na $t+1$ iteração, a_1 e a_2 são números aleatórios entre $[0,1]$, p^{i}é a melhor posição encontrada pela i-ésima partícula, p_t^sé a melhor posição dentre todas as partículas na iteração t, θé o parâmetro de inércia no intervalo $[0.4\,,1.4]$ e c_1o parâmetro de confiança em si mesmo e c_2 de confiança na população. Logo, em um seguinte momento tem-se a atualização das posições da população, que é dada por:

$$x_{t+1}^{i} = x_{t}^{i} + \boldsymbol{v}_{t+1}^{i} \tag{30}$$

Note-se que as variáveis de otimização x_t^i são encaradas como a posição da partícula.

Com este procedimento torna-se possível uma exploraçãodiferenciada pelo espaço de busca. Permitindo que as partículas se aproximem parcialmente das melhores soluções encontradas, pois levam em consideração dois parâmetros (c_1 ec_2), eavaliando sua vizinhança com possíveis candidatos a nova solução com melhor incremento. Tal propriedade agrega uma opção de busca para uma heurística híbrida com o algoritmo SaMDE.

3.3 ESCOLHA DE HIBRIDAÇÃO PARA O ALGORITMO SAMDE

Na procurade uma forma de hibridar o algoritmo SaMDE, visando a melhoria de desempenho, vale destacar alguns fatos:Sabe-se que o algoritmo SaMDE possui uma rápida convergência (SILVA, 2010), logo o objetivo se concentrou na diversificação do algoritmo, ao invés da intensificação. Além disso, ressalta-se que o algoritmo SaMDE obteve eficiente convergência para as variáveis inteiras, nos casos estudados. Assim, limitou-se possíveis hibridações apenas as variáveis reais, que requerem maior trabalho.

Logo o Algoritmo de Busca por Enxame de Partículas se tornou uma boa opção porque possui um processo de busca que permite avaliar o comportamento coletivo dos indivíduos da população, em especial para variáveis contínuas. Assim, é possível adaptar o incremento de velocidade como uma diversificação ao algoritmo SaMDE.

Uma fase interessante para isto se destaca no *Passo 3* da Figura 12,em que se escolhe cinco soluções da população para serem utilizadas no processo de mutação diferencial. Desse modo pode-se adicionar uma pequena diversificação a estas soluções, piorando ou melhorando o seu desempenho, permitindo que percorra diferentes posições do espaço de busca sem sair de sua vizinhança próxima. Posteriormente deve-se então devolver tais soluções já diversificadas ao processo original do algoritmo SaMDE.

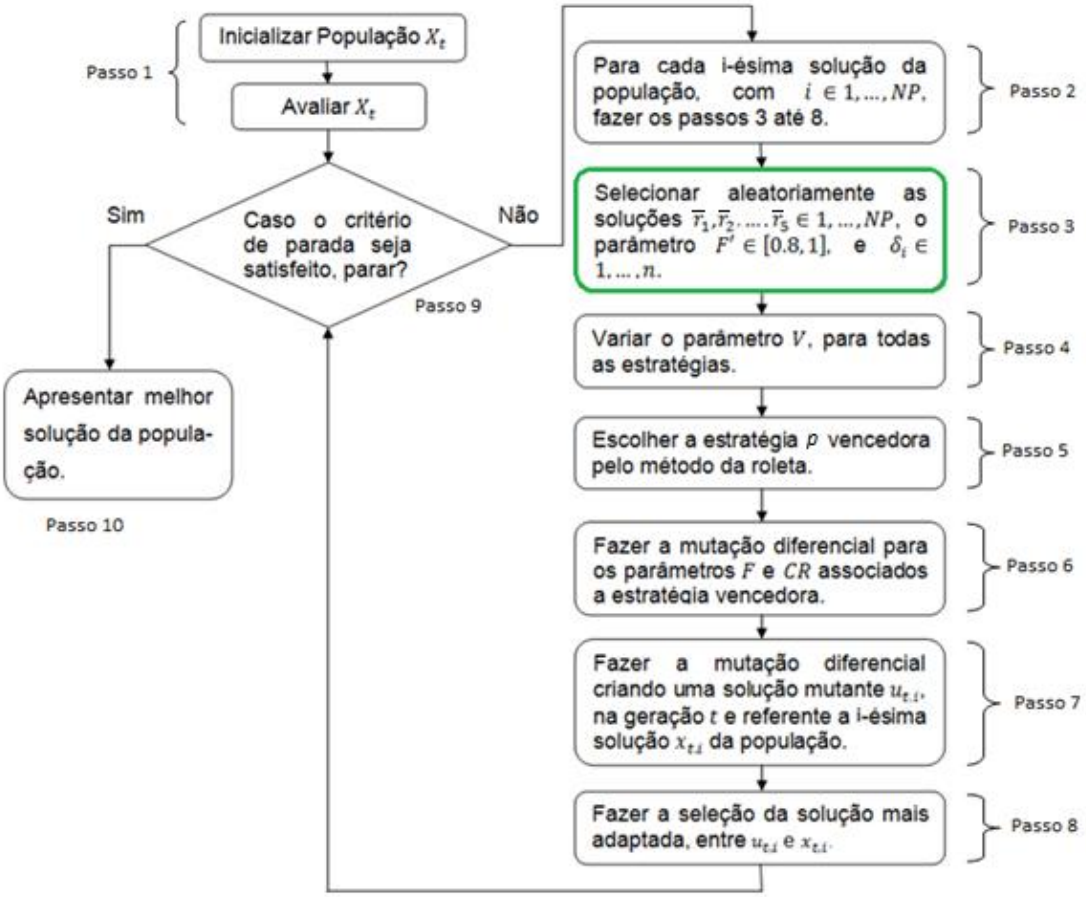

Figura 12: Pseudo-código do algoritmo SaMDE, com destaque a fase escolhida para hibridação.

3.4 PROCESSO DE DIVERSIFICAÇÃO BASEADO NO ALGORITMO DE BUSCA POR ENXAME DE PARTÍCULAS

Faz-se a hibridação pelo processo de diversificação baseado no algoritmo de busca por enxame de partículas. Iniciando-se com a escolha das cinco soluções da população,$\bar{r}_1, \bar{r}_2, \ldots, \bar{r}_3 \in 1, \ldots, NP$, que serão usadas posteriormente na mutação diferencial do SaMDE. Tais soluções devem ser escolhidas aleatoriamente, porém evitou-se a escolha da solução com melhor valor otimizado, porque apresentou desempenho não satisfatório na convergência, acelerando precocemente o algoritmoSaMDE.

Como o algoritmo de busca por enxame de partículas é um algoritmo populacional, logo se tornou necessária a estrutura de soluções vizinhas às soluções $\bar{r}_1, \bar{r}_2, \ldots, \bar{r}_5$, que formam assim uma população interna a este processo de diversificação.

Para este caso adotou-se dez soluções vizinhas para cada uma das $\bar{r}_k$ soluções, com $k = 1,2,3,4,5$. Tais vizinhos são cópias da solução $\bar{r}_k$ (Equação 31), que recebem uma pequena modificação em sua estrutura (Equação 32). Escolhem-se aleatoriamente duas posições j, com $j \in 1,2,\ldots,5$, referentes à variável real r_i (confiabilidade dos componentes do sistema), e aplica-se uma variação conforme a Equação 32.

$$\boldsymbol{vizinho_{u,k}} = \bar{r}_k \tag{31}$$

$$\boldsymbol{vizinho_{u,k}}(j) = \bar{r}_k(j) + (-1)^{\lambda}\frac{rand}{10^5} \tag{32}$$

Em que λ é uma variável de valor aleatório 0 ou 1, e $rand$ é um valor aleatório no intervalo $[0,1]$, e $\boldsymbol{vizinho_{u,k}}$é o u-ésimo vizinho em relação a solução$\bar{r}_k$**,** em que $u = 1,2,\ldots,10$ e $k = 1,2,\ldots,5$. Assim, se são considerados dez vizinhos por solução $\bar{r}_k$, então ao todo a população de vizinhos será de cinquenta, ver Figura 13. Observa-se, pela expressão (32), que a modificação permanece na quinta casa decimal, permitindo que a mudança não faça grandes alterações na solução.

Vale ainda observar que tanto o número de posições a serem modificadas quanto a modificação na quinta casa decimal, foram previamente testados para se estabelecer a opção de melhor desempenho.

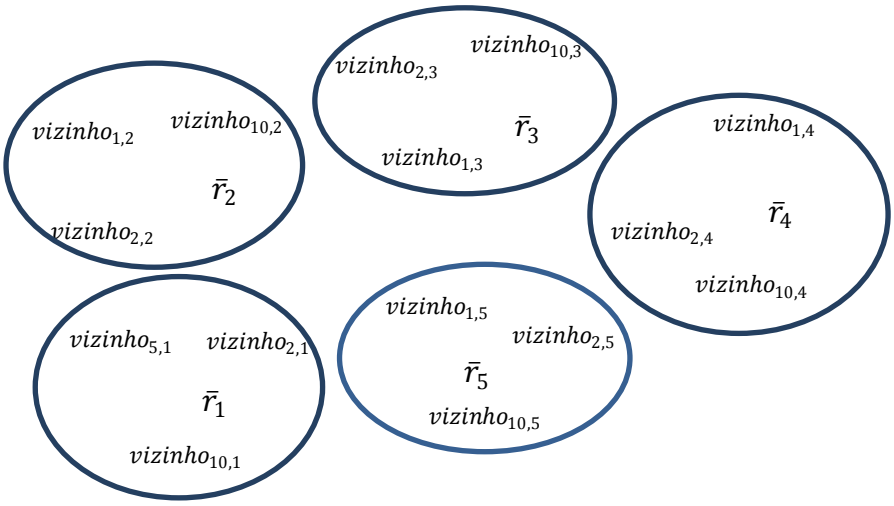

Figura 13: Representação da população de soluções vizinhas as soluções $\bar{r}_k$**,** com $k = 1,2,\ldots 5$.

Após esta modificação, convém validar e ajustar os intervalos das variáveis modificadas, para que estejam no intervalo válido, que no caso é no intervalo $[0,1]$. Além disso, podem-se haver soluções infactíveis, mesmo que suas variáveis estejam nos intervalos considerados. Para estes casos, desconsideram-se as soluções infactíveis, dado que não há um método simples que proporcione o retorno da factibilidade.

Após isso, o próximo passo é calcular a velocidade para as soluções$\bar{r}_k$. Para isto devemos ter a informação da solução global encontrada pela população de vizinhos, e a solução local.

A interpretação da solução global e da solução local é associada a população de vizinhos da seguinte forma: a melhor solução encontrada pela população de vizinhos será considerada como a melhor solução global (única para a população de vizinhos), e a melhor solução encontrada entre os dez vizinhos gerados de cada solução$\bar{r}_k$**,** será a melhor solução própria (tendo então cinco soluções próprias).

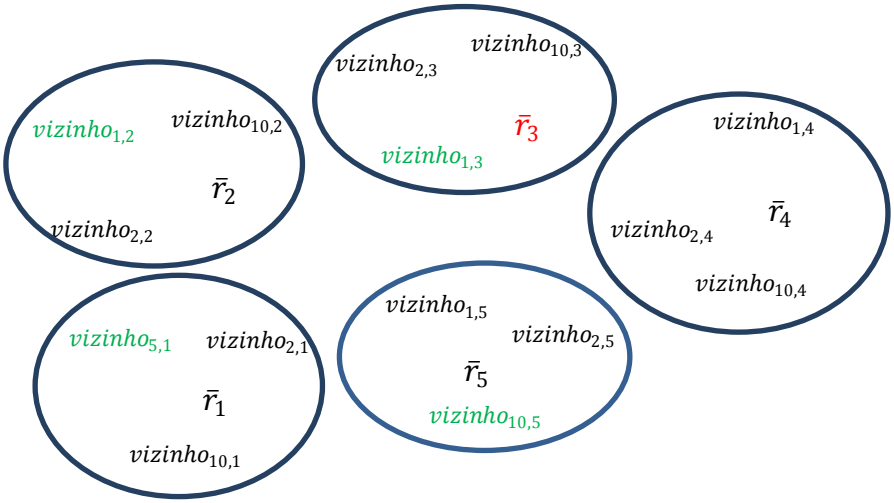

Figura 14: Representação da população de vizinhos, com destaque para a solução global (em vermelho) e para as soluções locais (em verde).

Após tudo isto, faz-se o cálculo da velocidade. Nota-se que originalmente este processo é recursivo no algoritmo de busca por enxame de partículas, porém nesta adaptação a recursividade causou grande diversificação as soluções, levando a uma convergência pré-matura do algoritmo SaMDE. A seguir a formula de cálculo de velocidade adaptada:

$$\boldsymbol{v}^k = \theta + c_1 a_1 \left(p^i - \bar{r}_k\right) + c_2 a_2 (p^{best} - \bar{r}_k) \quad (33)$$

Em que $\boldsymbol{v}^k$ é a velocidade referente à solução $\bar{r}_k$, a_1 e a_2 são números aleatórios entre $[0,1]$, p^ié a melhor solução própria encontrada pelas soluções vizinhas referente à $\bar{r}_k$, p^{best}é a melhor solução global, θé o parâmetro de inércia que originalmente está no intervalo $[0.4\,, 1.4]$, mas neste caso tomou-se por $\theta = 0{,}001$. O parâmetro $c_1$de confiançaem si mesmo, adotado por $c_1 = 0{,}1$, e o parâmetro $c_2$de confiança na população, adotado por$c_2 = 0{,}13$. Observa-se que se tem maior confiança na população do que em si mesmo.

Ainda deve-se verificar que a velocidade não ultrapasse um valor de 0,01, pois a velocidade acresce diretamente a solução e portanto não deve interferir ao menos nos valores das duas primeiras casas decimais. Desse modo minimizando o impacto da mudança nas soluções, que apresentam grande sensibilidade. Caso haja esta violação, faz-se a multiplicação necessária para atender o critério.

Finalmente, a atualização de cada $\bar{r}_k$solução, com $k = 1, 2, \ldots, 5$, é dividida entre duas possibilidades:

- Se todas as soluções próprias forem descartadas por causa da infactibilidade, então a atualização se dá por:

$$\bar{r}_k = \bar{r}_k + \boldsymbol{v}^k \quad (34)$$

- Caso contrário:

$$\bar{r}_k = p^i + \boldsymbol{v}^k \quad (35)$$

Neste momento, é novamente avaliada e ajustada as variáveis nos limites estabelecidos pelo problema.

Após todo o processo de diversificação, retorna-se ao algoritmo SaMDE original, levando as $\bar{r}_k$soluções geradas para a mutação diferencial. Ressalta-se que a última atualização de $\bar{r}_k$(Equações 34 e 35) pode gerar soluções infactíveis, porém no próprio critério de seleção do algoritmo SaMDE é também levada em consideração a factibilidade das soluções. Evitando que convirja para além do espaço de busca do problema.

Embora o algoritmo SaMDE seja recorrente, com diversas gerações, o processo proposto anteriormente não é. Isto, porque, a cada recorrência sua

levaria a diversificação a se tornar um processo de busca, intensificando o algoritmo como um todo. Logo deseja-se apenas a diversificação, que é obtida quando esta hibridação não é recorrente.

Como consequência de uma hibridação não recorrente, tem-se que seus parâmetros não podem ser auto-adaptados juntamente com o algoritmo SaMDE. Sendo, portanto, um objetivo de melhorias futuras.

3.5 AVALIAÇÃO DE LIMITES NO ALGORITMO SAMDE PARA O PROBLEMA DE ALOCAÇÃO DE CONFIABILIDADE-REDUNDÂNCIA

A avaliação de limites para o problema de alocação de confiabilidade-redundância pelo SaMDE faz-se diferente do apresentado originalmente por Silva (SILVA, 2010). Entendendo-se que foram necessárias adaptações para o problema, pois o método original não obteve resultados factíveis, dado que os intervalos de variação são muito pequenos.

Portanto, foi proposto dois métodos simples de avalição de limites e reposição das variáveis dentro desses limites. Um para as variáveis reais, no Quadro 3, e outro para as variáveis inteiras, no Quadro 4. Tais métodos são baseados na versão original, porém específicos para o tamanho dos intervalos necessários na implementação e não permitindo que as variáveis saíssem desses intervalos.

Quando a variável tratada for real, sendo no problema todas elas delimitadas ao intervalo $[0,1]$, se faz o seguinte ajuste:

1:	**If** $x < 0 \;\&\; x > -1$
2:	$x = 1 + x$
3:	**ElseIf** $x < -1$
4:	$x = rand/2$
5:	**Else If** $x \geq 1 \;\&\; x \leq 1{,}3$
6:	$x = 1{,}9 - x$
7:	**Else If** $x > 1{,}3$
8:	$x = 0{,}5 + rand/2$
9:	**End If**

Quadro 3:Pseudocódigo do método para avaliar e limitar as variáveisreais.

Já quando a variável for natural, observando que nos problemas aplicados o intervalo de variação é de $[1,5]$, é feito o seguinte ajuste:

1:	**If** $x < 0$
2:	$x = 1$
3:	**Else If** $x > na \& x < 2na$
4:	$x = 2na - x$
5:	**Else If** $x \geq 2na$
6:	$x = na$
7:	**Else If** $x = 0$
8:	$x = 2$
9:	**End If**

Quadro 4:Pseudocódigo do método para avaliar e limitar as variáveisinteiras.

Em que na é o número máximo de alocações permitidas pelo exemplo aplicado.

3.6 INICIALIZAÇÃO DA POPULAÇÃO E CRITÉRIO DE PARADA

Ambos os parâmetros de número de indivíduos na população (NP) e número máximo de gerações, são de escolha do pesquisador e devem ser estabelecidos no início da implementação.

A quantidade de indivíduos que constituirão a população não é um parâmetro auto-adaptado do algoritmo SaMDE, sendo necessários testes para avaliar qual o tamanho que gere melhora na convergência, porém sem sobrecarregar a quantidade de gerações para isto. Observando que a quantidade de gerações é também estabelecida pelo próprio usuário. No caso, tomou-se o máximo de gerações permitidas como condição de parada do algoritmo.

Além disso, a inicialização da população exige que, ao menos uma vez, seja gerada a população por meio de uma distribuição uniforme, que seja capaz de se espalhar por todo espaço de busca. Para isto, devem-se obter soluções, em que suas variáveis estejam dentro dos intervalos especificados. As variáveis reais r_i, referentes à confiabilidade dos componentes, originalmente são valores aleatórios no intervalo $[0,1]$, no entanto as variáveis não podem tomar valor nulo porque isto representaria um componente com nenhuma confiabilidade, logo um componente com falha.

Mas, também valores muito baixos causam um pior desempenho de convergência do algoritmo SaMDE. Assim, permitindo uma pequena

modificação, que retira o espaço de busca que, em geral, não é objetivado. Logo se o valor escolhido for abaixo de 0,1, é tomado um novo intervalo de escolha, sendo entre $[0{,}75; 0{,}9]$.

Ainda, as variáveis inteiras n_i da mesma forma não podem assumir valores nulos, pois estaríamos representando um subsistema sem componentes.

4 APLICAÇÃO AO PROBLEMA DE ALOCAÇÃO DE CONFIABILIDADE-REDUNDÂNCIA

Nesse capítulo é aplicado o algoritmo SaMDE e sua versão híbrida proposta, para o problema de alocação de confiabilidade-redundância. Também é comparado seus desempenhos para alguns exemplos clássicos da literatura.

4.1 FORMULAÇÃO DO PROBLEMA

A aplicação à exemplos mono-objetivo do problema de alocação de confiabilidade-redundância é relatada por diversos trabalhos da literatura, em que são usados diferentes algoritmos para sua resolução. Dentre esses, estão o algoritmo de evolução diferencial (LIU, QIN, 2014), o algoritmo genético (GEN, YUN, 2006), o algoritmo de sistema imune artificial (CHEN, 2006), o algoritmo de restrições substitutivas (HIKITA, 1992) e outros.

Além disso, tornou-se clássico a resolução de alguns exemplos referentes ao sistema linear, série-paralelo e complexo. Para cada problema a formulação da função objetivo é diferente, pois está ligada a estrutura proposta.

Para o problema com um sistema linear, como na Figura 15, segue-se diretamente o uso da Equação 2 do Capítulo 2 para se obter a função objetivo.

$$f(r,n) = \prod_{i=1}^{5} R_i(n_i) \tag{36}$$

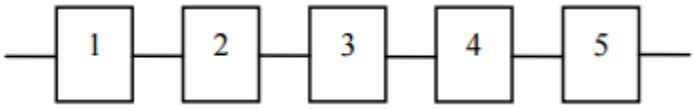

Figura 15: Estrutura do sistema em série do exemplo a ser implementado.

Já para o problema referente ao sistema em série-paralelo e complexo (Figuras 16 e 17), são usadas as Equações 4 e 6, e a decomposição usando o método RBD. Obtendo-se as seguintes funções objetivo:

$$f(r,n) = 1 - (1 - R_1R_2)(1 - (1 - R_3)(1 - R_4)R_5) \quad (37)$$

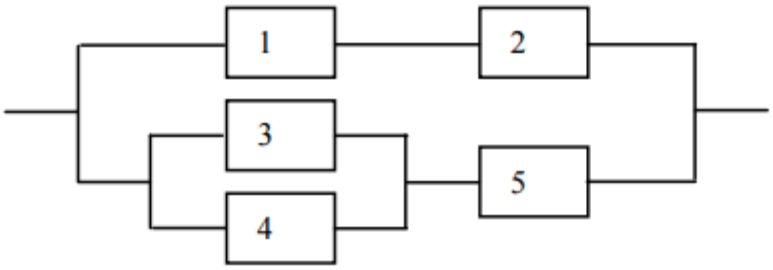

Figura 16: Estrutura do Sistema em série-paralelo do exemplo a ser implementado.

$$f(r,n) = R_1R_2 + R_3R_4 + R_1R_4R_5 + R_2R_3R_5 - R_1R_2R_3R_4 - R_1R_2R_3R_5 - R_1R_2R_4R_5 - R_1R_3R_4R_5 - R_2R_3R_4R_5 + 2R_1R_2R_3R_4R_5 \quad (38)$$

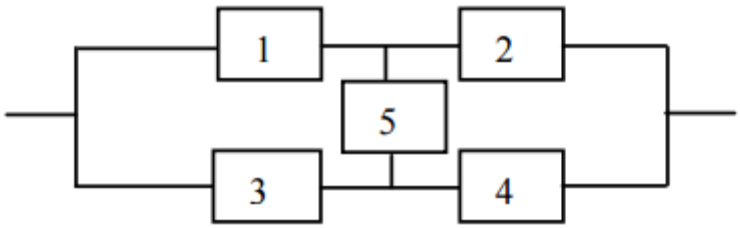

Figura 17: Estrutura do Sistema complexo do exemplo a ser implementado.

Todos os exemplos estão sujeitos as mesmas restrições, que se referem as variáveis de custo, peso e volume:

$$g_1(r,n) = \sum_{i=1}^{5} w_i v_i^2 n_i^2 \leq V \quad (39)$$

$$g_2(r,n) = \sum_{i=1}^{5} \alpha_i (-1000/\ln r_i)^{\beta_i} \left(n_i + exp(n_i/4)\right) \leq C \quad (40)$$

$$g_1(r,n) = \sum_{i=1}^{5} w_i n_i exp(n_i/4) \leq W \quad (41)$$

$$0 \leq r_i \leq 1, n_i \in \mathbb{Z}^+$$

Além disso, foram estabelecidos anteriormente na literatura valores aos parâmetros utilizados nas restrições, postos a seguir na Tabela 2 e Tabela 3.

Subsistema i	$10^5\alpha_i$	β_i	$w_i v_i^2$	w_i	V	C	W
1	2,500	1,5	2	3,5	180	175	100
2	1,450	1,5	4	4,0			
3	0,541	1,5	5	4,0			
4	0,541	1,5	8	3,5			
5	2,100	1,5	4	4,5			

Tabela 2: Parâmetros utilizados nas restrições do sistema em série-paralelo.

Subsistema i	$10^5\alpha_i$	β_i	$w_i v_i^2$	w_i	V	C	W
1	2,330	1,5	1	7	110	175	200
2	1,450	1,5	2	8			
3	0,541	1,5	3	8			
4	8,050	1,5	4	6			
5	1,950	1,5	2	9			

Tabela3: Parâmetros utilizados nas restrições do sistema em série e complexo.

Após otimizada a solução do problema, tem-se que a comparação é realizada observando pequenas escalas de fatores. Contudo, esta característica relaciona-se ao fato de que as instâncias consideradas são pequenas. Caso contrário, se as instâncias fossem de grande porte, haveria substancial diferença no valor observado.

Isto acontece porque a confiabilidade do sistema está diretamente associada à sua estrutura. A Tabela 4, mostra o impacto na confiabilidade de um sistema em série, supondo dois casos com uma pequena diferença de $0{,}009$ pontos percentuais nas componentes. Quanto maior o tamanho do sistema, incrementando seus componentes em série, maior o impacto sobre sua confiabilidade. Em grandes casos, já é possível ver notória diferença dentre os dois casos. Evidenciando, então, como mesmo em pequeno porte as poucas diferenças obtidas podem ser significativas.

Número de componentes em série	Confiabilidade dos componentes	
	99,999%	99,990%
	Confiabilidade do sistema	
100	99,90%	99,01%
250	99,75%	97,53%
500	99,50%	95,12%
1.000	99,01%	90,48%
10.000	90,48%	36,79%
100.000	36,79%	0,01%

Tabela 4: Impacto da confiabilidade de componentes na confiabilidade de sistemas em série (SALGADO, 2008).

4.2 RESULTADOS DA OTIMIZAÇÃO E COMPARAÇÕES

Foram aplicados a resolução dos exemplos o algoritmo SaMDE e sua versão híbrida proposta nesse trabalho. Ambas ainda não haviam sidoutilizadas na literatura para a resolução desses problemas.

Para isso, utilizou-se um número de indivíduos na população de 25 soluções, por 2500 gerações, lembrando que se tomou o número máximo de gerações como critério de parada.

Assim, foram realizadas100 repetições para cada problema em relação aos algoritmos aplicados. A seguir estão dispostos os dados referentes as confiabilidades dos sistemas.

A medida de Máxima Melhoria Possível - MMP, foi utilizada em (CHEN, 2006), em que se mede a proximidade percentual das soluções já conhecidas (R_{outros}) comparadas à solução por ele apresentada (R_s).

$$MMP = \frac{R_S - R_{outros}}{1 - R_{outros}}$$

Na tabela 5, 6 e 7, a comparação é feita entre a melhor solução obtida pelo algoritmo SaMDE e pela sua versão híbrida. Nota-se que as soluções se

mantiveram próximas, a não ser pelo caso do exemplo do sistema série-paralelo.

	SaMDE	SaMDE híbrido
Melhor	0,928986200502	**0,931680853389125**
Pior	0,886735505527079	0,910048853799822
Média	0,918904110852795	0,92522242
Desvio Padrão	0,009744327	0,005623755
MMP(%)	3,79%	

Tabela 5: Dados das 100 repetições referentes ao sistema em série da figura 15.

	SaMDE	SaMDE híbrido
Melhor	**0,99997608867**	0,99997209217
Pior	0,999497819563095	0,999653850898189
Média	0,999852599190094	0,999863808263684
Desvio Padrão	9,77338E-05	6,2825256E-05
MMP(%)	--	

Tabela 6: Dados das 100 repetições referentes ao sistema em série-paraleloda figura 16.

	SaMDE	SaMDE híbrido
Melhor	0,999889178233	**0,999889636623816**
Pior	0,99961474434	0,999803539541
Média	0,999818763519275	0,999884199994612
Desvio Padrão	7,8865E-05	1,80869E-05
MMP(%)	0,41%	

Tabela 7: Dados das 100 repetições referentes ao sistema em complexoda figura 17.

Observa-se que para os problemas do sistema em série e do sistema complexo houve melhor desempenho da versão híbrida, enquanto apenas para o problema do sistema em série-paralelo o algoritmo SaMDEobteve a melhor solução encontrada.

Para o problema em série, a diferença entre o SaMDE híbrido e o SaMDE se apresentou logo na segunda casa decimal, embora seu $MMP = 3{,}79\%$ seja menor comparado ao do problema em série-paralelo.Além disso, suas outras medidas (média, desvio padrão, pior solução) também se mostraram melhores em relação ao SaMDE.

Para o problema em série-paralelo, a melhor solução encontrada foi pelo algoritmo SaMDE, único caso em que ele superou a sua versão híbrida.A sua distância percentual não se aplica, dado que o SaMDE obteve melhor resposta. No entanto, a média, desvio padrão e pior solução obtiveram melhores valores pela versão híbrida do SaMDE.

E o problema do sistema complexo a melhor solução foi obtida pela versão híbrida do SaMDE, com uma pequena diferença na 10^{-7}. A distância percentual foi de $MMP = 0{,}41\%$, se mostrando soluções próximas. Quanto as outras medidas, a versão híbrida apresentou melhores resultados.

Ainda se nota que em todos os casos o desvio padrão constatado foi pequeno, ressaltando a baixa variação entre os resultados e evidenciando que ambos os algoritmos apresentaram robustez.

Ademais, o algoritmo SaMDE se mostrou mais rápido na resolução dos problemas, característica já esperada pois sua versão híbrida apenas adiciona processos ao algoritmo SaMDE original.

Nas Tabelas 8, 9 e 10, são apresentadas as melhores soluções relatadas na literatura para os exemplos tomados nesse trabalho.

Método	n	r_1	r_2	r_3	r_4	r_5	g_1	g_2	g_3	R_s
PSFS (MELLAL, ZIO, 2016)	(3,2,2,3,3)	0,77939888253	0,8718370134	0,90288535705	0,71140251717	0,7877994847351	27	4,132516551 x 10^{-11}	7,5289282411	**0,931682387907**
CS-GA (KANAGARAJ et al, 2013)	(3,2,2,3,3)	0,779398871	0,871837021	0,902885355	0,711402515	0,787799488	27	3,6977516515	7,5189182411	0,9316823876898
ABC (GARG et al, 2013)	(3,2,2,3,3)	0,779403565208	0,87183320141	0,902886411643	0,711398061305	0,787808548579	27	2,558957 x 10^{-10}	7,518918241	0,931682387672
BBO (GARG, 2015)	(3,2,2,3,3)	0,7794057271	0,8718339252	0,9028816332	0,7114011594	0,7878058784	27	4,7045 x 10^{-7}	7,518918	0,9316823874
ES-PSO (LIU, QIN, 2014)	(3,2,2,3,3)	0,7793997	0,878379	0,9028848	0,7114028	0,7877971	27	1,64373786 x 10^{-5}	7,5189182411	0,931682378235
IA (HSIEH, 2011)	(3,2,2,3,3)	0,779462304	0,871883456	0,902800879	0,711350168	0,787861587	27	5,284 x 10^{-7}	7,518918	0,931682340
PSSO (HUANG, 2015)	(3,2,2,3,3)	0,77946654	0,87173278	0,90284951	0,71148780	0,78781644	27	4,90819664 x 10^{-5}	7,51891824	0,931682297215271
NAFSA (HE et al, 2015)	(3,2,2,3,3)	0,77938841387	0,8718370134	0,90288535705	0,71140251717	0,7877994847351	27	6,7347 x 10^{-9}	7,518918	0,93168226855
ABC (YEH, HSIEH, 2011)	(3,2,2,3,3)	0,779399	0,871837	0,902885	0,711403	0,787800	27	-2,183652 x 10^{-4}	7,518918241	0,931682
SaMDE híbrido	$(3,2,2,3,3)$	***0,779168165902***	***0,872269897260***	***0,902507477728***	***0,711166687916***	***0,788199776175***	***27***	***0,000038***	***7,518918***	***0,931680853389***
IAs (CHEN, 2006)	(3,2,2,3,3)	0,779266	0,872513	0,902634	0,710648	0,788406	27	1,559 x 10^{-3}	7,518918	0,931678
GA (HSIEH et al, 1998)	(3,2,2,3,3)	0,780874	0,871292	0,902316	0,711945	0,786995	27	0,00395148681	7,5189182411	0,9316734167501
GA (GEN, YUN, 2006)	(3,2,2,3,3)	0,779427	0,869482	0,902674	0,714038	0,786896	27	0,121454	7,518918	0,931578
SSO (HUANG, 2015)	(3,2,2,3,3)	0,78271484	0,8735199	0,90264893	0,71313477	0,77729797	27	0,00182140	7,51891824	0,93150199
SaMDE	$(3,3,2,3,2)$	***0,7777246190467***	***0,8214829913357***	***0,8969007271623***	***0,7052398385025***	***0,8582981450313***	***27***	***0,00001***	***10,572476***	***0,928986200502***
PSO (HUANG, 2015)	(2,3,2,4,2)	0,80059281	0,74049316	0,82914384	0,63686144	0,88704276	4	16,775727	4,8156147	0,8885037

Tabela 8: Melhores soluções para o problema de sistema em série.

Método	n	r_1	r_2	r_3	r_4	r_5	g_1	g_2	g_3	R_s
ABC (YEH, HSIEH, 2011)	(2,2,2,2,4)	0,8197457	0,8450080	0,8954581	0,9009032	0,8684069	40	-1,4695223388	1,6092889667	**0,99997731**
ES-PSO (LIU, QIN, 2014)	(2,2,2,2,4)	0,819655	0,849975	0,895509	0,895509	0,868449	40	1,37197921 x 10^{-4}	1,6092889667	0,999976698998558
GA-PSO (SHEIKHAHSHAHI, 2013)	(2,2,2,2,4)	0,819640	0,845091	0,895482	0,895517	0,868430	40	-4,8540641 x 10^{-4}	1,6092889667	0,999976649287932
TS-DE (LIU, QIN, 2014)	(2,2,2,2,4)	0,819659	0,844981	0,895507	0,895506	0,868448	40	-2,66935542 x 10^{-4}	1,609288966	0,9999766491
PSFS (MELLAL et al, 2016)	(2,2,2,2,4)	0,81965939118	0,84498085296	0,89550643076	0,89550645172	0,86844769346	40	1,850821718 x 10^{-18}	1,6092889667	0,999976649066172
DE (LIU, QIN, 2015)	(2,2,2,2,4)	0,81965932	0,84498074	0,89550642	0,89550643	0,86844775	40	1,961642794 x 10^{-7}	1,609288966	0,999976649066076
CS-GA(KANAGAREJ et al, 2013)	(2,2,2,2,4)	0,819483232488	0,844783084455	0,895519305	0,895492245	0,868447587	40	-2,7177208 x 10^{-7}	1,6092889667	0,999976649065963
ABC (GARG et al, 2013)	(2,2,2,2,4)	0,819737753469	0,844991099776	0,895529543820	0,895433687206	0,868434824469	40	1,39152 x 10^{-10}	1,609288966	0,999976649054
BBO (GARG, 2015)	(2,2,2,2,4)	0,8196583448	0,8449101406	0,8954871713	0,8955148963	0,8684681613	40	1,748541669 x 10^{-5}	1,6092889667	0,999976649048875
IA (HSIEH, 2011)	(2,2,2,2,4)	0,8195911561	0,844951068	0,895428548	0,895522339	0,868490229	40	5,9845376654 x 10^{-8}	1,6092889667	0,999976649036520
PSSO (HUANG, 2015)	(2,2,2,2,4)	0,81958939	0,84458412	0,89534134	0,89581626	0,8685902	40	8,11794613 x 10^{-5}	1,609288966	0,999976648738107
NAFSA (HE et al, 2015)	(2,2,2,2,4)	0,819787575473	0,845671943728	0,894868363315	0,895908268569	0,868295830551	40	3,1248 x 10^{-8}	1,609289	0,999976648004
IAs (CHEN, 2006)	(2,2,2,2,4)	0,812485	0,843155	0,897385	0,894516	0,870590	40	2,627 x 10^{-3}	1,609289	0,99997658
SSO (HUANG, 2015)	(2,2,2,2,4)	0,813858803	0,83912659	0,89366150	0,89845276	0,87106323	40	0,0024	1,609288966	0,99997657
SaMDE	***(2,2,2,2,4)***	***0,8383503691***	***0,8410071930***	***0,8940014860***	***0,8904214673***	***0,8645799430***	***40***	***0,011777565***	***1,609288967***	***0,9999760886***
GA (HSIEH, 1998)	(2,2,2,2,4)	0,785452	0,842998	0,885333	0,917958	0,870318	40	1,194440	1,609289	0,99997418
SaMDE híbrido	***(2,2,2,2,4)***	***0,75029459350***	***0,8510353245***	***0,91304640187***	***0,90247962606***	***0,872567520842***	***40***	***0,052792216***	***1,609288967***	***0,99997209217***
PSO (HUANG, 2015)	(4,3,2,1,2)	0,84025282	0,88865099	0,62375055	0,93984950	0,75158691	68	0,916915088	4,017703641	0,99985845
GA-PSO (SAHOO et al, 2014)	(4,3,2,3,2)	0,860328	0,827071	0,872994	0,937884	0,701148	4	-22,2116202	-13,7167075	0,999504300525587

Tabela 9: Melhores soluções para o problema de sistema em série-paralelo.

Método	n	r_1	r_2	r_3	r_4	r_5	g_1	g_2	g_3	R_s
ES-PSO (LIU, QIN, 2014)	(3,3,2,4,1)	0,828082	0,857812	0,914241	0,648155	0,704066	5	-2,3814399 x 10^{-4}	1,560466288	**0,999889638110263**
GA-PSO (SHEIKHAHSHAHI, 2013)	(3,3,2,4,1)	0,828134	0,857831	0,914192	0,648069	0,704476	5	-2,7258899 x 10^{-4}	1,560466281	0,999889638109295
PSFS (MELLAL et al, 2016)	(3,3,2,4,1)	0,828121141729	0,85781341076	0,91423527822	0,64807680660	0,70424641245	5	2,854643724 x 10^{-6}	1,560466288	0,999889637512067
DE (LIU, QIN, 2015)	(3,3,2,4,1)	0,82808641	0,85780478	0,91424067	0,64814622	0,70416210	5	1,987824296 x 10^{-5}	1,560446288	0,999889637503023
TS-DE (LIU, QIN, 2014)	(3,3,2,4,1)	0,828086	0,857805	0,914241	0,648146	0,704162	5	3,71181366 x 10^{-5}	1,560466288	0,999889637462077
BBO (GARG, 2015)	(3,3,2,4,1)	0,8208606892	0,8580404545	0,9141487486	0,6479689012	0,7042048796	5	1,4541 x 10^{-4}	1,560466	0,9998896364
NAFSA (HE et al, 2015)	(3,3,2,4,1)	0,82832179189	0,85797450730	0,91422098825	0,64775717018	0,0,70300666185	5	1,54851x 10^{-5}	1,56046629	0,99988963601
ABC (GARG et al, 2013)	(3,3,2,4,1)	0,827970276262	0,57874758586	0,914186404228	0,648355386813	0,703575311047	5	3,74636763 x 10^{-4}	1,560466288	0,999889635809
PSSO (HUANG, 2015)	(3,3,2,4,1)	0,82783292	0,85771241	0,91437458	0,64861002	0,70287554	5	2,50290205 x 10^{-5}	1,560466288	0,999889635738075
CS-GA (KANAGARAJ et al, 2013)	(3,3,2,4,1)	0,82808567	0,85780695	0,91424006	0,64814375	0,70718228	5	-7,3070123 x 10^{-4}	1,560466288	0,9998896
IA (HSIEH, 2011)	(3,3,3,3,1)	0,816624176	0,868767396	0,858748781	0,710279379	0,753429200	18	1,494 x 10^{-8}	4,264770	0,9998893505
SaMDE híbrido	***(3,3,3,3,1)***	***0,8280288269027***	***0,8579120318311***	***0,9142770255737***	***0,6481878752261***	***0,7029133123747***	***18***	***0,000539***	***4,264770***	***0,999889178233***
GA (GEN, YUN, 2006)	(3,3,3,3,1)	0,808258	0,866742	0,861513	0,716608	0,766894	18	9 x 10^{-4}	4,264770	0,999889
SSO (HUANG, 2015)	(3,3,2,4,1)	0,82008362	0,85119629	0,91854858	0,66072083	0,70275879	5	0,00437599	1,560466288	0,99988862
GA (HSIEH, 1998)	(3,3,3,3,1)	0,814090	0,864614	0,890291	0,701190	0,734731	18	0,376347	4,264770	0,99987916
PSO (HUANG, 2015)	(3,3,2,2,3)	0,77061588	0,90109253	0,89278651	0,6083008	0,73451002	37	16,54571312	1,4118003224	0,9996740
ABC (YEH, HSIEH, 2011)	(3,3,2,4,1)	0,828087	0,857805	0,704163	0,648146	0,914240	5	-25,43392577	1,560466288	0,999484073426439
IAs (CHEN, 2006)	(3,3,3,3,1)	0,812485	0,867661	0,861221	0,7'3852	0,756699	18	1,494 x 10^{-3}	4,264770	0,9992639646662971
GA-PSO (SAHOO et al, 2014)	(4,3,3,3,1)	0,858430	0,7	0,922386	0,7	0,7	11	-24,896621498	-27,3901210	0,999263964662971
SaMDE	***(3,3,3,3,1)***	***0,8205824352642***	***0,8701587825315***	***0,8546345419890***	***0,7061795421273***	***0,7533129622607***	***5***	***0,00001***	***1,560466***	***0,9988963662382***

Tabela 10: Melhores soluções para o problema de sistema complexo

Observa-se que há um eficaz desempenho do algoritmo SaMDEe de sua versão híbrida, com diferenças apenas na 10^{-6} e 10^{-5} casas decimais,mostrando resultados melhores do que alguns métodos propostos anteriormente, no entanto não ultrapassando as melhores soluções conhecidas.

Por final temos que tanto o algoritmo SaMDE quanto sua versão híbrida proposta nesse trabalho mostraram satisfatória convergência para os exemplos tomados. Porém, ainda assim não obtendo maior desempenho do que as já conhecidas melhores soluções.

5 CONSIDERAÇÕES FINAIS E SUGESTÕES PARA TRABALHOS FUTUROS

No presente trabalho foi utilizado o problema de alocação de confiabilidade-redundância para ser otimizado pelo algoritmo SaMDE, sendo que ainda não haviam relatos de sua implementação para esse problema. Além disso, também se propôs uma versão híbrida com o algoritmo de Otimização por Enxame de Partículas, no intuito de melhorar o desempenho por meio de uma maior diversificação para o processo de busca, evitando intensificar sua convergência que já se apresentava suficiente.

Ambos os algoritmos foram implementados ao problema de alocação de confiabilidade-redundância.Considerou-se três exemplos clássicos na literatura, referentes ao sistema em série, série-paralelo e complexo.

Comparando entre os dois algoritmos, a versão híbrida proposta apresentou melhor desempenho no problema do sistema em série e complexo, já o algoritmo SaMDE se sobressaiu no problema do sistema em série-paralelo, porém mostrando baixo desempenho para os outros exemplos.

Já comparando as soluções conhecidas na literatura, observa-se que os resultados, das implementações desse trabalho, mostraram bom desempenho, estando próximos as melhores soluções.

Recomendam-se para trabalhos futuros, ainda considerando a resolução do problema de alocação de confiabilidade-redundância, o uso de métodos exatos juntos as metas-heurísticas, com o intuito de minimizar o espaço de busca. E também podendo melhorar sua busca local, aspecto importante a problemas com a sensibilidade observada.

Também, um futuro algoritmo baseado na versão híbrida, totalmente auto-adaptativo. Possivelmente utilizando o algoritmo de Busca por Enxame de Partículas auto-adaptativo.

Além disso, recomenda-se ainda algumas mudanças para o SaMDE híbrido. Seu processo de busca se baseia na produção de uma população de soluções vizinhas das soluções selecionadas $\bar{r}_k$. Estes vizinhos são cópias das soluções, que recebem uma modificação, apresentada na Equação 32. Tal modificação pode ser substituída por outros métodos, que tragam melhores

características a esta população de soluções vizinhas, melhorando então o processo de busca.

REFERÊNCIAS

BREST, J. **Constrained Real-ParameterOptimizationwith e-Self-AdaptiveDifferentialEvolution**. In: Mezura-Montes, Constraint-Handling in EvolutionaryComputation, chapter 4, p. 73-93. Springer. Studies in ComputationalIntelligence, v. 198, Berlin, ISBN: 978-3-642-00618-0, 2009.

CHANG, C. S.; XU, D. Y.; QUEK, H. B. Pareto-optimal set basedmultiobjectivetuningoffuzzyautomatictrainoperation for masstransit system. IEEE Proceedings on Electric Power Applications 146: 577-583.

CHEN, T. C. IAsbased approach for reliabilityredundancyallocationproblems. **AppliedMathematicsandComputation**, v. 182, p. 1556-1567, 2006.

CHEN, T. C.; YOU, P. S. Immune Algorithm based approach for redundant reliability porblems. **Computers in Industry**, v. 56, p. 195-205, 2005.

CHERN, M. S. Onthecomputationalcomplexityofreliabilityredundancyallocation in a series system. **Oper. Res. Lett**, v. 11, p.309-315, 1992.

COELHO, L. C. Na efficientparticleswarm approach for mixed-integerprogramming in reliability-redundancyoptimizationapplications. **ReliabilityEngineeringand System Satefy**, v. 94, p. 830-837, 2009.

COIT, D. W.; SMITH, A. E. Reliability Optimization of series-parallel systems using a genetic algorithm. **IEEE Transaction on Reliability**, v. 45, p. 254-260, 1996.

DAS, S.; MULLICK, S. S.; SUGANTHAN, P. N. Recentadvances in differentialevolution - Na updatedsurvey. **SwarmandEvolutionaryComputation**, v. 27, p. 1-30, 2016.

DINGRA, A, K. OptimalApportionmentofReliability&Redundancy in Series Systems underMultipleObjectives. **IEEE TransactionsonReliability**, v. 41, n. 4, p. 576-582, 1992.

EIBEN, A. E.; MICHALEWICZ, Z.; SCHOENAUER, M.; SMITH, J. E. Parametercontrol in evolutionaryalgorithms. **IEEETransactionsonEvolutionaryComputation**.

GARG, H. Na efficientbiogeographybasedoptimizationalgorithm for solvingreliabilityoptimizationproblems. **SwarmandEvolutionaryComputation**, v. 24, p. 1-10, 2015.

GARG, H.; RANI, M.; SHARMA, S. P. Na efficienttwophase approach for solvingreliability-redundancyallocationproblemusing artificialbeecolonytechnique. **Comput. Oper. Res.**, v. 40, n. 12, p. 2961-2969, 2013.

GEN, M.; YUN, Y. S. Seftcomputing approach for reliabilityoptimization: stateof-the-artsurvey. **Reliab. Eng. Syst. Sef.**, v. 91, n. 9, p. 1008-1026, 2006.

GUIMARÃES, F. G. Algoritmos de Evolução Diferencial. Manual de Algoritmos Evolutivos e Metaheurísticas.

GOPAL, K.; AGGARWAL, K. K. A new method for solvingreliabilityoptimization problem. **IEEE TransactiononReliability**, v. 28, p. 36-38, 1978.

HALPERN, S. The assurance Science. An Introduction to Quality Control and Reliability. New Jersey, USA, 1978.

HE, Q.; HU, X.; REN, H.; ZHANG, H. A novel artificial fishswarmalgorithm for solvinglarge-scalereliability-redundancyapplication problem. **ISA Trans.**, v. 59, p.105-113, 2015.

HIKITA, M.; NAKAGAWA, Y.; HARIHISA, H. Reliabilityoptmizationof systems by a surrogateconstraintsalgorithm. **IEEE TransactiononReliability**, v. 41, n. 3, p. 473-480, 1992.

HSIEH, Y. C.; CHEN, T. C.; BRICKER, D. L. Geneticalgorithm for reliability design problems. **MicroelectronReliab.**, v. 38, n. 10, p. 1599-1609, 1998.

HSIEH, Y. C.; YOU, P. S. Na effectiveimmunebasedtwo-phase approach for theoptimalreliability-redundancyallocation problem. **Applic. Math Comput.**, v. 2, n. 4, p. 1297-1307, 2011.

HUANG, C.L. A particle-basedsimplifiedswarmoptimizationalgorithm for reliabilityredundancyallocationproblems. **Reliab. Eng. Syst. Sef.**, v. 142, p. 221-230, 2015.

INCONSE. **SistemEngineeringHandbook**. v. 3, p. 185, ISBN: 2003 002 03, 2006.

QIN, A. K.; SUGANTHAN, P. N. **Self-adaptivedifferentialevolutionalgorithm for numericaloptimization**. In: IEEE CongressonEvolutionaryComputation (CEC 2005), p. 1785-1791, IEEE Press.

KANAGARAJ, G.; PONNAMBALAM, S. G.; JAWAHAR, N. Ahybridcuckoosearchandgeneticalgorithm for reliability-redundancyallocationproblems. **Comput. Ind. Eng.**, v. 66, n. 4, p. 1115-1124, 2013.

KUO, W.; LIN, H.; XU, A.; ZHANG, W. ReliabilityoptimizationwiththeLagrangemultiplierandbranchandboundtechnique. **IEEE TransactiononReliability**, v. 36, 624-630, 1987.

KUO, W.; PRASAD, V. R. An notated Overview of System-ReliabilityOptimization. **IEEE TransactionsonReliability**, v. 49, n. 2, p. 176-187, 2000.

LIU, Y.; QIN, G. A hybrid TS-DE algorithm for reliabilityredundancyoptimization problem. **J. Comput.**, v. 9, n. 9, p. 2050-2057, 2014.

LIU, Y.; QIN, G. A DE algorithmcombinedwithLévyflight for reliabilityredundancyallocationproblems. **Int. J. Hybrid Inf. Technol.**, v. 8, n. 5, p. 113-118, 2015.

LIU, Y.; QIN, G. Reliabilityredundancyoptimizationalgorithmbasedoneaglestrategyand PSO. **Int. J. MultimedUbiquitous Eng.**, v. 9, n. 12, p. 375-382, 2014.

MARTORELL, S.; SANCHEZ, A.; CARLOS, S.; SERRADELL, V. Alternativeandchallenges in optimizing industrial safetygeneticalgorithms. **Reliab. Eng. Syst. Saf.**, v. 86, n.1, p. 25-38, 2004.

MELLAL, M. A.; ZIO, E. A penalityguidedstochastic fractal search approach for system reliabilityoptimization. **ReliabilityEngineeringand System Safety**, v. 152, p. 213-227, 2016.

MEZURA-MONTES, Efr. n; VELAZQUES-REYES, J.; COELLO, C. A. C. A comparativestudyofdifferentialevolutionvariants for global optimization. In: GECCO'06: Proceedingsofthe 8th annualconferenceonGeneticandevolutionarycomputation, p. 485-492, New York, NY, USA. ACM, 2006

OUZINEB, M.; NOURELFATH, M; GENTREAU, M. Tabu search for theredundancyallocationproblemofhomogenous series-parallelmulti-state systems. **Reliab. Eng. Syst. Saf.**, v. 93, n. 8, p. 1257-1272, 2008.

RONKKONEN, J.; KUKKONEN, S.; PRICE, K. V. Real-parameteroptimizationwithdifferentialevolution. **IEEE TransactinonReliability**, v.1, p.506-513, 2005.

SADJADI, S. J.; SOLTANI, R. Alternative design redundancyallocationusing an efficientheuristicand a honeybeematingalgorithm. **Expert Systems withApplications**, v. 39, p. 990-999, 2012.

PEREIRA, JOSÉ FRANCISCO. Avaliação do Impacto Radiológico Potencial Relativo à visita de Submarinos nucleares ao porto do rio de janeiro. 2004. p. 95. **Dissertação** - Comissão Nacional de Energia Nuclear: Instituto de Radioproteção e dosimetria, Rio de Janeiro, Rio de Janeiro, 2004. Disponível em: < http://www.iaea.org/inis/collection/NCLCollectionStore/_Public/39/088/ 39088888.pdf>. Acesso em: 17 jan. 2017.

SAHOO, L.; BANERJEE, A.; BHUNIA, A. K.; CHATOOPADHYAY, S. Na efficient GA-PSO approach for solvingmixed-

integernonlinearprogramingproblem in reliabilityoptimization. **SwarmEvolut. Comput.**, v. 19, p.43-51, 2014.

SALGADO, MARCIA DE FATIMA PLATILHA. Aplicação de Técnicas de Otimização à Engenharia de Confiabilidade. 2008. p. 122. **Dissertação** - Universidade Federal de Minas Gerais, Belo Horizonte, Minas Gerais, 2008. Disponível em: <http://www.bibliotecadigital.ufmg.br/dspace/bitstream/handle/

1843/BUOS-8CDG4N/marcia_de_fatima_platilha_salgado.pdf?sequence=1>. Acesso em: 17 jan. 2017.

SCHWEFEL, H. **Numericaloptimizationofcomputermodels**. Wiley, Chichester, WS, UK, 1981.

SHEIKHALISHAHI, M.; EBRAHIMIPOUR, V.; SHIRI, H.; ZAMAM, H.; JEIHOONIAN, M. A hybrid GA-PSO approach for reliabilityoptimization in redundancyallocation problem. **Int. J. Adv. Manuf. Technol.**, v. 68, n. 1, p. 317-338, 2013.

SILVA, Rodrigo Cesar Pedroza. Um estudo sobre a autoadaptação de parâmetros na Evolução Diferencial. 2010. p. 54. **Dissertação** - Universidade Federal de Ouro Preto, Ouro Preto, Minas Gerais, 2010. Disponível em: < http://www.decom.ufop.br/menotti/monoII102/files/BCC391-102-vf-07.1.4116-RodrigoCesarPedrosaSilva.pdf>. Acesso em: 17 jan. 2017.

STORN, R.; PRICE, K. **Differencialevolution - a simpleandefiicientadaptivescheme for global optimization over continuousspaces.**Technical report.

STORN, R.; PRICE, K. Differentialevolution - a simpleandefficientheuristic for global optimization over continuousspaces. **J. of Global Optimization**, v. 11, n. 4, pp. 341-359, 1997.

TAVAKKALI-MOGHADDAN, R.; SAFARI, J.; SASSANI, F. Reliabilityoptimizationof series-parallel systems with a choiseofredundancystrategiesusing a geneticalgorithm. **Reliab. Eng. Syst. Saf.**, v. 93, p.550-556, 2008.

TEO, J. Exploringdynamic self-adaptivepopulations in differentialevolution. **Soft Comput.**, v. 10, n. 8, p. 673-686, 2006.

TILLMAN, F. A.; HWANG, C.L.; KUO, W. Determiningcomponentreliabilityandredundancy for optimum system reliability. **IEEE TransactiononReliability**, v. 26, n. 3, p. 162-165, 1977.

YEH, W. C.; HSIEH, T. J. Solvingreliabilityredundancyallocationproblemsusing na artificial beecolonyalgorithm. **Computers&OperationsResearch**, v. 38, p. 1465-1473, 2011.

ZOU, D.; GAO, L.; LI, S.; WU, J. Na effective global harmonysearchalgorithm for reliabilityproblems. **Expert Systems withApplications**, v. 38, p. 4642-4648, 2011.

ZOUFAGHARI, H.; HAMADANI, A. Z.;ARDAKAN, M. A.; Bio-objectiveredundancyallocationproblem for a system withmixedrepairableand non-repairablecomponents. **ISA Transactions**, v. 53, p. 17-24, 2014.