

**UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CAMPUS GUARAPUAVA
COORDENAÇÃO DE ENGENHARIA MECÂNICA**

GUILHERME DE PAULA PRADO

**ALGORITMO GENÉTICO DE CODIFICAÇÃO REAL APLICADO À
OTIMIZAÇÃO TERMoeCONÔMICA**

**GUARAPUAVA
2021**

GUILHERME DE PAULA PRADO

**ALGORITMO GENÉTICO DE CODIFICAÇÃO REAL APLICADO À
OTIMIZAÇÃO TERMoeCONÔMICA**

Trabalho de Conclusão de Curso apresentado
à Coordenação de Engenharia Mecânica,
como requisito parcial à obtenção do Título
de Bacharel em Engenharia Mecânica, da
Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Dr. Renan M. Galante

GUARAPUAVA

2021



[4.0 International](https://creativecommons.org/licenses/by-nc/4.0/)

Esta licença permite remixe, adaptação e criação a partir do trabalho, para fins não comerciais, desde que sejam atribuídos créditos ao(s) autor(es). Conteúdos elaborados por terceiros, citados e referenciados nesta obra não são cobertos pela licença.

TERMO DE APROVAÇÃO

ALGORITMO GENÉTICO DE CODIFICAÇÃO REAL APLICADO À OTIMIZAÇÃO TERMoeconômica

GUILHERME DE PAULA PRADO

Este Trabalho de Conclusão de Curso foi apresentado em Guarapuava, Paraná, no dia 07 de dezembro de 2021, de forma remota, como requisito parcial à obtenção do título de Bacharel em Engenharia Mecânica. O candidato foi arguido pela Banca Examinadora, composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Renan Manozzo Galante
Prof. Orientador

Luan Jose Franchini Ferreira
Membro da banca

Sérgio Dalmas
Membro da banca

Sérgio Dalmas
Coordenador do Curso de Engenharia Mecânica

Guilherme de Paula Prado

AGRADECIMENTOS

Aos meus pais, Dimas Samoel Prado e Adriana Ap. P. Prado, pela criação, pelo amor e carinho, e pelo suporte emocional e financeiro durante este período da minha vida.

Aos meus irmãos, Leonardo P. Prado e Laryssa P. Prado, que têm participação significativa na minha formação como pessoa.

Ao meu orientador, Renan M. Galante, que além de muito contribuir para a realização deste trabalho, compartilhou conhecimentos de valores imensuráveis, e em nenhum momento, durante a orientação, deixou de prezar pelo aspecto humano.

À minha namorada, Amanda Molinari, a pessoa que melhor me conhece neste mundo, com quem posso contar sempre que necessário, e que me acompanhou durante praticamente todo o período de minha graduação.

À Universidade Tecnológica Federal do Paraná, seu corpo docente e demais servidores que contribuíram de forma direta e indireta para a minha formação como acadêmico e futuro profissional.

Por fim, agradeço a todos os meus familiares, amigos e colegas que me acompanharam durante a minha trajetória.

A História está repleta de pessoas que, como resultado do medo, ou por ignorância, ou por cobiça de poder, destruíram conhecimentos de imensurável valor que, em verdade, pertenciam a todos nós. Nós não devemos deixar isso acontecer de novo. (SAGAN, Carl, 1980).

RESUMO

PRADO, Guilherme. **Algoritmo Genético de Codificação Real Aplicado à Otimização Termoeconômica**. 2021. 100 f. Trabalho de Conclusão de Curso – Coordenação de Engenharia Mecânica, Universidade Tecnológica Federal do Paraná. Guarapuava, 2021.

Este trabalho propôs a utilização de um Algoritmo Genético de Codificação Real (*RCGA* – do inglês *Real-Coded Genetic Algorithm*) em substituição à análise exergoeconômica clássica, na otimização de um sistema térmico de geração de energia por incineração de resíduos sólidos. A otimização é realizada utilizando como função objetivo o período de *payback*, isto é, o período de tempo de recuperação do capital investido. O RCGA baseia-se na teoria da seleção natural, e busca soluções que minimizam o período de *payback* a partir de soluções candidatas iniciais, geradas aleatoriamente, que são melhoradas ao longo de iterações, chamadas gerações, por meio de operadores que simulam reprodução, mutação e seleção, conforme observado na natureza. O sistema é modelado com base nas 1ª e 2ª lei da termodinâmica, sendo implementado na linguagem Python. Partindo de um caso padrão com resíduo sólido urbano (RSU) sem custo e 3,89 anos de *payback*, a otimização pelo RCGA resultou em uma redução de 32,90%, chegando em 2,61 anos. Nesse cenário, quando comparado ao caso ótimo encontrado pela análise exergoeconômica clássica, com 3,06 anos de *payback*, a redução é de 14,71%. Os resultados da otimização via RCGA apontam para a redução da eficiência exergética de alguns equipamentos para a otimização de todo o sistema. Alguns trocadores de calor tiveram suas áreas de troca térmica reduzidas no caso ótimo. A otimização se dá pelo incremento de 57,60% no fluxo de caixa (R\$/h) e de 5,75% de aumento no capital total investido (R\$). A vazão mássica de resíduo sólido queimado do incinerador (\dot{m}_{RSU}) é aumentada em 33,33%, passando de 525 kg/h no caso padrão para 700 kg/h no caso ótimo.

Palavras-chave: Otimização. Algoritmo Genético. Exergia. Exergoeconomia.

ABSTRACT

PRADO, Guilherme. **Real-Coded Genetic Algorithm Applied to Thermoeconomic Optimization**. 2021. 100 f. Trabalho de Conclusão de Curso – Coordenação de Engenharia Mecânica, Universidade Tecnológica Federal do Paraná. Guarapuava, 2021.

This work proposed the use of a Real-Coded Genetic Algorithm (RCGA) in classical exergoeconomic analysis substitution on optimization of a thermal power generation system by solid waste incineration. The optimization is performed by using payback as objective function. The RCGA is based on natural selection theory and looks for optimal solutions from randomly generated initial solutions that are improved along iterations called generations, by operators that simulates crossover, mutation and selection as observed in nature. The system is modelled based on 1st and 2nd laws of thermodynamics with implementation in Python. From a standard case with municipal solid waste (MSW) with no costs and 3.89 years of payback, optimization by RCGA leads to a 32,90% reduction, reaching 2.61 years. In this scenario, when compared with the optimal case by classical exergy analysis, with 3.06 years of payback, the reduction is of 14.71%. The optimization with RCGA results show that the exergetic efficiency of some equipments is decreased. Some heat exchangers had their thermal exchange areas reduced in the optimal case. Optimization takes place by the 57,60% cash flow increase (BRL/h) and 5,57% increase in total capital investment cost (BRL). The solid waste flow (\dot{m}_{MSW}) increase from 525 kg/h to 700kg/h in optimal case.

Keywords: Optimization. Genetic Algorithm. Exergy. Exergoeconomy.

LISTA DE FIGURAS

Figura 1 – Classificação dos algoritmos de otimização	17
Figura 2 – Curvas de custos de um trocador de calor	19
Figura 3 – Fluxograma do procedimento de design ótimo	25
Figura 4 – Representação dos módulos que compõem o NPDEAS	32
Figura 5 – Ilustração da relação entre a entropia gerada e a exergia destruída durante um processo de transferência de calor	33
Figura 6 – Como funciona a seleção natural.	35
Figura 7 – Ciclo evolucionário	36
Figura 8 – Incinerador, ciclo Rankine e fotobiorreatores interligados	45
Figura 9 – Variação da taxa de custo associado ao gás de combustão e ao vapor através dos equipamentos	46
Figura 10 – Fluxograma do procedimento de análise e otimização exergoeconômica	49
Figura 11 – Fluxograma do processo evolutivo do RCGA	51
Figura 12 – Funções de Benchmark	55
Figura 13 – Resultado para a minimização de r_{HX2} em função da área de troca térmica de HX2	57
Figura 14 – Resultado para a minimização de r_{HX1} em função de T_{vap1}	58
Figura 15 – Melhores soluções encontradas em cada geração, em trinta execuções do RCGA para as soluções $(\mu + \lambda)$ e (μ, λ)	68
Figura 16 – Melhores soluções encontradas em cada geração, para trinta execuções do RCGA para as soluções $(\mu + \lambda)$ e (μ, λ)	68
Figura 17 – Desvio padrão do <i>payback</i> ao longo das gerações, em trinta execuções do RCGA para cada seleção de substituição	69
Figura 18 – Desvio padrão do <i>payback</i> ao longo das gerações, em trinta execuções do RCGA para cada seleção de substituição	70
Figura 19 – Melhores soluções encontradas em cada geração do RCGA, para as soluções $(\mu + \lambda)$ e (μ, λ)	70
Figura 20 – Melhores soluções encontradas em cada geração do RCGA, para as soluções $(\mu + \lambda)$ e (μ, λ)	71
Figura 21 – Desvio padrão do <i>payback</i> ao longo das gerações, em trinta execuções do RCGA para cada seleção de substituição	72
Figura 22 – Desvio padrão do <i>payback</i> ao longo das gerações, em trinta execuções do RCGA para cada seleção de substituição	72
Figura 23 – Taxas de custo do gás e vapor conforme passam pelos equipamentos do sistema	73
Figura 24 – Variação de r_{turb}	75

LISTA DE QUADROS

Quadro 1 – Funções de <i>Benchmark</i> para validação do RCGA	56
---	----

LISTA DE TABELAS

Tabela 1 – Condições para minimização de r_{HX2} em função do produto e combustíveis do equipamento HX2	57
Tabela 2 – Variáveis escolhidas por Galante (2019) para a otimização dos equipamentos e seus respectivos limites	59
Tabela 3 – Variáveis que determinam o ponto ótimo de cada um dos equipamentos	59
Tabela 4 – Variáveis escolhidas para a otimização via RCGA do sistema térmico em estudo	60
Tabela 5 – Comparação entre valores de temperatura, pressão e vazão mássica . .	64
Tabela 6 – Comparação entre valores de exergia específica, taxa de exergia, custo médio por unidade de exergia e taxa de custo	65
Tabela 7 – Resultados da otimização das funções de <i>Benchmark</i> com o RCGA, utilizando a seleção $(\mu + \lambda)$	66
Tabela 8 – Resultados da otimização das funções de <i>Benchmark</i> com o RCGA, utilizando a seleção (μ, λ)	67
Tabela 9 – Valores das variáveis para as soluções ótimas encontradas	73
Tabela 10 – Eficiência exergética e capital de investimento dos principais equipamentos do sistema	74
Tabela 11 – Variação das áreas de troca térmica entre os casos padrão e ótimo via RCGA	76
Tabela 12 – Variação de custos do sistema e <i>payback</i>	77

LISTA DE ABREVIATURAS E SIGLAS

CI	Capital de Investimento
DE	Evolução Diferencial (do inglês – <i>Differential Evolution</i>)
DNA	Ácido Desoxirribonucleico (do inglês – <i>Deoxyribonucleic Acid</i>)
EA	Algoritmo Evolucionário (do inglês – <i>Evolutionary Algorithm</i>)
EC	Computação Evolucionária (do inglês – <i>Evolutionary Computation</i>)
EES	<i>Engineering Equation Solver</i>
EP	Programação Evolucionária (do inglês – <i>Evolutionary Programming</i>)
ES	Estratégia Evolutiva (do inglês – <i>Evolutionary Strategy</i>)
FBR	Fotobiorreatores
FPS	<i>Fitness Proportionate Selection</i>
GA	Algoritmo Genético (do inglês – <i>Genetic Algorithm</i>)
GLP	Gás Liquefeito de Petróleo
NPDEAS	Núcleo de Pesquisa e Desenvolvimento de Energia Autossustentável
OM	Operação e Manutenção
PCS	Poder Calorífico Superior
NPDEAS	Núcleo de Pesquisa e Desenvolvimento de Energia Autossustentável
RCGA	Algoritmo Genético de Codificação Real (do inglês – <i>Real-Coded Genetic Algorithm</i>)
RSU	Resíduo Sólido Urbano
SA	Recozimento Simulado (do inglês – <i>Simulated Annealing</i>)
SGA	Algoritmo Genético Simples (do inglês – <i>Simple Genetic Algorithm</i>)
SBX	Cruzamento Binário simulado (do inglês – <i>Simulated Binary Crossover</i>)
SGA	Algoritmo Genético Simples (do inglês – <i>Simple Genetic Algorithm</i>)
WtE	Waste to Energy

LISTA DE SÍMBOLOS

B	Constante de ajuste do CI
c	Referente ao condensador
c	Custo médio por unidade de exergia
CBG	Combustível gasoso
CO_2	Dióxido de carbono
\dot{C}	Taxa de custo
D	Referente à destruição
e	Exergia específica
e^{ch}	Exergia química
\dot{E}	Taxa de exergia
f_k	Importância em custo
$f(x)$	Função objetivo
F	Referente ao combustível
g	Aceleração da gravidade
$g_k(x)$	Restrição de desigualdade
h	Entalpia específica
h^0	Entalpia padrão
$h_k(x)$	Restrição de igualdade
H	Fração mássica de hidrogênio em base seca no combustível
HX	Referente aos trocadores de calor do sistema
inc	Referente ao incinerador
L	Relacionado a perdas
\dot{m}	Vazão mássica
n_{gen}	Número máximo de gerações

n_{pop}	Tamanho da população
N	Fração mássica de nitrogênio em base seca no combustível
O	Fração mássica de oxigênio em base seca no combustível
$poscomb$	Referente à câmara de pós combustão
p	Referente ao produto
p_c	Probabilidade de cruzamento
p_m	Probabilidade de mutação
r_k	Custo relativo de um equipamento k
s	Entalpia específica
s^0	Entalpia padrão
tot	Referente ao total
$turb$	Referente à turbina
t	Geração ou iteração
T	Temperatura
V	Velocidade
x_i	Um determinado gene de um indivíduo/cromossomo
x	Referente a um indivíduo na população
Δ	Referente a diferença ou variação
ε_E	Eficiência exergética
λ	Número de filhos gerados
μ	Número de indivíduos na população atual
η_c	Índice de distribuição do operador de cruzamento
η_m	Índice de distribuição do operador de mutação
η_{turb}	Eficiência da turbina

LISTA DE ALGORITMOS

Algoritmo 1 – Algoritmo Evolucionário	36
Algoritmo 2 – Algoritmo Genético	39

SUMÁRIO

1 – INTRODUÇÃO	16
1.1 JUSTIFICATIVA	20
1.2 OBJETIVOS	22
1.2.1 Objetivo Geral	22
1.2.2 Objetivos Específicos	22
1.3 ORGANIZAÇÃO DO TRABALHO	22
2 – FUNDAMENTAÇÃO TEÓRICA	23
2.1 CONCEITOS FUNDAMENTAIS DE OTIMIZAÇÃO	23
2.1.1 Conjunto Viável e Funções de Penalidade	26
2.2 MODELAGEM E OTIMIZAÇÃO DE SISTEMAS TÉRMICOS	28
2.3 SISTEMA DE GERAÇÃO DE ENERGIA POR INCINERAÇÃO DE RESÍDUOS COM FILTRO BIOLÓGICO DE EMISSÕES	31
2.3.1 Exergia	32
2.4 ALGORITMOS EVOLUCIONÁRIOS	34
2.5 ALGORITMOS GENÉTICOS	37
2.5.1 Seleção Parental	39
2.5.2 Algoritmo Genético de Codificação Real	40
2.5.3 Operadores de Cruzamento e Mutação	41
2.5.4 Estratégias de Substituição	43
3 – ASPECTOS METODOLÓGICOS	45
3.1 CONSIDERAÇÕES SOBRE O OBJETO DE ESTUDO	45
3.2 CONSIDERAÇÕES SOBRE O RCGA	50
3.2.1 População Inicial	50
3.2.2 Seleção Parental	52
3.2.3 Operador de Cruzamento	52
3.2.4 Operador de Mutação	53
3.2.5 Seleção de Substituição	54
3.3 IMPLEMENTAÇÃO E VALIDAÇÃO DO RCGA	54
3.4 OTIMIZAÇÃO TERMOECONÔMICA	56
3.4.1 Validação do Modelo	56
3.4.2 Restrições do Modelo	57
4 – RESULTADOS E DISCUSSÕES	63
4.1 VALIDAÇÃO DO MODELO	63
4.2 VALIDAÇÃO DO RCGA	66
4.3 OTIMIZAÇÃO TERMOECONÔMICA	67
5 – CONCLUSÃO	78

REFERÊNCIAS	80
-----------------------	----

Apêndices	85
-----------	----

APÊNDICE A–Real-Coded Genetic Algorithm For Double Stage Organic Rankine Cycle Exergy Optimization	86
--	----

APÊNDICE B–RCGA Script	97
----------------------------------	----

1 INTRODUÇÃO

Otimização é a tarefa de encontrar a melhor solução para um determinado problema. Geralmente procura-se por uma solução ótima global, que é a melhor solução para todo o espaço de busca. Problemas de otimização são encontrados em diversas áreas, desde a vida cotidiana à matemática, engenharia, ciências naturais e sociais e ciência da computação, por exemplo. Resumidamente, toda situação em que busca-se maximizar algo como lucro, ou minimizar algo como gasto energético, trata-se de um problema de otimização (KRAMER, 2017).

Problemas de otimização são abundantes nos mais diversos campos da engenharia. Otimização é recorrente em aplicações de projeto, modelagem, caracterização, controle, produção, entre muitas outras. Vários problemas da vida real possuem mais de uma solução e, assim, a otimização pode ser realizada encontrando-se o melhor conjunto de parâmetros entre estas soluções, em termos de algum critério de desempenho. À melhor solução, dá-se o nome de solução ótima. Se o problema admite somente uma solução, apenas um conjunto de parâmetros é aceitável e, portanto, a otimização não pode ser executada (ANTONIOU; LU, 2007).

Nos problemas de otimização, um critério de performance F , geralmente chamado de função objetivo ou função de custo, deve ser escrito em termo de n parâmetros x_1, x_2, \dots, x_n como

$$F = f(x_1, x_2, \dots, x_n), \quad (1)$$

em que F é uma quantidade escalar que pode assumir várias formas e pode representar o custo de produção ou a diferença entre desempenho desejado e desempenho real de um sistema, por exemplo. As variáveis x_1, x_2, \dots, x_n são os parâmetros que influenciam no custo de produção no primeiro caso, ou no desempenho real no segundo caso. Essas variáveis, chamadas de variáveis ou parâmetros de otimização, podem ser independentes, como o tempo, ou parâmetros de controle que podem ser ajustados (ANTONIOU; LU, 2007).

O conjunto de técnicas, métodos, procedimentos e algoritmos utilizados para encontrar a solução ótima é chamada de prática de otimização. Algumas das abordagens gerais disponíveis para a solução de problemas de otimização são (ANTONIOU; LU, 2007):

1. Métodos analíticos;
2. Métodos gráficos;
3. Métodos experimentais;
4. Métodos numéricos.

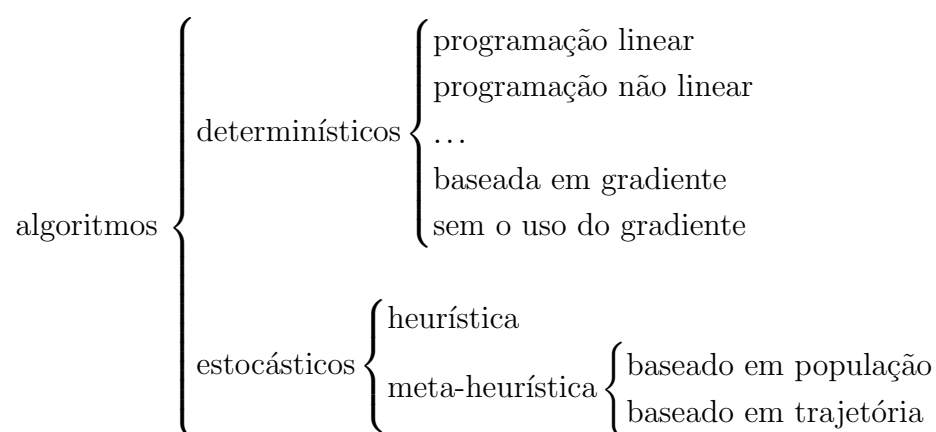
Os métodos analíticos, gráficos e experimentais possuem diversas limitações. O

método analítico é baseado no cálculo diferencial, depende do gradiente das funções objetivo, e não pode ser aplicado em problemas com um alto grau de não linearidade. O método gráfico depende da plotagem da função objetivo, não sendo possível ser aplicado em problemas com mais de duas variáveis de otimização. O método experimental pode levar à uma condição ótima de operação (ou muito próxima à ótima), mas depende de muitas execuções do experimento e não permite o ajuste simultâneo de vários parâmetros de otimização. Desta forma, o método numérico torna-se atrativo e, por isso, é a mais importante abordagem no processo de otimização (ANTONIOU; LU, 2007).

Na abordagem numérica, a busca pela solução ótima se inicia com uma estimativa inicial que soluciona o problema. Então, processos iterativos são utilizados para gerar soluções melhoradas. Os processos iterativos são encerrados quando algum critério de parada é alcançado. Um exemplo de critério de parada é quando a diferença entre as iterações torna-se insignificante (ANTONIOU; LU, 2007).

Diferentes algoritmos são formulados e estudados para a solução de problemas de otimização. Estes algoritmos dividem-se, basicamente, em determinísticos e estocásticos. Algoritmos determinísticos seguem procedimentos rigorosos e os valores das variáveis de otimização e das funções objetivo são repetíveis, isto é, para o mesmo ponto de partida, o mesmo caminho é repetido em diferentes execuções. Algoritmos estocásticos, por sua vez, estão atrelados à aleatoriedade. Nos algoritmos estocásticos, para cada execução, há um diferente caminho, mas o resultado final é igual ou suficientemente igual. A Figura 1 apresenta as classificações dos algoritmos de otimização (YANG, 2010).

Figura 1 – Classificação dos algoritmos de otimização



Fonte: Adaptado de Yang (2010)

Segundo Yang (2010), os algoritmos estocásticos se dividem em outras duas classes: heurística e meta-heurística, com pequenas diferenças. Rayward-Smith (1996) descreve uma técnica heurística como um método que busca por uma solução suficientemente próxima à solução ótima, com um custo computacional relativo, sem garantir que a solução ótima seja

alcançada. Embora os métodos heurísticos não garantam a solução ótima global, muitas técnicas heurísticas modernas produzem soluções de ótima qualidade.

A meta-heurística é resultado do desenvolvimento e aperfeiçoamento dos algoritmos heurísticos. Os algoritmos meta-heurísticos geralmente apresentam melhor desempenho que os algoritmos heurísticos simples. Além disso, os algoritmos meta-heurísticos caracterizam-se por usar randomização e busca local. A maior parte dos algoritmos meta-heurísticos são baseados em alguma abstração da natureza, a qual encontrou soluções ótimas para uma grande quantidade de problemas, através de milhões de anos de evolução. Os algoritmos meta-heurísticos podem ser classificados como baseados em população ou trajetória. Por exemplo, algoritmo genético (GA – do inglês *Genetic Algorithm*) é baseado em população, enquanto recozimento simulado (SA – do inglês *Simulated Annealing*) é baseado em trajetória (YANG, 2010).

Borne, Popescu e Philip (2014) descrevem os métodos meta-heurísticos como aqueles que podem ser implementados em um computador, baseando-se na busca de uma solução ótima pela simulação do comportamento de um sistema biológico ou evolução de um sistema natural.

Os métodos heurísticos devem ser considerados quando a opção por um método exato, que garante uma solução ótima, é impraticável. Nestes casos, os métodos heurísticos podem ser a única alternativa para a busca de soluções razoavelmente aceitáveis. Os motivos que justificam a utilização e promoção de técnicas heurísticas incluem (SALHI, 2017):

1. Podem ser a única forma de gerar soluções para problemas de larga escala, combinatórios e de otimização global.
2. São adaptáveis e acessíveis a tarefas adicionais ou restrições, caso necessário.
3. Podem ser facilmente suportadas por interface gráfica, o que pode auxiliar o usuário a visualizar e entender o andamento da busca, possibilitando a proposta de modificações.
4. Ao gerarem várias soluções, possibilitam a escolha de uma ou mais delas para análises e investigações do problema.

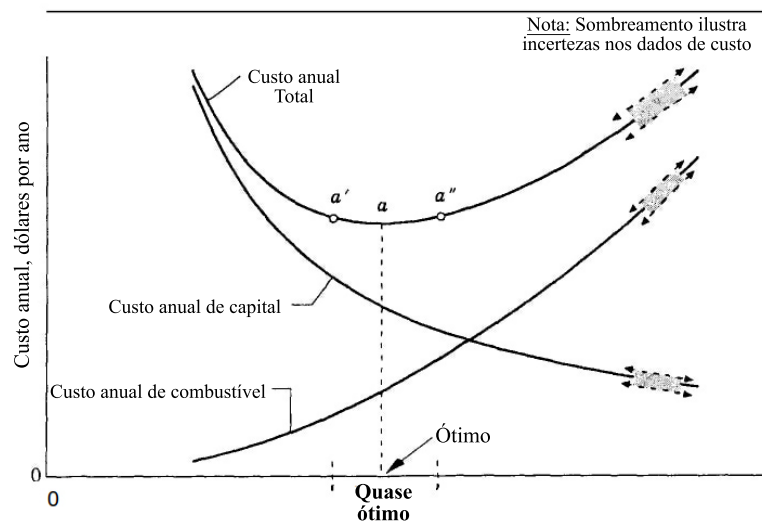
A otimização de sistemas térmicos é um exemplo que pode exigir a utilização de métodos meta-heurísticos. O problema de otimização de um sistema térmico pode apresentar um grande espaço de busca, ser não linear, não convexo, multimodal, multidimensional e implícito, além de poder não prover informações suficientes em relação ao gradiente. Nestas condições, métodos analíticos podem se tornar impraticáveis e estagnarem em ótimos locais. Deste modo, as técnicas metaheurísticas são frequentemente consideradas como a melhor abordagem para encontrar soluções suficientemente próximas à solução ótima global (PATEL; SAVSANI; TAWHID, 2019).

Para ilustrar a dificuldade em se determinar uma solução ótima global para um

sistema térmico, Bejan, Tsatsaronis e Moran (1995) utilizam o exemplo de um trocador de calor contra corrente, um equipamento comum em diversos sistemas térmicos. Nestes equipamentos, uma importante variável de decisão é a diferença mínima de temperatura entre os fluidos quente e frio (ΔT_{min}).

Para o exemplo, considera-se que o custo anual associado a um trocador de calor varia conforme ΔT_{min} varia, e mantém-se as outras variáveis fixas. Da engenharia termodinâmica, sabe-se que um trocador de calor se aproxima da idealidade conforme ΔT_{min} se aproxima de zero. Em um sistema que tem este trocador de calor como equipamento, o combustível que fornece energia para todo o sistema deve suprir as fontes de não idealidade do equipamento. Deste modo, um grande valor de ΔT_{min} corresponde a um trocador de calor menos ideal, aumentando o custo associado a esse combustível. Os custos anuais do trocador de calor e do combustível em relação a ΔT_{min} são mostrados na Figura 2 (BEJAN; TSATSARONIS; MORAN, 1995).

Figura 2 – Curvas de custos de um trocador de calor



Fonte: Bejan, Tsatsaronis e Moran (1995)

Em um primeiro momento, supõe-se que para minimizar os custos, basta reduzir a diferença de temperatura entre os fluidos quente e frio. Entretanto, pelo estudo de trocadores de calor, sabe-se que a minimização desta diferença de temperatura implica em um aumento da área de troca de calor, o que exige um trocador de calor maior e, conseqüentemente, mais caro (BEJAN; TSATSARONIS; MORAN, 1995).

Na Figura 2, observa-se que o custo anual associado à aquisição e manutenção do trocador de calor é inversamente proporcional a ΔT_{min} , enquanto o custo anual associado ao combustível é diretamente proporcional. O custo anual total de operação é a soma dos custos anuais associados ao trocador de calor e ao combustível. A curva do custo anual total apresenta concavidade voltada para cima, na qual o ponto designado pela letra *a*

representa o projeto com custo anual total mínimo, localizado em uma região entre o custo mínimo associado ao combustível e o custo mínimo associado ao trocador de calor. Projetos cujo custo anual total se localizam entre os pontos designados por a' e a'' podem ser considerados projetos quase ótimos (BEJAN; TSATSARONIS; MORAN, 1995).

Problemas de projeto de sistemas térmicos são ainda mais complexos que o exemplo apresentado. Primeiramente, custos não podem ser calculados com a precisão mostrada pelas curvas da Figura 2, pois dependem de previsões do preço do combustível, cuja variação depende de diversos fatores do mercado mundial. O custo de um trocador de calor além de poder depender de um processo de licitação, não varia de forma contínua. Outro fator apontado para o aumento da complexidade de sistemas térmicos diz respeito à configuração dos mesmos, que envolve vários equipamentos interagindo entre si. Nesta interação, a minimização do combustível em um equipamento pode, por exemplo, levar ao aumento de consumo de combustível em outro, ou no sistema como um todo. Além disso, diferentemente do exemplo apresentado, na otimização de um equipamento, mais de uma variável de otimização pode ser considerada. Em um sistema térmico composto por vários equipamentos, a otimização de cada equipamento individualmente geralmente não resulta na otimização do sistema (BEJAN; TSATSARONIS; MORAN, 1995).

Pelas propriedades anteriormente apresentadas, pela afirmação de Patel, Savsani e Tawhid (2019) e pelo exemplo abordado por Bejan, Tsatsaronis e Moran (1995), a utilização de métodos metaheurísticos torna-se atraente para problemas de otimização de sistemas térmicos. Desta forma, o presente trabalho propõe-se a apresentar um desses métodos como alternativa à otimização termoeconômica de um sistema de geração de energia por incineração de resíduos com filtro biológico de emissões, estudado por Galante (2019). O método de otimização proposto é o algoritmo genético de codificação real (RCGA – do inglês *Real-Coded Genetic Algorithm*). A razão da escolha por esse método de otimização é apresentada a seguir.

1.1 JUSTIFICATIVA

A proposta pela utilização de um método metaheurístico para a otimização do sistema térmico analisado e estudado por Galante (2019) se dá pela complexidade associada a este tipo de sistema. Como apresentado anteriormente e abordado por Bejan, Tsatsaronis e Moran (1995), tais sistemas são formados por vários equipamentos interagindo entre si, os quais podem possuir muitas variáveis de otimização, que precisam ser analisadas simultaneamente para a busca de uma solução ótima. Além disso, a complexidade envolve a previsão de custos associados ao combustível e aos próprios equipamentos que compõem o sistema.

Em seu estudo, Galante (2019) efetuou a análise e otimização exergoeconômica de um sistema de geração de energia por incineração de resíduos com filtro biológico

de emissões. O procedimento de otimização foi realizado pela minimização de *payback* (tempo de recuperação do capital investido), seguindo a metodologia descrita por Bejan, Tsatsaronis e Moran (1995), baseada na minimização do custo relativo de cada componente individualmente. Como resultado, Galante (2019) obteve uma redução de *payback* de 4,18 anos para 3,19 anos, ao considerar custos associados ao resíduo sólido urbano (RSU), combustível do sistema, e concluiu que a otimização individual de apenas quatro dos sete equipamentos considerados levaram a uma redução do *payback*. Ao considerar RSU sem custo, a redução foi de 3,89 anos de *payback* para 3,06 anos.

Para Yang (2010), as vantagens na utilização de GAs envolvem a capacidade desses métodos em solucionar problemas complexos, sejam eles lineares ou não lineares, contínuos ou descontínuos e a possibilidade de implementação de paralelismo. De acordo com Sivanandam e Deepa (2007), o GA é uma poderosa ferramenta de otimização, principalmente pela sua robustez, isto é, sua capacidade em solucionar de forma consistente uma gama enorme de problemas. Haupt e Haupt (1998) afirma que algumas das vantagens dos GAs incluem: otimização com variáveis discretas ou contínuas; não requerem informações da derivada da função; conseguem lidar com um número grande de variáveis e geram uma lista de soluções, não somente uma.

Para justificar a proposta de utilização de um RCGA para a otimização de um sistema térmico, é necessário esclarecer que este método se difere de um GA comum, principalmente pela sua forma de codificação. Em um RCGA, as variáveis de decisão são codificadas como números reais ao invés da codificação binária. Nos computadores, os números reais são representados por ponto flutuante e, segundo Michalewicz (1996), este tipo de representação torna o GA mais eficiente quando o mesmo é aplicado em problemas multidimensionais, de domínio contínuo e de alta precisão.

Apresentadas algumas das principais vantagens dos GAs, a escolha por este método para a otimização do sistema analisado por Galante (2019) se justifica principalmente pela facilidade de implementação e pela possibilidade de variar mais parâmetros de otimização simultaneamente, o que permite a realização da otimização do sistema de maneira integral. Tendo em vista que a otimização de cada equipamento individualmente não implica necessariamente na otimização do sistema, variar mais parâmetros de otimização simultaneamente pode produzir melhores soluções.

Considerando que o problema de otimização de sistemas térmicos é geralmente analisado com variáveis contínuas ou discretas, a escolha pela codificação com números representados por ponto flutuante é mais sensata. Este tipo de codificação permite que os valores das variáveis sejam implementados diretamente como números reais, sem a necessidade de algum tipo de conversão pelo usuário ou pelo algoritmo. No entanto, a codificação com números reais exige a utilização de operadores genéticos diferentes dos usados nos GAs comuns de codificação binária.

1.2 OBJETIVOS

1.2.1 Objetivo Geral

O objetivo do presente trabalho é realizar a otimização de um sistema térmico utilizando um RCGA. Este RCGA será aplicado no problema de otimização termoeconômica da mesma planta térmica analisada e otimizada no trabalho intitulado *Análise e Otimização Termoeconômica de Sistemas de Geração de Energia por Incineração de Resíduos com Filtro Biológico de Emissões*, por Galante (2019). O referido sistema térmico trata-se de uma planta integrada de geração de potência que aproveita a energia térmica resultante da incineração de resíduos sólidos provenientes de um campus universitário.

1.2.2 Objetivos Específicos

Visando atingir o objetivo geral, define-se para o presente trabalho os seguintes objetivos específicos:

- Adaptar e implementar um modelo termodinâmico para o sistema a ser analisado;
- Desenvolver e implementar o RCGA;
- Verificar a convergência do RCGA;
- Avaliar as soluções encontradas.

1.3 ORGANIZAÇÃO DO TRABALHO

O presente trabalho é organizado em cinco capítulos. No Capítulo 1, encontra-se a Introdução, onde é apresentado um contexto inicial sobre otimização, algoritmos de otimização e a dificuldade envolvida na otimização de sistemas térmicos, além das justificativas e objetivos do presente trabalho. O Capítulo 2 contém a Fundamentação Teórica, onde são apresentados conceitos fundamentais de otimização, uma contextualização sobre modelagem e otimização de sistemas térmicos e trabalhos já reproduzidos neste assunto. Além disso, na Fundamentação Teórica se encontram às seções referentes a explicação do sistema de geração de energia por incineração de resíduos com filtro biológico de emissões, objeto de estudo do presente trabalho, e de sua modelagem. Posteriormente são apresentados os algoritmos evolucionários (EAs – do inglês *Evolutionary Algorithms*), seguidos de um contextualização sobre GAs e, principalmente, RCGAs e seus operadores genéticos. No Capítulo 3, denominado Aspectos Metodológicos, encontra-se a explicação da metodologia adotada para a implementação e validação do modelo do sistema térmico e do RCGA, além do procedimento de otimização desse sistema por este algoritmo. No Capítulo 4 são apresentados os resultados da implementação do modelo, da validação do RCGA, e da otimização do modelo termodinâmico por meio deste algoritmo. Por fim, no Capítulo 5 são apresentadas as considerações finais, junto às sugestões para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo abordará os conceitos necessários para o entendimento e desenvolvimento do estudo realizado no presente trabalho, cujo principal objetivo é avaliar a utilização de um RCGA na otimização de um sistema térmico. Primeiramente, conceitos fundamentais de otimização são abordados. Em seguida, faz-se uma breve discussão sobre problemas de otimização de sistemas térmicos e, posteriormente, a planta térmica a ser analisada é apresentada junto com a metodologia utilizada por Galante (2019) para a sua modelagem, análise e otimização termoeconômica. Depois, a explicação partirá para uma breve introdução ao tema EAs, uma família de algoritmos da qual os GAs fazem parte, e em seguida, GAs são apresentados dando enfoque ao RCGA e seus operadores genéticos.

É válido destacar que, embora os GAs tenham surgido como estudo e implementação dos mecanismo de seleção natural em computador, ao serem utilizados como método de otimização, os GAs não têm intenção de explicar o fenômeno da evolução. Mesmo assim, como métodos de otimização bioinspirados, os GAs emprestam alguns conceitos básicos da genética, mas a explicação desses conceitos, apresentada nesse trabalho, têm o único intuito de facilitar o entendimento do processo de otimização por um GA, não devendo estender-se para o entendimento da genética, da seleção natural e da evolução em um contexto biológico amplo e detalhado, sendo meramente uma abstração.

2.1 CONCEITOS FUNDAMENTAIS DE OTIMIZAÇÃO

Uma etapa inicial crítica no projeto de sistemas é definir a quais requisitos o sistema deve atender, e expressá-los de forma quantitativa, isto é, formular as especificações do mesmo. Um projeto que satisfaz todas as especificações do sistema é dito viável. Dentre os sistemas viáveis existe um projeto ótimo, que é considerado o melhor em relação a alguma especificação: custo, tamanho, peso, confiabilidade, entre outras (BEJAN; TSATSARONIS; MORAN, 1995).

Em projetos industriais, por muitas vezes um projeto simplório é escolhido comparando-se uma quantidade bastante limitada de soluções alternativas, criadas a partir de um conhecimento prévio do problema. Nesse processo, a viabilidade de cada solução é analisada e uma estimativa do objetivo (lucro, custo, etc.) de cada solução é calculada, e a melhor solução baseada nessa estimativa é escolhida. Este processo é frequentemente utilizado por limitação de tempo e/ou recurso. No entanto, muitas vezes este procedimento é escolhido somente por desconhecimento dos procedimentos e algoritmos de otimização existentes. Independentemente do motivo, um produto de qualidade e competitivo nem sempre é garantido por este método (DEB, 2012).

No processo de projeto, o projetista estima um projeto de teste para o sistema baseado em experiência, intuição, ou alguma análise matemática simples. O projeto de teste é então analisado para verificar se o mesmo é aceitável. Caso seja aceitável, o processo de projeto é encerrado. No processo de otimização, o projeto de teste é analisado para verificar se o mesmo é o melhor, e não somente aceitável, em relação ao custo-benefício, eficiência, confiabilidade, durabilidade, etc., a depender das especificações (ARORA, 2017).

Uma representação matemática geral para o problema de otimização de um projeto é definido como a minimização de uma função objetivo, satisfazendo todas as restrições do problema. Neste modelo, todas as restrições de desigualdade são transformadas como do tipo menor ou igual (\leq). Na literatura, este modelo padrão é chamado de problema de programação não linear (NLP – do inglês *Nonlinear Programming Problem*) (ARORA, 2017).

Para Deb (2012), o propósito do procedimento de formulação, isto é, escrever o problema de otimização conforme um NLP, é criar um modelo matemático para o problema de otimização que então pode ser resolvido por meio de um algoritmo de otimização. Na Figura 3 é apresentado um esboço das etapas geralmente envolvidas no procedimento de formulação de um projeto como problema de otimização.

Conforme as etapas da Figura 3, o procedimento de formulação se inicia pela percepção da necessidade de utilização de otimização num problema de projeto. Em seguida, o projetista deve escolher as variáveis de otimização associadas ao problema. A formulação também envolve outras etapas e considerações, como restrições, função objetivo, e limites das variáveis de otimização. Todas essas etapas são dependentes umas das outras, mas o projetista pode atualizar as variáveis de otimização, função objetivo, restrições e limites das variáveis algumas vezes até que se sinta satisfeito com a formulação. Algumas iterações possíveis são representadas pelas flechas do fluxograma (DEB, 2012).

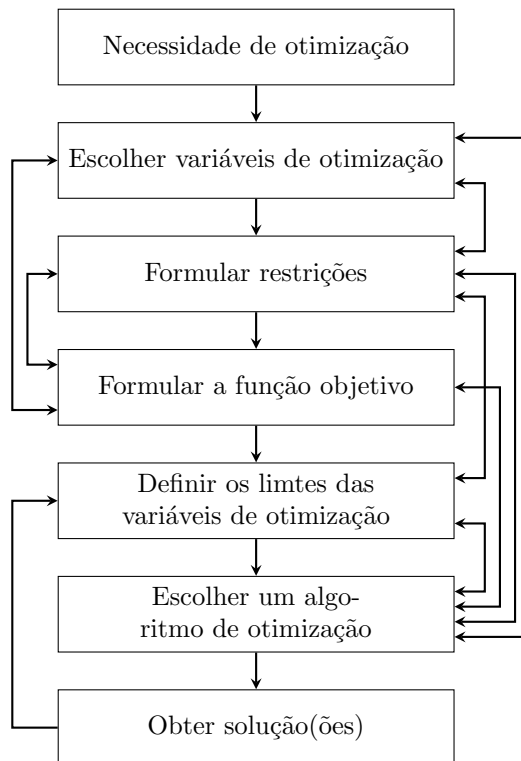
Em relação às etapas geralmente envolvidas no procedimento de formulação, seguem as seguintes considerações, conforme Deb (2012):

- Não existe uma regra para escolher os parâmetros que podem ser importantes em um problema, um parâmetro pode ser significativo em relação a minimização do custo total, mas insignificante quanto à maximização da vida útil, por exemplo. A escolha pelas variáveis de otimização depende muito do usuário e do objetivo. Os parâmetros escolhidos devem ser variados durante o processo de otimização.
- As restrições representam algumas relações funcionais entre algumas variáveis e outros parâmetros de projeto, satisfazendo alguns fenômenos físicos e limitações de recursos. Algumas especificações de projeto podem requisitar a permanência em equilíbrio estático ou dinâmico. Em alguns projetos de Engenharia Mecânica e Civil, por exemplo, as restrições devem satisfazer limites de tensão e deflexão.
- O objetivo do projeto (minimização de peso, custo ou maximização de lucro, vida

útil, etc.) deve ser escrito como uma função (chamada função objetivo) em termos das variáveis de otimização e outros parâmetros. Geralmente, não é necessário que a função objetivo seja expressa de forma matemática. Em muitos casos é necessário a utilização de pacotes de simulação para calcular e avaliar a função objetivo.

- Em alguns problemas de otimização, a busca pela solução ótima deve estar confinada aos limites inferiores e superiores das variáveis de otimização. Geralmente, todas as N variáveis de otimização devem estar entre os limites mínimos e máximos.

Figura 3 – Fluxograma do procedimento de design ótimo



Fonte: Adaptado de Deb (2012)

O modelo padrão do problema de otimização ou NLP é descrito, conforme Arora (2017), por:

$$\begin{aligned}
 \min \quad & f(x) = f(x_1, x_2, \dots, x_n), \\
 \text{sujeita a} \quad & h_j(x) = h_j(x_1, x_2, \dots, x_n) = 0, \quad (j = 1, 2, \dots, J), \\
 & g_k(x) = g_k(x_1, x_2, \dots, x_n) \leq 0, \quad (k = 1, 2, \dots, K),
 \end{aligned} \tag{2}$$

em que $f(x)$ refere-se à função objetivo, $h_j(x)$ refere-se às J restrições de igualdade, e $g_k(x)$ refere-se às K restrições de desigualdade.

No modelo padrão, apenas os problemas de minimização são tratados. Um problema de maximização de uma função $F(x)$ pode ser abordado como a minimização de

uma função transformada $f(x) = -F(x)$. As restrições do tipo maior ou igual (\geq) podem ser facilmente convertidas para a forma padrão. A restrição $G_k(x) \geq 0$ é equivalente à restrição do tipo menor ou igual $g_k(x) = -G_k(x) \leq 0$. Portanto, é possível multiplicar qualquer restrição do tipo maior ou igual por -1 para convertê-la para o tipo menor ou igual. Considera-se que as restrições dos limites mínimo e máximo das variáveis de otimização são incluídos no conjunto de desigualdades da Equação (2) (ARORA, 2017).

Deb (2012) afirma que os objetivos e as variáveis de otimização associadas a um problema de otimização variam de produto para produto, e diferentes técnicas precisam ser utilizadas em diferentes problemas. O autor também afirma que certos problemas de otimização na engenharia envolvem termos lineares para as restrições e funções objetivos, enquanto outros problemas apresentam termos não lineares, e que em alguns casos as funções e restrições não são explícitas em relação às variáveis de otimização.

Infelizmente, não há um único algoritmo de otimização que funciona para todos os problemas de otimização de forma eficiente. Alguns algoritmos apresentam melhor desempenho em um problema específico, mas têm péssimo desempenho em outros. A literatura de otimização contém um grande número de algoritmos, cada um mais adequado para a resolução de um tipo específico de problema. A escolha de um algoritmo de otimização adequado depende muito da experiência do usuário em resolver problemas similares (DEB, 2012).

2.1.1 Conjunto Viável e Funções de Penalidade

A solução de um problema é um conjunto de valores numéricos atribuídos às variáveis de otimização (por exemplo, o vetor coluna $x = (x_1, x_2, \dots, x_n)$). Mesmo que a solução gere resultado absurdo (por exemplo, raio negativo) ou inadequado em relação à função objetivo, esta ainda pode ser considerada uma solução. É óbvio que algumas soluções podem ser úteis e outras, nem tanto. Uma solução que satisfaz todos os requisitos do sistema é chamada de solução viável. Se a solução não satisfaz um ou mais requisitos, a mesma é considerada inviável (ARORA, 2017).

A coleção de todas as soluções viáveis é chamada de conjunto viável. Matematicamente, o conjunto viável, representado por S , é o conjunto de soluções que satisfazem todas as restrições do sistema, conforme segue (ARORA, 2017):

$$S = \{x \mid h_j(x) = 0, j = 1, 2, \dots, J; g_k(x) \leq 0, i = 1, 2, \dots, K\} \quad (3)$$

Um problema restrito pode ser aproximado de um problema sem restrições por meio de métodos de barreira e penalidade. No caso dos métodos de penalidade, a aproximação é realizada adicionando à função objetivo um alto valor pela violação das restrições. No método de barreira, adiciona-se um valor à função objetivo que favorece as soluções

interiores em relação àqueles próximas à barreira do conjunto viável (LUENBERGER, 2016).

Segundo Freund (2004), em um método de penalidade, o conjunto viável do problema de otimização é expandido de S , dado pela Equação (3), para todo o \mathbb{R}^n , mas um grande valor de penalidade é acrescentado à função objetivo de soluções que localizam-se fora do conjunto viável original S . O autor ainda afirma que, em um método de barreira, dada uma solução localizada no interior do conjunto viável S , acrescenta-se um alto valor à função objetivo de soluções viáveis conforme essas se aproximam da fronteira de S , criando assim uma barreira para sair do conjunto viável.

Coello (2002) afirma que, dentro da comunidade de usuários de EAs, as funções de penalidade representam a principal abordagem para se lidar com as restrições dos problemas de otimização. De forma diferente de Freund (2004), Coello (2002) classifica o método que se inicia com uma solução inviável e move-se em direção ao interior do conjunto viável, como função de penalidade exterior; já o método que penaliza os indivíduos viáveis conforme estes se aproximam da fronteira do conjunto viável, como função de penalidade interior.

Coello (2002) disserta sobre algumas das funções de penalidade existentes: penalidades estáticas, penalidades dinâmicas, penalidade de recozimento (baseada no método *simulated annealing*), penalidades adaptativas, penalidades coevolucionárias, GA segregado e pena de morte (do inglês – *death penalty*). Esta última é a função de penalidade escolhida para o RCGA do presente trabalho, que será utilizado na otimização da planta integrada de geração de potência estudada por Galante (2019).

A pena de morte é uma função de penalidade baseada na rejeição das soluções inviáveis. É provavelmente a forma mais fácil de se lidar com as restrições dos problemas de otimização, e é eficiente em relação ao custo computacional, porque quando uma solução é inviável, simplesmente atribui-se um alto valor para a função objetivo. Nesta abordagem, nenhum tipo de cálculo é necessário para estimar o grau de violação de uma solução (COELLO, 2002).

Além do inconveniente de não utilizar nenhum tipo de informação dos pontos inviáveis para guiar a busca pela solução ótima, um possível problema da abordagem da pena de morte é que se não houver soluções viáveis nas iterações iniciais, o algoritmo de otimização pode estagnar, já que todas as soluções terão o mesmo valor. Existem experimentos bem documentados nos quais o uso da pena de morte não é recomendável, que mostram que as funções de penalidade definidas em termos de distância ao conjunto viável superam a pena de morte (COELLO, 2002).

2.2 MODELAGEM E OTIMIZAÇÃO DE SISTEMAS TÉRMICOS

Todo sistema cuja operação consiste em converter energia térmica em energia mecânica, ou em quaisquer outras formas de energia, é considerado um sistema térmico. A energia térmica pode ser disponibilizada ao sistema pela queima de combustíveis, por energia solar, por energia geotérmica, entre outros meios. Essa energia térmica é fornecida ao fluido de trabalho do sistema, que passa por diversos processos antes de ser convertido para outras formas de energia. Esses processos de conversão acontecem em diferentes equipamentos do sistema. Conforme o fluido de trabalho passa por estes equipamentos, sua pressão, temperatura, volume e outras propriedades termodinâmicas como densidade, viscosidade, condutividade térmica, entre outras, mudam constantemente. Desta forma, para que o sistema opere de forma apropriada, a modelagem e o projeto de cada equipamento do sistema têm grande importância (PATEL; SAVSANI; TAWHID, 2019).

Ao se projetar um sistema térmico, inclui-se um processo de otimização na qual os projetistas consideram algum objetivo, como transferência de calor, eficiência térmica, capacidade de resfriamento, perda de carga, etc., a depender dos requisitos. Além dos objetivos mencionados anteriormente, é razoável minimizar o custo total do sistema. Deste modo, a otimização do projeto de um sistema térmico completo resulta em uma função objetivo complexa, com um grande número de variáveis e restrições de projeto. Os métodos tradicionais de otimização consistem em um processo iterativo considerando as especificações do sistema, e assumindo variáveis de decisão para diversas configurações, até encontrar uma forma de atender os requisitos do sistema dentro de um conjunto de restrições. Geralmente, essa abordagem resulta em equipamentos superdimensionados, além do procedimento de projeto ser complicado (PATEL; SAVSANI; TAWHID, 2019).

Como mencionado anteriormente na Introdução, Bejan, Tsatsaronis e Moran (1995) ilustram a complexidade que envolve a otimização de sistemas térmicos ao apresentar o caso de um trocador de calor. Neste exemplo, considera-se apenas uma variável de otimização: a diferença mínima de temperatura entre os fluidos quente e frio (ΔT_{min}). É observado que o custo anual associado à aquisição e manutenção do trocador de calor é inversamente proporcional a ΔT_{min} , enquanto o custo anual associado ao combustível é diretamente proporcional. Ao se analisar graficamente as curvas de custos associados ao trocador de calor e ao combustível e a curva de custo anual total, apresentado na Figura 2, a solução ótima, ou seja, a que apresenta menor custo anual total, se localiza em uma região entre os custos mínimos associados ao trocador de calor e ao combustível. Bejan, Tsatsaronis e Moran (1995) afirmam com este exemplo que projetos de sistemas térmicos são ainda mais complexos, pois os mesmos são formado por diversos equipamentos interagindo entre si, muitas vezes com mais de uma variável de decisão, e que os cálculos de custos dependem de diversos fatores de mercado e de algumas previsões e indicadores.

Em problemas reais de projeto, o número de variáveis de decisão pode ser muito

grande, e a influência dessas variáveis na função objetivo pode ser imprevisível, com comportamento não linear. Além disso, a função objetivo pode apresentar muitos ótimos locais. Nesses casos, torna-se atrativa a utilização de algoritmos avançados de otimização, que encontram soluções suficientemente próximas do ótimo global, com custo computacional e tempo razoáveis (PATEL; SAVSANI; TAWHID, 2019).

Na Introdução foi apresentada a classificação dos algoritmos de otimização segundo Yang (2010). Considerando essa mesma classificação, o presente trabalho tem como proposta a utilização de um algoritmo mataheurístico baseado em população, conhecido como GA, para a otimização do sistema térmico apresentado por Galante (2019). A escolha pelo GA se dá principalmente pela facilidade de implementação e pela possibilidade de variar muitos parâmetros de otimização de forma simultânea. O problema é analisado em um domínio contínuo, ou seja, os parâmetros de otimização são considerados números reais, e desta forma utiliza-se um tipo específico de GA, o RCGA, codificado com números representados por ponto flutuante. A utilização de GAs em problemas de otimização de sistemas térmicos não é incomum e alguns exemplos são mostrados a seguir.

Mohammadkhani, Khalilarya e Mirzaee (2011) utilizaram um GA na otimização de um sistema de cogeração com motor de combustão interna. O sistema estudado pelos autores produz 251 kW de eletricidade e eleva a temperatura da água (fluido de trabalho) de 80°C para 120°C a 2 bar de pressão e vazão mássica de 1,75 kg/s, e é otimizado usando princípios de exergoeconomia. A função objetivo utilizada representa o custo total da planta em dólar por segundo (\$/s), definida pela soma dos custos associados ao consumo de combustível e o capital de investimento para compra, operação e manutenção dos equipamentos, além do custo associado à destruição de exergia. A aplicação do GA resultou em uma redução de custo de 15,1% (MOHAMMADKHANI; KHALILARYA; MIRZAEI, 2011).

Nazari, Heidarnejad e Porkhial (2016) propuseram um ciclo combinado para recuperar o calor dos gases de exaustão gerados por uma turbina. O sistema combinado proposto inclui um ciclo Rankine a vapor subcrítico acoplado a um ciclo Rankine orgânico transcrito. Considerou-se os fluidos R124, R152a e R134a para monitorar a performance termodinâmica e exergoeconômica do sistema. Assume-se R124 e R152a para um caso base com 52,62% de eficiência térmica e 396,7 \$/h de taxa de custo total de produto. Um estudo dos parâmetros de otimização é realizado e, por fim, o sistema é otimizado via GA, considerando duas funções objetivo: eficiência exergética e taxa de custo total de produto do sistema. Os resultados apontam que o sistema tem melhor performance utilizando apenas o R152a, do ponto de vista exergoeconômico e termodinâmico (NAZARI; HEIDARNEJAD; PORKHIAL, 2016).

Patel, Savsani e Tawhid (2019) testaram onze diferentes algoritmos, incluindo um GA de codificação binária, na otimização de trinta e seis equipamentos e sistemas

térmicos. Os algoritmos são comparados pelo teste de Friedman (1937), considerando o valor da média, do melhor, do pior e a taxa de sucesso (0,1% de variação em relação ao ótimo global), em cem execuções. O GA apresenta um desempenho relativamente ruim, ficando em primeiro somente na otimização de um sistema de refrigeração em cascata, e em segundo na otimização de uma torre solar. A Evolução Diferencial (DE – do inglês *Differential Evolution*) e o Algoritmo de Busca Cuco (CSA – *Cuckoo Search Algorithm*) se destacaram como os melhores métodos, ficando ambos em primeiro, na otimização de dez sistemas térmicos. Vale ressaltar que o GA testado por Patel, Savsani e Tawhid (2019) utiliza codificação binária, diferentemente do GA do presente trabalho.

Prado e Galante (2020) propuseram um RCGA como alternativa ao algoritmo de força bruta (busca exaustiva), utilizado por Braimakis e Karellas (2018), na otimização de um ciclo Rankine orgânico de duplo estágio (DS ORC – do inglês *Double Stage Organic Rankine Cycle*). Prado e Galante (2020) modelaram o DS ORC por dois ciclos Rankine orgânicos simples conectados em série, utilizando a mesma metodologia de Braimakis e Karellas (2018). A otimização com o RCGA é executada considerando as mesmas variáveis de otimização de Braimakis e Karellas (2018), com adição da temperatura da fonte quente, e utilizando a eficiência exergética como função objetivo. Os resultados mostraram que o RCGA é capaz de encontrar resultados similares aos encontrados pelo algoritmo de força bruta. Os resultados além de validarem a otimização via RCGA, apontam que ao utilizar a temperatura da fonte quente como variável de decisão, o processo de otimização tende a levar a uma configuração de único estágio. No entanto, ao manter a temperatura da fonte quente fixada em determinados valores, a otimização com o RCGA resultou em cenários nos quais a configuração de duplo estágio é mantida, o que também é discutido por Braimakis e Karellas (2018). O artigo deste estudo é apresentado no Apêndice A.

Tendo em vista a possibilidade da utilização de GAs na otimização de sistemas térmicos, a performance de um RCGA será testada na otimização de um sistema de geração de energia por incineração de resíduos com filtro biológico de emissões, analisado e otimizado exergoeconomicamente por Galante (2019). A otimização via RCGA é proposta como alternativa à abordagem utilizada por Galante (2019), que segue metodologia proposta por Bejan, Tsatsaronis e Moran (1995), baseada na análise exergoeconômica e minimização da diferença de custo relativo de cada equipamento (r_k). O presente trabalho utilizará a mesma modelagem realizada por Galante (2019), diferindo-se na metodologia apenas pela abordagem no processo de otimização, em substituição à análise exergoeconômica e minimização de r_k . A planta desse sistema térmico, objeto de estudo do presente trabalho, é apresentada na próxima seção.

2.3 SISTEMA DE GERAÇÃO DE ENERGIA POR INCINERAÇÃO DE RESÍDUOS COM FILTRO BIOLÓGICO DE EMISSÕES

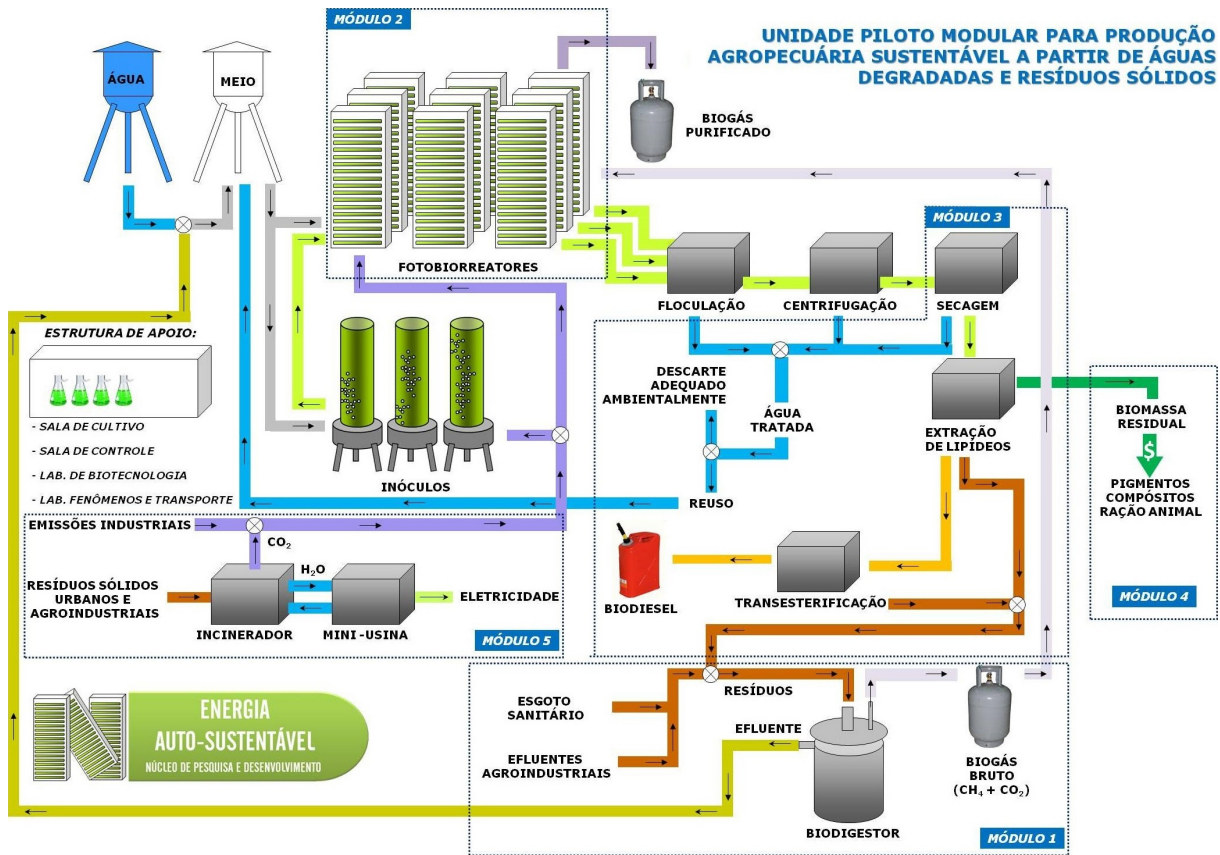
O presente trabalho utiliza como objeto de estudo os módulos do sistema integrado do Núcleo de Pesquisa e Desenvolvimento de Energia Autossustentável (NPDEAS) analisados e otimizados exergoeconomicamente por Galante (2019), na tese intitulada *Análise e Otimização Termoconômica de Sistemas de Geração de Energia por Incineração de Resíduos com Filtro Biológico de Emissões*. O sistema integrado e o estudo realizado por Galante (2019) serão apresentados nesta seção.

O sistema integrado do NPDEAS, localizado no campus Politécnico da Universidade Federal do Paraná, é uma proposta de geração de energias renováveis utilizando resíduo sólido urbano (RSU). Esta proposta visa a geração de eletricidade e biocombustíveis de forma integrada e sustentável. O sistema integrado e seus respectivos módulos são ilustrados na Figura 4 (GALANTE, 2019).

O trabalho de Galante (2019) analisa o módulo 5 integrado ao módulo 2, atentando-se às entradas e saídas do sistema e às eficiências de conversão de energia. No módulo 5, RSU é incinerado com a abordagem WtE (*Waste to Energy*), ou seja, aproveitando-se a energia térmica dos RSU para cogeração. Esse aproveitamento de energia utiliza um ciclo Rankine com água como fluido de trabalho, cujos gases de combustão são direcionados ao módulo 2, provendo CO_2 para microalgas, reduzindo a liberação de gases de efeito estufa do sistema e auxiliando na geração de biomassa usada para a produção de biodiesel. A visualização dos fluxos de exergia ao longo do sistema é feita por análise termodinâmica, o que permite a otimização de parâmetros atrelados a fatores econômicos. Esse tipo de abordagem é chamado de análise exergoeconômica, que permite a verificação da viabilidade técnica e financeira do projeto (GALANTE, 2019).

O módulo 5 integrado ao módulo 2 contempla a unidade de incineração e ventilação, a caldeira, trocadores de calor, o ciclo Rankine e a unidade dos fotobiorreatores (FBR). O sistema composto pelo módulo 5 integrado ao módulo 2 é otimizado por Galante (2019) pela minimização de *payback*, seguindo uma abordagem clássica de análise exergoeconômica, conforme Bejan, Tsatsaronis e Moran (1995). Para melhor entendimento da análise exergoeconômica performada no estudo de Galante (2019), o conceito de exergia deve ser introduzido.

Figura 4 – Representação dos módulos que compõem o NPDEAS



Fonte: Galante (2019)

2.3.1 Exergia

A exergia é a medida do potencial de trabalho, também chamada de disponibilidade ou energia disponível, que nada mais é do que o máximo trabalho útil que pode ser obtido do sistema (ÇENGEL; BOLES, 2013).

A exergia também pode ser entendida como a medida do quanto o estado de um sistema se afasta do estado morto (condição ambiente), sendo assim um atributo do sistema e do ambiente em que o sistema se encontra. Diferentemente da energia que não pode ser destruída, a exergia se destrói conforme a entropia do sistema aumenta. A relação entre o aumento da entropia e destruição de exergia é ilustrado na Figura 5 (GALANTE, 2019).

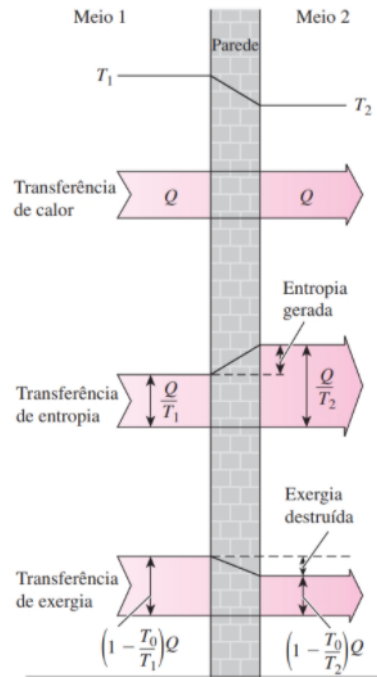
A exergia específica (kJ/s) associada a uma vazão mássica em um volume de controle é representada por

$$e = (h - h_0) - T_0 \cdot (s - s_0) + \frac{1}{2} \cdot V^2 + g \cdot z + e^{ch}, \quad (4)$$

onde os dois primeiros termos do lado direito desta equação correspondem a exergia física (função da temperatura e pressão). O terceiro e quarto termo correspondem à exergia associada às energias cinética e potencial gravitacional, que geralmente são consideradas

desprezíveis em sistemas estacionários. O último termo diz respeito à entrada de exergia no volume de controle a partir de um potencial químico. Esta exergia química (e^{ch}) é associada à entrada/saída de produtos/reagentes em reações químicas, como combustíveis (GALANTE, 2019).

Figura 5 – Ilustração da relação entre a entropia gerada e a exergia destruída durante um processo de transferência de calor



Fonte: Çengel e Boles (2013)

Em uma análise detalhada em todos os pontos de um processo térmico, a exergia de entrada, proveniente da fonte de energia, pode ser classificada como exergia do combustível, \dot{E}_F (kW), exergia dos produtos (\dot{E}_P), exergia destruída (\dot{E}_D), e exergia perdida (\dot{E}_L), que se relacionam do seguinte modo (GALANTE, 2019):

$$\dot{E}_F = \dot{E}_P + \dot{E}_D + \dot{E}_L \quad (5)$$

A exergia do combustível é entendida como a exergia proveniente de todos os elementos que darão origem ao produto. Numa câmara de combustão a soma da exergia do próprio combustível e do ar queimado é classificado como exergia do combustível. A exergia do produto será a exergia dos gases quentes de combustão. A eficiência de 2ª lei, ou eficiência exergética é determinada pela relação entre a exergia dos produtos e a exergia do combustível, conforme segue (GALANTE, 2019):

$$\varepsilon_E = \frac{\dot{E}_P}{\dot{E}_F} = 1 - \frac{\dot{E}_D + \dot{E}_L}{\dot{E}_F} \quad (6)$$

Entende-se eficiência exergetica (ϵ_E) como um quantificador da fração de exergia que foi de fato transferida do combustível para o produto, ou como a redução, a partir do 100% da exergia que foi perdida ou destruída. A exergia perdida é classificada como o potencial de trabalho não aproveitado devido as características próprias do sistema e do ambiente. A exergia destruída é referente ao potencial de trabalho não gerado devido a geração de entropia nos processos de troca de calor, válvulas de expansão e irreversibilidades dos equipamentos. As perdas de exergia podem ser reduzidas, mas isso acarreta em maiores custos de processo ou de equipamento (GALANTE, 2019).

2.4 ALGORITMOS EVOLUCIONÁRIOS

Como visto anteriormente, segundo Yang (2010), boa parte dos algoritmos metaheurísticos são inspirados na natureza, que em milhões de anos, através da seleção natural, encontrou soluções ótimas para uma enorme quantidade de problemas. Nesse mesmo sentido, Brabazon, O'Neill e McGarraghy (2015) afirmam que a evolução biológica se apresenta como um solucionador de problemas, que cria soluções suficientemente boas para executar o trabalho de sobrevivência em um determinado ambiente.

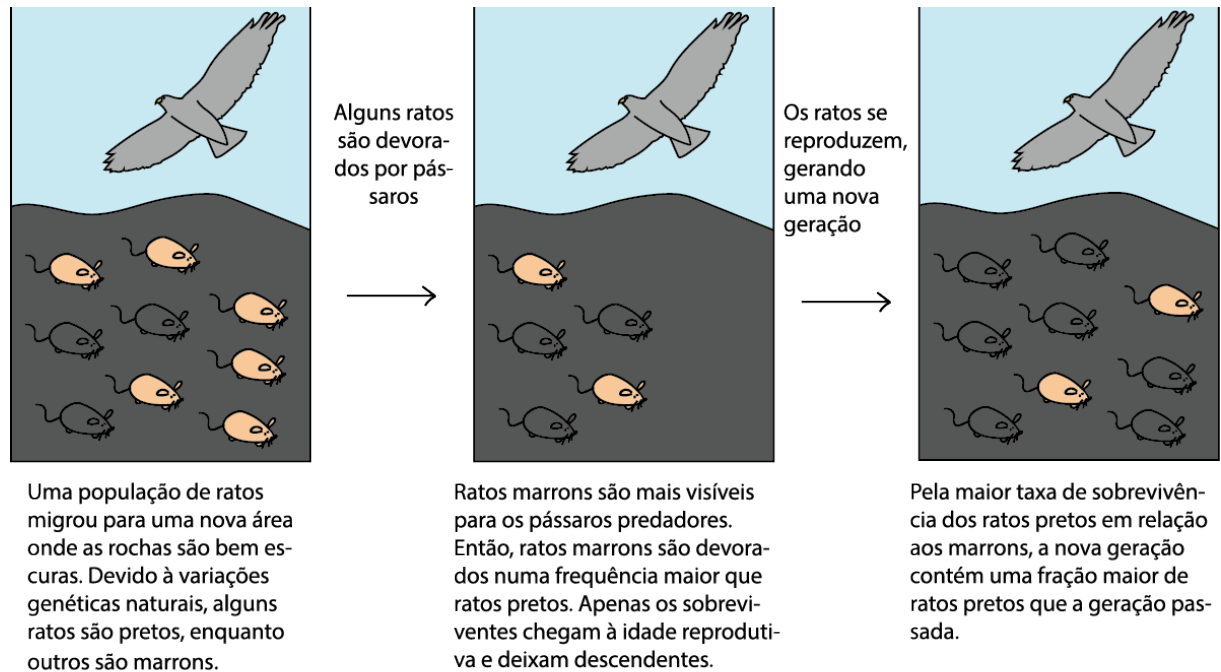
No artigo intitulado *Darwin, evolution & natural selection*, da organização Khan Academy, a Figura 6 é utilizada como um exemplo hipotético e bem simplificado do processo de seleção natural, em que um grupo de ratos com variação hereditária na coloração de seus pelos, migram para um novo ambiente onde as rochas possuem coloração escura. Nesse ambiente, os ratos são presas de falcões que enxergam ratos marrons com mais facilidade que ratos pretos, por conta da cor das rochas. Por esse motivo, ratos marrons são devorados com maior frequência e a população de sobreviventes que chegam à idade reprodutiva é majoritariamente formada por ratos pretos. Como a coloração do pelo é uma característica hereditária – sendo passada dos pais para os filhos – as gerações subsequentes tendem a apresentar uma fração de ratos pretos cada vez maior. Essa mudança nas características hereditárias de uma população é um exemplo de evolução biológica (KHAN ACADEMY, 2016).

A evolução biológica é dependente de diversidade e mutação. No exemplo dos ratos, devido a alguma variedade genética, alguns deles são pretos e outros marrons. Se todos os ratos fossem marrons, por exemplo, não haveria formas da população se adaptar ao ambiente. Essa variação genética – necessária ao processo de evolução – é resultado de mutações aleatórias que alteram a sequência de DNA dos indivíduos. Essas variedades genéticas são passadas para novas gerações, e através da reprodução se misturam e se combinam gerando ainda mais variações (KHAN ACADEMY, 2016).

O processo evolucionário pode ser caracterizado por quatro elementos chave: (I) uma população de entidades; (II) mecanismo de seleção; (III) registro de aptidão ou capacidade e (IV) geração de diversidade. Na evolução biológica, as espécies são escolhidas

(positivamente ou negativamente) de acordo com a sua capacidade de sobrevivência e reprodução no ambiente. Sobrevivência diferenciada e diversidade são a chave para a evolução (BRABAZON; O'NEILL; MCGARRAGHY, 2015).

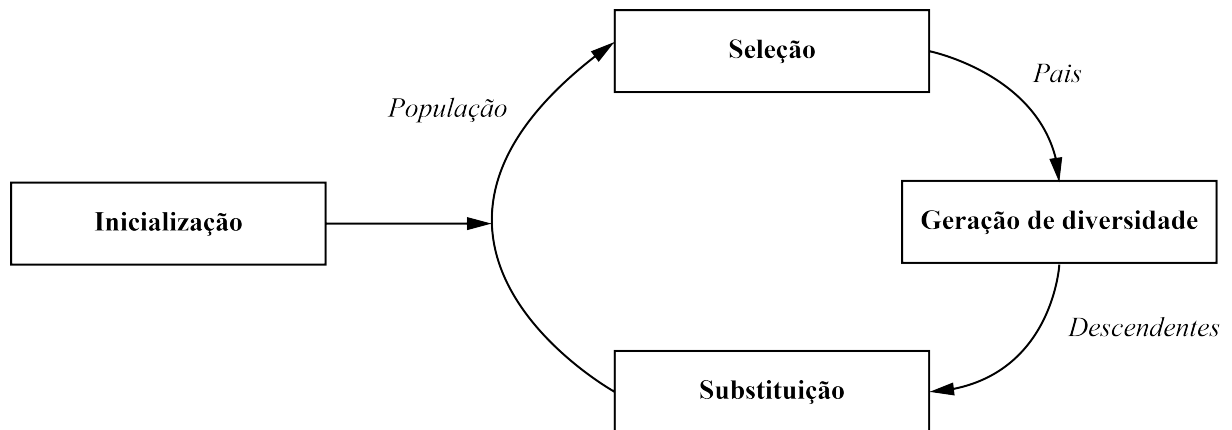
Figura 6 – Como funciona a seleção natural.



Fonte: Traduzido de KHAN ACADEMY (2016)

A popularização da teoria da Seleção Natural de Charles Darwin e a posterior descoberta da existência do DNA, e sua importância na determinação de traços hereditários, no século XX, possibilitaram o estudo da decifração do código genético, inspirando a criação de uma família de algoritmos computacionais conhecidos em conjunto como computação evolucionária (EC – do inglês *Evolutionary Computation*). Esses algoritmos, além do fato de serem inspirados na evolução biológica, se caracterizam pela utilização de alguns operadores genéticos como mutação e/ou cruzamento (recombinação gênica). A Figura 7 apresenta o ciclo do processo evolucionário artificial comum aos EAs (BRABAZON; O'NEILL; MCGARRAGHY, 2015).

O Algoritmo 1 descreve um EA. Existem diferentes EAs, e em cada um deles cada etapa pode ser implementada de diferentes formas, as quais são aqui descritas de forma genérica. A etapa de seleção tende a escolher os melhores indivíduos da população atual. A geração de novos indivíduos cria filhos semelhantes aos pais, mas não idênticos. Cada indivíduo representa uma solução experimental no ambiente, e os melhores indivíduos têm uma maior tendência em reproduzir e criar as gerações futuras. Este processo pode ser considerado como uma busca por soluções ótimas, por meio da criação de indivíduos melhores a cada geração (BRABAZON; O'NEILL; MCGARRAGHY, 2015).

Figura 7 – Ciclo evolucionário

Fonte: Adaptado de Brabazon, O'Neill e McGarraghy (2015)

Algoritmo 1: Algoritmo Evolucionário

Inicialização da população com soluções candidatas;

início

repita

 Seleção de indivíduos (pais) para a reprodução da população atual;
 Geração de novos indivíduos (descendentes) a partir dos pais;
 Substituição de parte ou de toda a população atual pelos novos
 indivíduos gerados;

até critério de parada;

fim

Fonte: Adaptado de Brabazon, O'Neill e McGarraghy (2015)

Inicialmente, diferentes EAs foram desenvolvidos de forma independente. O trabalho de Friedberg (1958) é tido como a origem do que hoje é conhecido como Programação Genética (GP – do inglês *Genetic Programming*). Na Europa, Schwefel (1965) e Rechenberg (1973) desenvolveram Estratégias de Evolução (ES – do inglês *Evolution Strategies*). Nos Estados Unidos, Fogel, Owens e Walsh (1966) desenvolveram a Programação Evolucionária (EP – do inglês *Evolutionary Programming*) e Holland (1975) desenvolveu o GA. Mais recentemente, Storn e Price (1995) desenvolveram a Evolução Diferencial (DE – do inglês *Differential Evolution*). Embora os referidos métodos evolucionários tenham surgido de diferentes linhas de pesquisa inicialmente distintas, atualmente estas abordagens muitas vezes se confundem, com propriedades, representações e estratégias sendo usadas alternadamente entre os diferentes algoritmos.

2.5 ALGORITMOS GENÉTICOS

Os GAs foram inventados por John Holland nos anos 60 e foram desenvolvidos por ele, seus estudantes e colegas da Universidade de Michigan, nas décadas de 60 e 70. Inicialmente, o desenvolvimento dos GAs não visava a solução de problemas de otimização, o principal objetivo era a implementação dos mecanismos de adaptação natural em computadores conforme ocorrem na natureza (MITCHELL, 1996).

É somente em 1975 que os GAs são apresentados formalmente como métodos de otimização, no livro *Adaption in Natural and Artificial Systems*, de Holland (1975). A popularização dos GAs como métodos de otimização deve-se muito à obra intitulada *Genetic algorithms in Search, Optimization, and Machine Learning*, escrita por um aluno de Holland, Goldberg (1989), que foi capaz de resolver um difícil problema de otimização de transmissão por gasoduto (HAUPT; HAUPT, 1998).

O primeiro GA apresentado como método de otimização, desenvolvido por Holland (1975), é conhecido como GA simples (SGA – do inglês *Simple Genetic Algorithm*) ou GA canônico. Este GA possui representação binária, seleção proporcional à aptidão, baixa probabilidade de mutação, com ênfase na recombinação para a geração de novas soluções (EIBEN, 2003).

Os GAs são métodos heurísticos de busca aplicáveis a uma enorme gama de problemas de otimização. A base de um GA é a evolução. Este embasamento se justifica pela variedade de espécies adaptadas a seus respectivos ambientes, que se desenvolveram em estruturas complexas, que as tornaram capazes de sobreviver. A reprodução e a adaptação dos descendentes corresponde ao princípio fundamental do sucesso da evolução. Estas são razões pertinentes para adaptar e implementar os princípios evolutivos em problemas de otimização. Resumidamente, GAs são métodos de otimização biologicamente inspirados, que imitam a seleção natural ao traduzirem o conceito biológico da evolução em algoritmos (KRAMER, 2017).

Como método bioinspirado, a implementação de um GA utiliza um vocabulário emprestado da genética, do qual alguns termos são apresentados a seguir (BORNE; POPESCU; PHILIP, 2014):

- *Cromossomo* – uma cadeia de ácido desoxirribonucleico (DNA – do inglês Deoxyribonucleic Acid) encontrada em células orgânicas, que codificam a identidade biológica característica de um determinado organismo vivo.
- *Gene* – um bloco funcional de um cromossomo que codifica uma proteína específica. Cada gene define uma determinada característica de um organismo vivo, como altura, cor dos olhos, etc. A posição de um gene no cromossomo é muito importante, pois alguma mudança nesse sentido pode produzir alterações significativas nas características dos indivíduos.

- *População* – um conjunto de cromossomos/indivíduos ou organismos que vivem simultaneamente em um determinado ambiente.
- *Geração* – uma estrutura de população em um determinado instante de tempo. Pode considerar-se que populações evoluem através das gerações por meio da seleção natural. Alguns indivíduos passam a existir em uma determinada geração, enquanto outros deixam de existir a partir de uma determinada geração.
- *Descendência (filhos)* – cromossomos ou organismos gerados pelo processo de cruzamento e/ou mutação entre pais selecionados. Como os pais passam certas características para os filhos, os mesmos também são conhecidos como herdeiros.

Um SGA tem a mesma estrutura que qualquer programa de evolução. Durante uma geração t , um GA mantém uma população de soluções candidatas (cromossomos, vetores), $P(t) = \{x_1^t, \dots, x_n^t\}$. Cada solução x_i^t é avaliada, calculando-se uma medida de sua aptidão (do inglês *fitness*). Então, uma nova população (iteração $t + 1$) é formada pela seleção dos indivíduos mais aptos. Alguns membros dessa nova população passam por alterações por meio de cruzamento e mutação, para gerar novas soluções. A operação de cruzamento combina as características de dois cromossomos pais para gerar dois cromossomos filhos similares, pela troca de segmentos correspondentes dos pais. Por exemplo, se os pais são representados pelos vetores de cinco elementos $(a_1, b_1, c_1, d_1, e_1)$ e $(a_2, b_2, c_2, d_2, e_2)$, o cruzamento dos cromossomos após o segundo gene geraria os filhos $(a_1, b_1, c_2, d_2, e_2)$ e $(a_2, b_2, c_1, d_1, e_1)$. A operação de cruzamento permite a troca de informações entre diferentes soluções candidatas (MICHALEWICZ, 1996).

A operação de mutação altera arbitrariamente um ou mais genes de um determinado cromossomo por uma mudança aleatória, com probabilidade de acordo com uma taxa de mutação. A mutação permite a introdução de variabilidade extra na população. Um GA utilizado para a otimização de um determinado problema deve conter os seguintes cinco componentes (MICHALEWICZ, 1996):

- Uma representação genética para as soluções candidatas do problema;
- Uma forma de criar uma população inicial de soluções candidatas;
- Uma função de avaliação, que desempenha o papel do ambiente, classificando as soluções de acordo com a aptidão;
- Operadores genéticos que alteram a composição dos filhos;
- Valores para vários parâmetros que GAs utilizam (tamanho da população, probabilidade de aplicação dos operadores genéticos, etc.).

No problema de otimização com um GA, uma solução é um potencial candidato a ótimo para o problema em estudo, cuja representação tem um importante papel, já que a mesma determina a escolha dos operadores genéticos. As representações geralmente são listas de valores, frequentemente baseadas em conjuntos de símbolos. As listas com valores contínuos são chamadas vetores; as listas contendo bits, são chamadas de cadeia de

bits (do inglês *bit strings*). Escolhida a representação, os operadores genéticos produzem novas soluções, que permitem a busca no espaço de solução. A codificação das soluções em uma determinada representação é chamada de genótipo ou cromossomo. Dependendo da representação, um mapeamento do cromossomo (genótipo) para a solução do problema (fenótipo) é necessária. A representação contínua não necessita do mapeamento genótipo-fenótipo, pois o genótipo é a própria solução (KRAMER, 2017).

Um GA típico é caracterizado por seguir as etapas mostradas no Algoritmo 2. Cada iteração do processo é chamada de geração, possuindo em torno de 50 a 500, ou até mais gerações. O conjunto completo de gerações é chamado de execução do algoritmo. Ao final de uma execução, normalmente há um ou mais cromossomos altamente aptos na população. Como a aleatoriedade tem grande importância na execução do GA, duas execuções, com diferentes sementes para a geração de números pseudoaleatórios, geralmente apresentam diferentes comportamentos. Embora o procedimento descrito no Algoritmo 2 descreva a base para a maioria das aplicações dos GAs, existem muitos detalhes a serem considerados, como a forma com que as soluções candidatas são codificadas, o tamanho da população, os detalhes e as probabilidades dos operadores de seleção, cruzamento e mutação, e o número máximo de gerações permitidos (MITCHELL; TAYLOR, 1999).

Algoritmo 2: Algoritmo Genético

Comece com uma população gerada aleatoriamente de n cromossomos;

início

repita

 Calcule a aptidão $f(x)$ de cada cromossomo x na população;

repita

 Selecione um par de cromossomos pais da seleção atual;

 Com probabilidade p_c (probabilidade de cruzamento), cruze o par tomando parte do cromossomo de um dos pais e a outra parte do outro pai, formando uma única descendência;

 Com probabilidade p_m (probabilidade de mutação), transforme cada gene e coloque o cromossomo resultante na nova população.

 Mutação tipicamente substitui o valor atual de um gene por outro valor (por exemplo, 0 por 1 na codificação binária);

até n descendentes tenham sido criados;

 Substitua a população atual pela nova população;

até Critério de parada;

fim

Fonte: Mitchell e Taylor (1999)

2.5.1 Seleção Parental

Na implementação de um GA, são necessários duas formas de seleção: a seleção de indivíduos que passarão pelos operadores de cruzamento e mutação, e a seleção dos indivíduos que sobreviverão e passarão para a próxima geração. A maneira com que os

indivíduos são escolhidos para o processo de cruzamento influencia diretamente no processo de evolução artificial das populações. Dois casos extremos são apresentados (BORNE; POPESCU; PHILIP, 2014):

1. Se a seleção for bastante rigorosa, poucos indivíduos serão aptos ao processo de cruzamento, e então, há o risco das populações se tornarem dominadas por indivíduos sub-ótimos, tendendo à redução da diversidade da população.
2. Se a seleção é muito tolerante, muitos indivíduos com baixa aptidão são envolvidos no processo de cruzamento, e então, as populações podem se tornar dispersas e a evolução em direção à uma solução ótima pode oscilar e ocorrer de forma muito lenta.

Um bom *trade-off* entre a diversidade da população (*exploration ability*) e a velocidade de evolução (*exploitation ability*) é recomendado. A escolha pelo tipo de seleção mais apropriado é um tópico aberto no estudo dos GAs. No entanto, algumas técnicas de seleção se mostraram eficientes em diversas aplicações (BORNE; POPESCU; PHILIP, 2014).

Embora as palavras *exploitation* e *exploration* sejam ambas traduzidas como exploração no português, elas possuem significados distintos na língua inglesa, principalmente no contexto de otimização. A discussão sobre a troca e o balanço entre *exploitation* e *exploration* é recorrente no estudo dos GAs. Segundo Brabazon, O'Neill e McGarraghy (2015), controlar o balanço entre *exploitation* e *exploration* é uma questão chave para a construção de um bom GA. Eiben e Schippers (1998) discorrem sobre *exploitation* e *exploration* nos EAs em *On Evolutionary Exploration and Exploitation*, e afirmam que *exploitation* está mais ligada às formas de seleção, enquanto *exploration* está mais ligada aos operadores de cruzamento e mutação.

Em relação às formas de seleção parental, Brabazon, O'Neill e McGarraghy (2015) dedicam uma subseção em *Natural Computing Algorithms*, onde apresentam a Seleção Proporcional à Aptidão (FPS – do inglês *Fitness Proportionate Selection*) e algumas seleções baseadas em classificação, incluindo as seleções de truncamento e de torneio.

2.5.2 Algoritmo Genético de Codificação Real

Os primeiros GAs utilizavam strings binárias (cadeia contendo bits, isto é, números '0' ou '1') para representar números inteiros e reais, no entanto, a depender do problema, essa representação não é a ideal. A precisão de um número real nessa representação fica limitada ao comprimento da string (número de bits). Ao determinar-se o comprimento das strings, as mesmas podem ser muito curtas, resultando em precisão insuficiente, ou serem extremamente longas (WIRSANSKY, 2020).

Um RCGA se difere de um GA comum pela forma com que os genes dos cromos-

somos são representados. No RCGA, os genes assumem valores reais ao invés de valores binários. Segundo Herrera, Lozano e Verdegay (1998), as vantagens dos GAs não advém da representação binária, e por essa razão, o uso de representação não-binária pode ser mais adequado, a depender do problema e aplicação. Os autores ainda afirmam que a representação por número real é uma das mais importantes, sendo razoável a sua utilização em problemas de otimização com variáveis de decisão pertencentes a um espaço contínuo de busca.

Embora os RCGAs tenham sido utilizados em problemas de otimização numérica com domínios contínuos, como os casos de Wright (1991), Michalewicz (1996), Eshelman e Schaffer (1993), entre outros; até 1991 não haviam estudos teóricos sobre o seu funcionamento, e o uso desse tipo de algoritmo era controverso. Pesquisadores que eram habituados com a teoria fundamental do GAs de representação binária não reconheciam as vantagens do RCGA, visto que a teoria indicava que a codificação binária era mais eficiente. Posteriormente, Wright (1991), Goldberg (1991), Radcliffe (2003), Eshelman e Schaffer (1993), apresentaram ferramentas para o tratamento e análise teórica dos RCGAs (HERRERA; LOZANO; VERDEGAY, 1998).

Michalewicz (1996) testou as representações binária e real na resolução de um problema de controle dinâmico e concluiu que a representação por ponto flutuante é mais rápida, consistente e garante uma precisão maior. Segundo o autor, a representação por ponto flutuante é mais facilmente implementada, e permite elaborar operadores com base em conhecimentos específicos do problema.

Nos RCGAs, os genes dos cromossomo dos indivíduos são representados por números reais, o que é justificável na resolução de problemas de otimização com parâmetros em domínios contínuos. Neste método, um cromossomo é um vetor com números no formato de ponto flutuante, cuja precisão é restrita ao computador no qual o algoritmo é executado. O tamanho de um cromossomo é o mesmo do vetor que soluciona o problema, e neste sentido, cada gene representa uma variável de decisão (HERRERA; LOZANO; VERDEGAY, 1998).

Segundo Sivanandam e Deepa (2007), embora o RCGA seja recomendado para alguns problemas especiais, sua utilização requer o desenvolvimento de operadores de cruzamento e mutação diferentes dos utilizados em GAs de representação binária. Herrera, Lozano e Verdegay (1998) afirmam que esses diferentes operadores genéticos precisam obedecer o seguinte requisito: os valores dos genes devem permanecer dentro do intervalo estabelecido pelas variáveis de decisão a que representam.

2.5.3 Operadores de Cruzamento e Mutação

A operação de cruzamento (do inglês *crossover*) é um operador genético que permite a combinação do material genético de duas ou mais soluções, implementando um mecanismo que mistura o material genético dos pais. Na representação binária, um

operador de cruzamento bastante conhecido é o *n-point crossover*, que divide duas soluções em n posições e alternadamente as monta em novas. Por exemplo, se 0010110010 é o primeiro pai e 1111010111 é o segundo, *one-point crossover* (cruzamento em um ponto), aleatoriamente escolheria uma posição, e supondo que tenha sido a posição 4, geraria duas soluções candidatas **0010010111** e **1111110010** (KRAMER, 2017).

A mutação pode ser resumida como a alteração do valor de um ou mais genes. Na natureza a mutação não é tão frequente, e geralmente gera indivíduos com DNA inviáveis. No entanto, em problemas de otimização, alterações aleatórias geradas pela operação de mutação podem auxiliar na exploração do domínio do problema (SIVANANDAM; DEEPA, 2007).

Nos GAs, o operador de mutação tem a finalidade de preservar a diversidade da população. Os indivíduos mutantes são geralmente piores que os indivíduos iniciais e a taxa de mutação (p_m) é pequena se comparada à taxa de cruzamento (p_c). No entanto, os indivíduos mutantes, quando envolvidos em operação de cruzamento com indivíduos mais aptos, podem produzir filhos afastados de uma determinada zona de busca local, e neste sentido, a condução para um ótimo local pode ser evitada e superada. Em pequena proporção, a preservação da diversidade da população evita a estagnação do algoritmo em um ótimo local, mas por outro lado, o excesso de diversidade impede a busca por uma solução ótima e estimula a oscilação na busca. É interessante que as populações se mantenham focadas e, ao mesmo tempo, preservem diversidade suficiente (BORNE; POPESCU; PHILIP, 2014).

Em GAs com representação binária, o método *bit-flip*, que pode ser traduzido como método de inversão de bit, é comumente utilizado. Este método consiste na inversão do valor de um gene representado por um bit, em que um bit de valor ‘0’ se torna ‘1’, e um de valor ‘1’ se torna ‘0’ (KRAMER, 2017).

Os operadores mencionados anteriormente operam em cromossomos de representação binária. Para os RCGAs, em que os cromossomos são codificados como números reais, assumindo os próprios valores dos parâmetros de otimização, vários operadores genéticos foram desenvolvidos, e esses são amplamente estudados. Sorsa, Peltokangas e Leiviska (2008) utilizaram um RCGA na identificação de parâmetro do modelo quimiostático macroscópico e apresentaram alguns operadores genéticos de cruzamento e mutação, incluindo: cruzamento aritmético (*arithmetic crossover*), cruzamento heurístico (*heuristic crossover*), cruzamento centrado nos pais (*parent-centric crossover*), cruzamento linear (*linear crossover*), mutação uniforme (*uniform mutation*) e mutação não-uniforme de Michalewicz (*Michalewicz’s non-uniform mutation*).

Herrera, Lozano e Verdegay (1998) citam que as similaridades entre RCGA e ES permitem a troca de alguns operadores entre os métodos. Os autores também apresentam muitos outros operadores além dos apresentados por Sorsa, Peltokangas e Leiviska (2008),

incluindo: cruzamento plano (*flat crossover*), cruzamento simples (*simple crossover*), cruzamento BLX- α (BLX- α *crossover*), cruzamento discreto (*discrete crossover*), cruzamento de linha estendida (*extended line crossover*), cruzamento intermediário estendido (*extended intermediate crossover*), cruzamento BGA linear (*linear BGA crossover*), cruzamento baseado em conectivos difusos (*fuzzy connectives based crossover*), mutação aleatória (*random mutation*), *real number creep*, mutação modal discreta (*discrete modal mutation*) e mutação modal contínua (*continuous modal mutation*).

Os operadores citados anteriormente ilustram a dimensão da área de estudos voltada para os RCGAs. Como pode ser observado pela quantidade de operadores genéticos, existem várias formas de se implementar um RCGA. O algoritmo do presente trabalho utiliza o operador de cruzamento binário simulado (*simulated binary crossover*), apresentado por Deb e Agrawal (1995), e o operador de mutação polinomial (*polynomial mutation*), apresentado por Deb e Deb (2012), os quais serão detalhados nos Aspectos Metodológicos.

2.5.4 Estratégias de Substituição

Brabazon, O'Neill e McGarraghy (2015) chamam de estratégia de substituição a escolha dos pais e filhos que sobreviverão e passarão para a próxima geração. Já Borne, Popescu e Philip (2014) referem-se a essa escolha por seleção de sobrevivência.

Após a reprodução (cruzamento e/ou mutação), um grupo de indivíduos incluindo pais e filhos é obtido (sendo alguns filhos mutantes). Esse grupo é muito grande para ser totalmente incluído na população. A operação de cruzamento inclui dois pais e dois filhos, e só dois indivíduos devem ser escolhidos para serem incluídos na nova população. Na mutação, um grupo contendo o indivíduo inicial e o indivíduo mutante é formado, e somente um deles deve ser incluído. Então é necessário especificar como os indivíduos sobreviventes são selecionados (BORNE; POPESCU; PHILIP, 2014).

No SGA, uma estratégia de substituição geracional é geralmente utilizada. O número de filhos produzidos em cada geração é igual ao número de indivíduos da população atual e, durante a etapa de substituição, toda a população atual é substituída pelo conjunto dos filhos (BRABAZON; O'NEILL; MCGARRAGHY, 2015).

A quantidade de filhos gerados não precisa ser necessariamente igual ao tamanho da população atual. A razão entre o número de filhos gerados e o tamanho da população atual é chamada de *generation gap*. Embora geralmente se considere *generation gap*=1, também é possível gerar mais filhos do que membros da população atual (*generation gap*>1), e então escolher os n melhores indivíduos (com n sendo o tamanho da população) entre os filhos, para permanecerem na próxima geração (BRABAZON; O'NEILL; MCGARRAGHY, 2015).

Borne, Popescu e Philip (2014) citam um método de seleção elitista, em que uma elite formada por ótimos atuais é mantida na população, enquanto os outros indivíduos são substituídos por seus herdeiros. Os autores apontam que, embora esta técnica forneça uma boa relação entre *exploitation* e *exploration*, a diversidade das gerações não decresce na velocidade necessária, e a busca pelo ótimo local pode ficar lenta e prejudicada.

Para Borne, Popescu e Philip (2014), a seleção chamada $(\mu + \lambda)$ (lê-se mu plus lambda) é provavelmente a melhor. Segundo os autores, esta técnica de seleção combina as vantagens da seleção elitista e da substituição geracional, preservando os melhores indivíduos entre os grupos dos reprodutores e dos herdeiros gerados. Borne, Popescu e Philip (2014) alertam para o fato deste método de substituição tender para *exploitation*, já que os indivíduos mutantes são frequentemente descartados. Kramer (2017) comenta que o método $(\mu + \lambda)$ escolhe os μ melhores indivíduos a partir dos λ filhos e μ indivíduos que os geraram, justificando o nome desta estratégia de substituição. O autor também cita o método (μ, λ) (lê-se mu comma lambda), em que os pais são descartados e os melhores indivíduos, somente entre o conjunto de filhos gerados, são selecionados para a próxima geração. Embora na seleção (μ, λ) bons pais possam ser descartados, segundo Kramer (2017), essa perda pode ser uma estratégia razoável para superar ótimos locais.

3 ASPECTOS METODOLÓGICOS

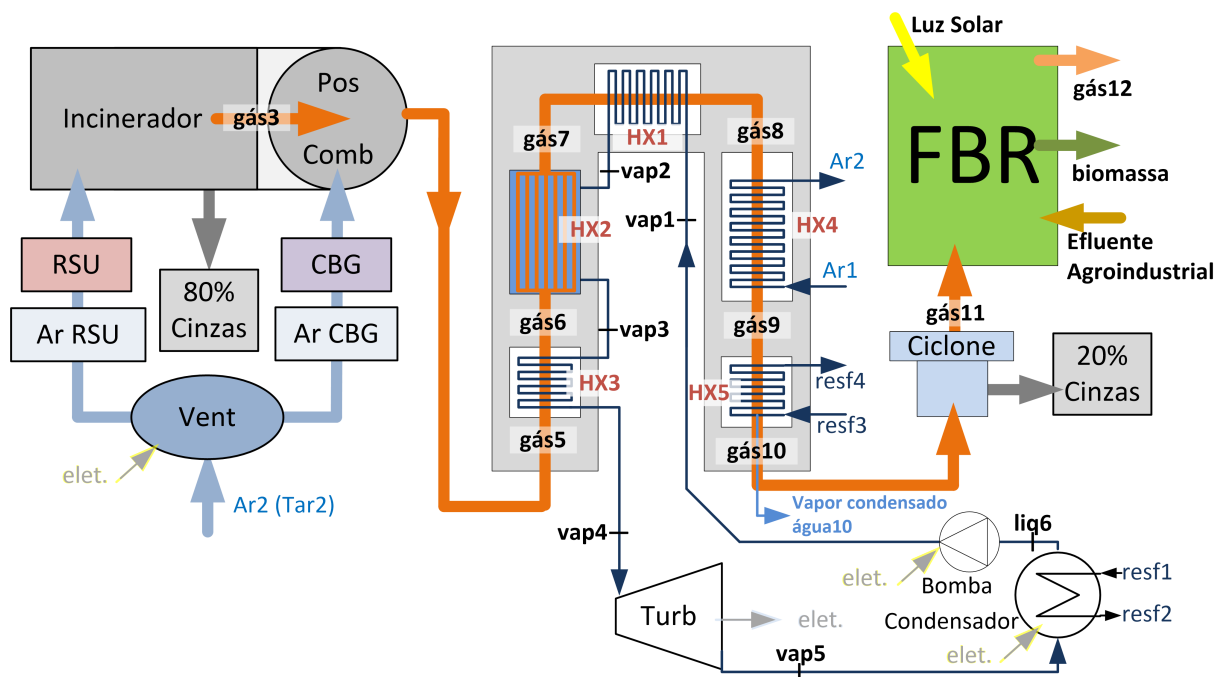
Neste capítulo será apresentada a metodologia utilizada para a criação e implementação de um RCGA e sua validação. Posteriormente, é apresentado um detalhamento da utilização do RCGA na otimização termoeconômica de um sistema térmico de geração de energia por incineração de RSU.

Para o melhor entendimento do processo de otimização termoeconômica, algumas considerações sobre o objeto de estudo (sistema térmico a ser otimizado) e o método utilizado para a otimização (RCGA) precisam ser feitas.

3.1 CONSIDERAÇÕES SOBRE O OBJETO DE ESTUDO

Como mencionado na Fundamentação Teórica de sua tese intitulada *Análise e Otimização Termoeconômica de Sistema de Geração de Energia por Incineração de Resíduos com Filtro Biológico de Emissões*, Galante (2019) analisa o módulo 5 integrado ao módulo 2 do sistema integrado do NPDEAS, e propõe uma abordagem clássica de otimização exergoeconômica, seguindo metodologia apresentada por Bejan, Tsatsaronis e Moran (1995). O módulo 5 integrado ao módulo 2 é ilustrado na Figura 8, que mostra a unidade de incineração e ventilação, a caldeira, trocadores de calor, o ciclo Rankine e a unidade dos FBR.

Figura 8 – Incinerador, ciclo Rankine e fotobiorreatores interligados



Fonte: Galante (2019)

A análise exergoeconômica é executada pela verificação do fluxo de custos ao longo das linhas do sistema, ponderadas pelo fluxo exergético. Em cada equipamento, um balanço de custo similar ao balanço de energia e exergia é realizado por

$$\dot{C}_{P,tot} = \dot{C}_{F,tot} + \dot{Z}_{tot}^{CI} + \dot{Z}_{tot}^{OM}, \quad (7)$$

em que \dot{C} (\$/h) corresponde à taxa de custos associados à exergia, enquanto \dot{Z} representa a taxa de custos não-exergéticos, associados ao capital de investimento (subscrito CI) e operação/manutenção (subscrito OM). (GALANTE, 2019).

A taxa de custo de exergia (\dot{C}) é calculada pelo produto entre custo médio por unidade de exergia, c (\$/kJ), e taxa de exergia, \dot{E} (kJ/h), ou pelo produto entre custo médio por unidade de exergia, vazão mássica, \dot{m} (kg/h), e exergia específica, e (kJ/kg), conforme segue:

$$\dot{C}_e = c_e \dot{E}_e = c_e (\dot{m}_e e_e) \quad (8)$$

$$\dot{C}_i = c_i \dot{E}_i = c_i (\dot{m}_i e_i) \quad (9)$$

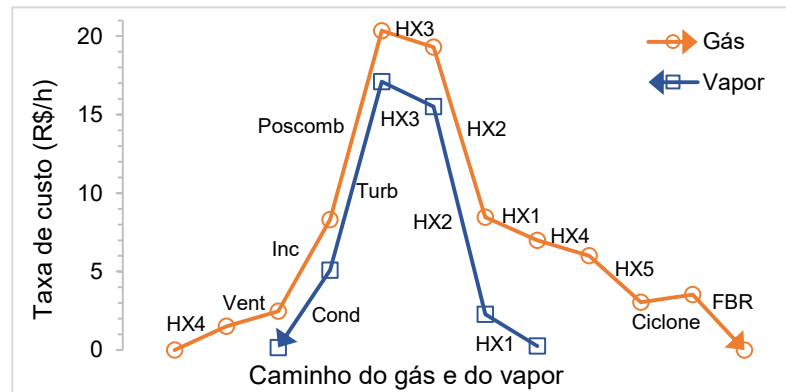
Nas Equações (8) e (9), há transporte de exergia na entrada (subscrito i) ou na saída (subscrito e) de um sistema. Nas condições em que o transporte de exergia acontece por calor ou trabalho, os fluxos de custos são calculados, respectivamente, como:

$$\dot{C}_Q = c_Q \dot{Q} \quad (10)$$

$$\dot{C}_W = c_W \dot{W} \quad (11)$$

A Figura 9 apresenta as taxas de custo para o gás de combustão e para o vapor, em que a taxa de custo associada ao gás de combustão aumenta até a entrada do trocador de calor HX3, e a partir desse ponto, o gás de combustão passa atuar como combustível dos trocadores de calor, agregando valor ao vapor do ciclo Rankine (GALANTE, 2019).

Figura 9 – Variação da taxa de custo associado ao gás de combustão e ao vapor através dos equipamentos



Fonte: Galante (2019)

Geralmente, os custos exergéticos são determinados para cada componente k do sistema, individualmente. Considera-se que a taxa de custos exergéticos que saem de um componente deve ser igual à taxa de custos que entram nesse equipamento somada às alterações referentes ao CI e OM, conforme segue:

$$\sum_e \dot{C}_{e,k} + \dot{C}_{W,k} = \dot{C}_{q,k} + \sum_i \dot{C}_{i,k} + \dot{Z}_k^{OM} + \dot{Z}^{CI} \quad (12)$$

Galante (2019) afirma que a análise exergoeconômica exige a estimativa de custos de equipamentos. Essas estimativas são representadas por funções que levam em conta as características operacionais e construtivas, como os custos do incinerador, da câmara de pós-combustão, e da turbina, por exemplo, dados respectivamente por

$$CI_{inc} = B_k \cdot \dot{m}_{RSU}^{0,67}, \quad (13)$$

$$CI_{poscomb} = B_k \cdot \dot{m}_{gás} \cdot (1 + \exp(0,018 \cdot T_{gás} - 26,4)), \quad (14)$$

$$CI_{turb} = B_k \cdot W_{turb}^{0,7} \cdot \left[1 + \left(\frac{0,05}{1 - \eta_{turb}} \right)^3 \right] \cdot \left[1 + 5 \cdot \exp \left(\frac{T_{vap} + 273,15 - 866}{10,42} \right) \right], \quad (15)$$

em que B refere-se a uma constante de ajuste do valor de cada um dos k equipamentos em suas condições de projeto.

A análise e otimização de Galante (2019) utiliza o *payback*, ou seja, o tempo de recuperação de capital investido na planta do NPDEAS como função objetivo. A princípio, Galante (2019) implementa um caso padrão da planta utilizando o método $\varepsilon - NUT$, descrito em Incropera (2008), para o dimensionamento dos trocadores de calor, e a metodologia de Çengel e Boles (2013) para a modelagem do ciclo Rankine, cujos estados termodinâmicos são calculados com auxílio do software *Engineering Equation Solver* (EES). Para este mesmo caso padrão, Galante (2019) realiza a análise exergoeconômica considerando dois cenários: RSU sem custo e RSU com custo associado à uma fração do custo de incineração. Nestes cenários, o caso padrão resulta em *payback* de 3,89 e 4,18 anos, respectivamente, calculados por

$$Payback = CI_{total} / (hr_{ano} \cdot (Caixa_{in} - Caixa_{out})), \quad (16)$$

em que hr_{ano} é a quantidade de horas de operação por ano e $Caixa_{in}$ e $Caixa_{out}$ são dados por:

$$Caixa_{in} = \dot{C}_{RSU} + \dot{C}_{bio} + \dot{C}_{W,turb} + \dot{C}_{resf2} + \dot{C}_{resf4} + \dot{C}_{cinza} \quad (17)$$

$$Caixa_{out} = \dot{C}_{CBG} + \dot{C}_{elet,pump} + \dot{C}_{elet,vent} + \dot{Z}_{OMtotal} \quad (18)$$

Após o cálculo de *payback* para o caso padrão, Galante (2019) calcula algumas variáveis que permitem a comparação e otimização dos componentes do sistema: custo de destruição de exergia ($\dot{C}_{D,k}$), diferença de custo relativo (r_k), fator exergoeconômico (f_k), importância em custos definida pela soma $\dot{Z}_k + \dot{C}_{D,k}$ identificada por $\dot{Z}_k \dot{C}_{D,k}$, eficiência exergética (ε_E) e por fim, razão de destruição de exergia (y_D).

A variável $\dot{C}_{D,k}$ e a variável r_k , que mostra o crescimento relativo do custo por unidade de exergia entre combustível e produto de cada componente, são calculadas por:

$$\dot{C}_{D,K} = \dot{E}_{D,K} \cdot c_{F,k} \quad (19)$$

$$r_k = \frac{1 - \varepsilon_{E,k}}{\varepsilon_{E,k}} + \frac{\dot{Z}_k}{C_{E,k} \cdot \dot{E}_{p,k}} \quad (20)$$

Segundo (GALANTE, 2019), a minimização de r_k indica que o preço do produto do equipamento cresceu o mínimo possível após a passagem por este equipamento, sendo então alvo de otimização. O termo f_k quantifica o quanto o custo exergético de um equipamento se deve ao CI e OM e o quanto se deve à exergia perdida ou destruída. Um alto valor de f_k indica que a diminuição dos custos do equipamento é mais vantajosa para o sistema, mesmo diminuindo-se a eficiência. Valores baixos de f_k indicam que melhorias de eficiência devem ser feitas, mesmo que isso exija mais CI. A variável f_k é calculada por:

$$f_k = \frac{\dot{Z}_k}{\dot{Z}_k + c_{F,k} \cdot \dot{E}_{D,K}} \cdot 100\% \quad (21)$$

Com as variáveis definidas, a metodologia de otimização de Galante (2019) é feita através da minimização da diferença de custo relativo (r_k). A ordem de prioridade de equipamentos a serem otimizados segue:

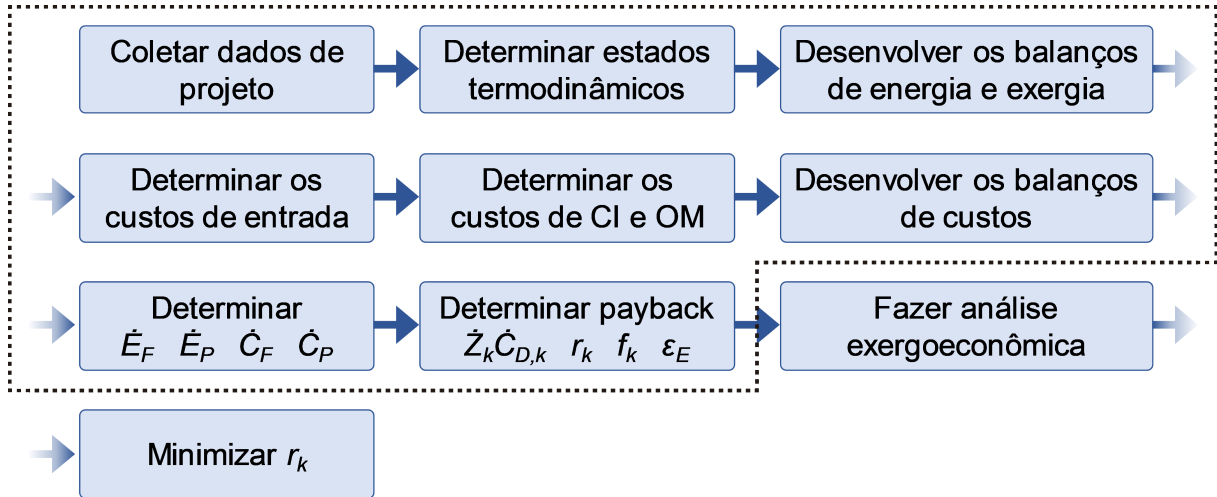
1. Priorizar equipamentos com maiores $\dot{Z}_k \dot{C}_{D,k}$;
2. Priorizar os equipamentos com maior diferença de custo relativo r_k ;
3. Verificar através de f_k se o equipamento precisa de redução de custos ou melhora na eficiência;
4. Verificar equipamentos com baixa ε_E , ou alta $\dot{E}_{D,k}$

O fluxograma da Figura 10 resume os procedimentos anteriormente citados, que devem ser realizados passo a passo para cada equipamento individualmente, com o intuito de executar a análise e otimização termoeconômica do sistema.

Partindo dos casos padrões de 3,89 anos e 4,19 anos de *payback*, para os casos de RSU sem custo e RSU com custo, respectivamente, ao utilizar a abordagem de minimização de r_k , Galante (2019) alcançou os resultados ótimos de 3,09 e 3,19 anos de *payback*, o que representa uma redução de aproximadamente 26% e 31%. No entanto, uma das conclusões de Galante (2019) foi a de que nem sempre a otimização de um equipamento

individualmente resulta na otimização do sistema, e que de sete equipamentos otimizados via minimização de r_k , apenas a otimização de quatro deles resultaram na redução de *payback* do sistema.

Figura 10 – Fluxograma do procedimento de análise e otimização exergoeconômica



Fonte: Galante (2019)

Um outro tipo de abordagem para a otimização do sistema integrado do NPDEAS será executada no presente trabalho. A otimização será realizada utilizando um RCGA. Além de possibilitar a variação de muitos parâmetros de otimização de forma simultânea, o que pode levar a uma redução de *payback*, a otimização via RCGA descarta as etapas de cálculo dos fatores f_k e r_k . A modelagem do sistema seguirá a mesma metodologia de Galante (2019), apenas utilizando outra abordagem em substituição à análise exergoeconômica clássica. Os conceitos necessários para o entendimento do RCGA e seu processo de otimização são apresentados nas próximas seções. Na Figura 10, as etapas que se encontram dentro da linha tracejada são consideradas na metodologia do presente trabalho.

Em relação ao objeto de estudo, é válido destacar que o presente trabalho não propõe uma nova modelagem termodinâmica, exérgica e termoeconômica, tendo como objetivo apresentar uma diferente abordagem para a etapa de otimização apenas. Nesse sentido, a modelagem e implementação do sistema térmico são realizadas seguindo a mesma metodologia apresentada e utilizada por Galante (2019), considerando as mesmas funções para a estimativa de custos dos equipamentos, que levam em conta as características operacionais e construtivas. A planta térmica é implementada na linguagem Python, utilizando as bibliotecas CoolProp e Pyromat para o cálculo das propriedades termodinâmicas. A implementação é realizada considerando o caso padrão de operação, que é utilizado por Galante (2019) para a análise exergoeconômica. A validação da implementação em Python é considerada comparando-se o caso padrão. A implementação é considerada válida caso os resultados forem significativamente próximos aos apresentados por Galante (2019) no caso padrão. Para um melhor detalhamento da modelagem do sistema térmico, a leitura

do estudo de Galante (2019) é recomendada.

3.2 CONSIDERAÇÕES SOBRE O RCGA

3.2.1 População Inicial

O processo de evolução de um GA se inicia pela criação de uma população inicial. Segundo Brabazon, O'Neill e McGarraghy (2015), se bons pontos iniciais forem conhecidos, a eficiência de um GA pode ser melhorada ao utilizar esta informação para alimentar a população inicial. Mas geralmente, segundo os autores, a população inicial é gerada aleatoriamente, em que, na representação binária, um número entre 0 e 1 pode ser gerado para cada gene do cromossomo, com números aleatórios $\geq 0,5$ resultando na colocação do número 1 no gene correspondente. Já na representação de valores reais, Brabazon, O'Neill e McGarraghy (2015) afirmam que se limites para cada gene podem ser determinados, cada elemento do cromossomo pode ser escolhido aleatoriamente dentro do intervalo desses limites.

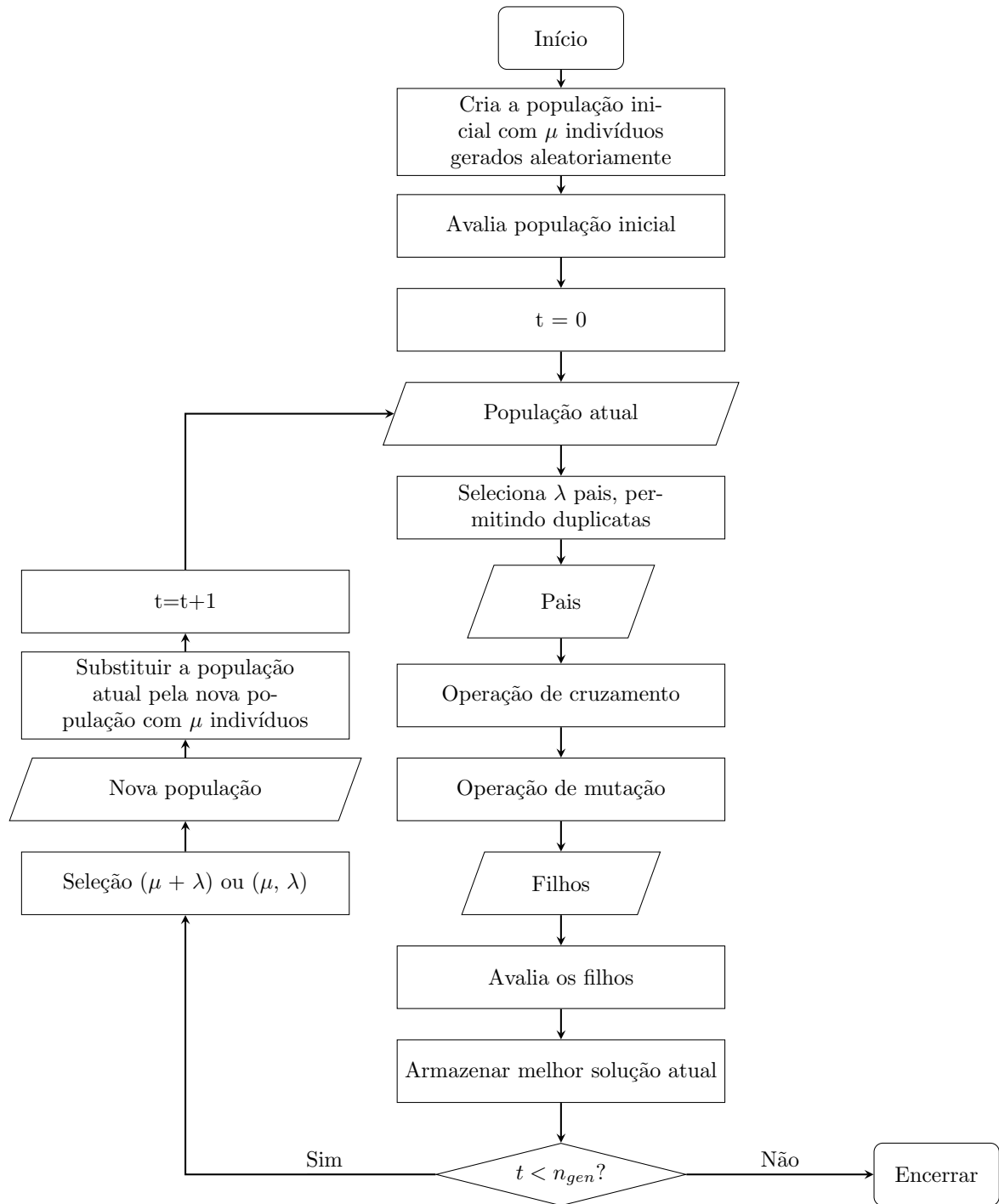
No RCGA do presente trabalho, os limites de cada gene são dados por

$$x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = (1, 2, \dots, N), \quad (22)$$

em que $x_i^{(L)}$ e $x_i^{(U)}$ são ditos limites inferiores e superiores das variáveis de otimização, respectivamente. Vale ressaltar que um gene na posição i deve estar confinado no intervalo limitado inferiormente por $x_i^{(L)}$ e superiormente por $x_i^{(U)}$, isto é, cada gene x_i possui seus limites respectivos. Deste modo, cada indivíduo é criado percorrendo as posições i do cromossomo, e atribuindo um valor aleatório no intervalo $[x_i^{(L)}, x_i^{(U)}]$ para o gene de posição i , com i variando no intervalo $[1, N]$, a depender da quantidade de variáveis de otimização do problema. Basicamente, no RCGA, um cromossomo é representado por um vetor de números de ponto flutuante, como na Equação (23).

$$x = (x_1, x_2, \dots, x_n)^T \quad (23)$$

Inicialmente, um número de indivíduos permitidos na população (μ), geralmente fixo, é definido pelo usuário. Então, na população inicial, μ indivíduos são criados aleatoriamente conforme mencionado anteriormente, respeitando sempre os limites $x_i^{(L)}$ e $x_i^{(U)}$. Um fluxograma que representa o procedimento de otimização com o RCGA é mostrado na Figura 11. Pelo fluxograma, é possível perceber que o procedimento de otimização se inicia pela criação de uma população inicial com μ indivíduos, o que caracteriza a geração de iteração $t = 0$ e, posteriormente, o ciclo de evolução artificial, definido pela seleção parental, cruzamento, mutação e seleção dos sobreviventes, se repete por um número de gerações (n_{gen}), definido pelo usuário. A melhor solução encontrada durante a execução do RCGA é considerada a solução ótima.

Figura 11 – Fluxograma do processo evolutivo do RCGA**Fonte:** Autoria própria

Após a criação da população inicial, os indivíduos são avaliados de acordo com a função objetivo do problema de otimização em questão. A função objetivo deve ser escrita em termos das variáveis de otimização do problema, isto é, os genes dos cromossomos. Resumidamente, uma função objetivo $f(x)$ recebe como entrada um indivíduo x , usando seus genes x_1, \dots, x_n como variáveis. Assumindo que o problema de otimização é escrito

conforme um NLP, dado pela Equação (2), em que todo problema é abordado como a minimização de uma função $f(x)$, a aptidão (F) dos indivíduos é entendida como $F \propto 1/f(x)$, ou seja, quanto menor o valor de $f(x)$, maior a aptidão. Para efeitos práticos, basta considerar que as melhores soluções são aquelas que reduzem o valor de $f(x)$.

3.2.2 Seleção Parental

Após a criação da população inicial, os indivíduos são avaliados conforme o valor de $f(x)$, para seus respectivos genes. Alguns indivíduos devem ser selecionados como pais para o processo de recombinação. Algumas estratégias de seleção parental, como a FPS, se baseiam na classificação dos indivíduos, em que cada indivíduo é classificado de acordo com a sua aptidão, e os pais são selecionados aleatoriamente com uma tendência em favorecer os indivíduos mais aptos. No RCGA do presente trabalho, no entanto, os pais são selecionados de forma aleatória para não aplicar muita pressão de seleção no algoritmo, preservando diversidade, e beneficiando *exploration* em detrimento de *exploitation*. Sivanandam e Deepa (2007) afirmam que, em média, a seleção parental aleatória causa mais perturbação no espaço de busca do que a FPS.

3.2.3 Operador de Cruzamento

No início do GA, um valor (p_c) é definido para a probabilidade de cruzamento e para cada par de pais selecionados, um valor aleatório é gerado a partir da distribuição uniforme $U(0,1)$. Se o valor é menor que p_c , o cruzamento é aplicado para gerar dois filhos, caso contrário, o cruzamento é ignorado e os dois filhos são clones de seus pais. Taxas de cruzamento são geralmente selecionadas no intervalo $p_c \in [0,6,0,9]$, mas se desejado, a taxa de cruzamento pode variar durante a execução do algoritmo (BRABAZON; O'NEILL; MCGARRAGHY, 2015).

No processo de cruzamento do RCGA do presente trabalho, duas soluções candidatas são escolhidas como pais pela seleção parental, e geram dois filhos, isto é, duas novas soluções candidatas pela recombinação de seus genes. O RCGA do presente trabalho utiliza o operador de cruzamento binário simulado (SBX – do inglês *Simulated Binary Crossover*). O SBX foi apresentado primeiramente por Deb e Agrawal (1995). Mais tarde, Deb e Agrawal (1999) estendem o SBX para a solução de problemas com espaço de busca limitado. O procedimento de criação de dois filhos x_1^{t+1} e x_2^{t+1} a partir de dois pais x_1^t e x_2^t pelo operador SBX segue as etapas, conforme Deb e Agrawal (1999):

1. Crie um número aleatório u entre 0 e 1.
2. Encontre um parâmetro $\bar{\beta}$ usando uma distribuição de probabilidade polinomial, do

seguinte modo:

$$\bar{\beta} = \begin{cases} (u\alpha)^{1/(\eta_c+1)}, & \text{se } u \leq \frac{1}{\alpha}, \\ \left(\frac{1}{2-u\alpha}\right)^{1/(\eta_c+1)}, & \text{caso contrário,} \end{cases} \quad (24)$$

em que $\alpha = 2 - \beta^{-(\eta_c+1)}$ e β é calculado como:

$$\beta = 1 + \frac{2}{x_{2i}^t - x_{1i}^t} \min \left[(x_{1i}^t - x_i^{(L)}), (x_i^{(U)} - x_{2i}^t) \right]. \quad (25)$$

O parâmetro η_c é o índice de distribuição do SBX, que pode receber qualquer valor não negativo. Um pequeno valor de η_c permite a criação de soluções distantes dos pais, enquanto um valor grande de η_c produz filhos mais próximos.

3. Então, as soluções filhas são calculadas do seguinte modo:

$$x_{1i}^{t+1} = 0,5 \left[(x_{1i}^t + x_{2i}^t) - \bar{\beta} |x_{2i}^t - x_{1i}^t| \right], \quad (26)$$

$$x_{2i}^{t+1} = 0,5 \left[(x_{1i}^t + x_{2i}^t) + \bar{\beta} |x_{2i}^t - x_{1i}^t| \right]. \quad (27)$$

Assume-se que $x_{1i}^t < x_{2i}^t$, em que uma simples modificação pode ser feita nas equações acima para os casos em que $x_{1i}^t > x_{2i}^t$. Cada variável é escolhida individualmente com uma probabilidade de 50%, e o operador SBX é aplicado variável por variável.

3.2.4 Operador de Mutação

Segundo Borne, Popescu e Philip (2014), o operador de mutação também é aplicado aleatoriamente, com base em uma taxa de probabilidade de mutação (p_m), cujo valor geralmente é escolhido no intervalo $p_m = [0,005,0,1]$.

O operador escolhido para o RCGA do presente trabalho é o operador de mutação polinomial (*Polynomial Mutation*). Este operador é utilizado no NSGA-II (*Nondominated Sorting Genetic Algorithm - II*) de Deb *et al.* (2002). Deb e Deb (2012) afirmam, citando Deb e Agrawal (1999), que o operador de mutação polinomial possui um índice definido pelo usuário (η_m) que induz um efeito de perturbação em uma variável. Segundo os autores, um valor de η_m no intervalo $[20, 100]$ se mostrou adequado para a maioria dos problemas em que o operador foi testado.

Neste operador, uma distribuição de probabilidade polinomial é utilizada para causar uma perturbação de valor $O((b-a)/\eta_m)$, em que a e b são os limites inferior e superior da variável, respectivamente. A distribuição de probabilidade à esquerda e à direita do valor de uma variável é ajustado para que nenhum valor fora do intervalo $[a, b]$ seja criado. Para um determinado indivíduo $x \in [a, b]$, a solução mutante x' é criada, variável por variável, a partir de um número u gerado aleatoriamente no intervalo $[0, 1]$, da seguinte forma (DEB; DEB, 2012):

$$x_i' = \begin{cases} x_i + \bar{\delta}_L (x_i - x_i^{(L)}), & \text{se } u \leq 0,5, \\ x_i + \bar{\delta}_R (x_i^{(U)} - x_i), & \text{caso contrário,} \end{cases} \quad (28)$$

em que $\bar{\delta}_L$ e $\bar{\delta}_R$ são calculados conforme segue:

$$\bar{\delta}_L = (2u)^{1/(1+\eta_m)} - 1, \quad \text{se } u \leq 0,5, \quad (29)$$

$$\bar{\delta}_R = 1 - (2(1-u))^{1/(1+\eta_m)}, \quad \text{caso contrário.} \quad (30)$$

O operador de mutação polinomial também recebe um argumento ind_{pb} , referente a probabilidade individual de cada gene ser modificado. No processo de mutação, um número é gerado no intervalo $[0, 1]$, e se o valor é menor que p_m , o indivíduo passa pelo processo de mutação. Caso contrário, uma cópia deste é adicionado ao conjunto de filhos. No processo de mutação, para cada gene do indivíduo, um outro número é gerado aleatoriamente, também no intervalo $[0, 1]$, se esse número é menor que ind_{pb} , o gene é modificado. Do contrário, o gene permanece com o mesmo valor (RAINVILLE *et al.*, 2021).

3.2.5 Seleção de Substituição

No presente trabalho, duas seleções de substituição serão implementadas e testadas no RCGA. A seleção $(\mu + \lambda)$ será testada e avaliada em relação à rápida convergência (*exploitation*), e a seleção (μ, λ) será testada e avaliada em relação à capacidade de superar ótimos locais (*exploitation*). Para a utilização das seleções mencionadas anteriormente, considera-se que pais são selecionados aleatoriamente, par a par, e são recombinaados, gerando dois filhos, até que λ filhos tenham sido criados. Alguns filhos sofrem modificações pelo operador de mutação. Os valores de μ e λ são definidos pelo usuário. Com o conjunto de λ filhos e a população da geração atual com μ indivíduos, na seleção $(\mu + \lambda)$, μ sobreviventes devem ser selecionados considerando os dois conjuntos (população atual \cup filhos). Já na seleção (μ, λ) , μ sobreviventes são selecionados somente entre o conjunto de filhos gerados. A seleção dos sobreviventes marca a transição entre a geração atual e a próxima.

3.3 IMPLEMENTAÇÃO E VALIDAÇÃO DO RCGA

O RCGA do presente trabalho é implementado em Python com a biblioteca DEAP (*Distributed Evolutionary Algorithms in Python*).

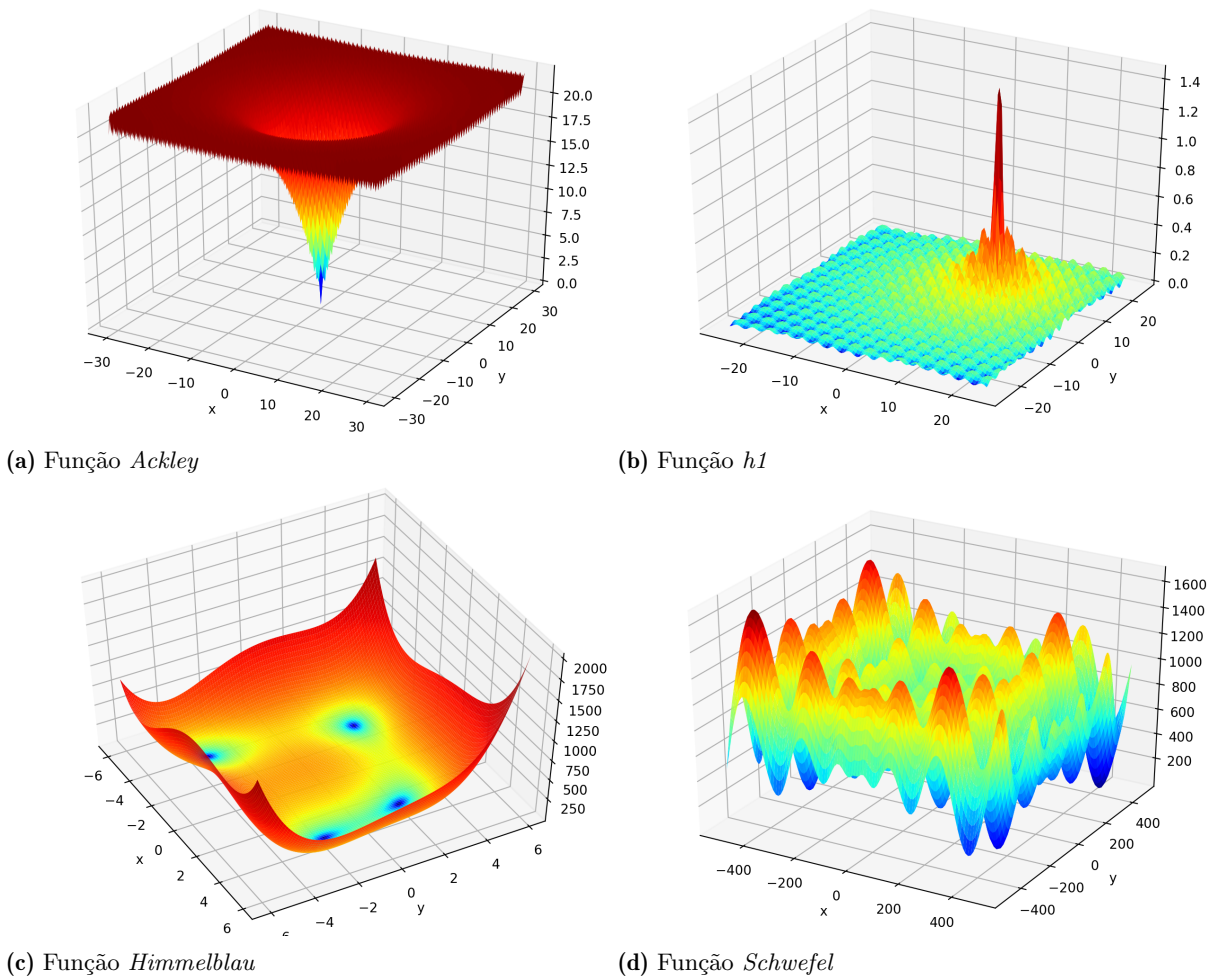
DEAP é um *framework* construído sobre a linguagem de programação Python, cujo objetivo é prover ferramentas práticas para a rápida prototipagem de EAs personalizados, em que cada etapa do processo é tão explícita e de fácil leitura e entendimento quanto possível. O DEAP valoriza tanto a compactação quanto a clareza do código (FORTIN *et al.*, 2012).

O objetivo do DEAP é encorajar os usuários a escreverem seus próprios códigos, e a controlarem cada aspecto do processo evolutivo: tipos de dados, medição de aptidão, inicialização da população, operadores, ciclo evolutivo, etc. Além disso, o *framework* busca

forneer paralelismo de forma transparente, tanto quanto possível (RAINVILLE *et al.*, 2012).

A implementação do RCGA do presente trabalho, com auxílio do *framework* DEAP, é apresentado no Apêndice B, conforme as etapas apresentadas no fluxograma da Figura 11. Esta implementação será testada com as funções de *benchmark* disponíveis no módulo de *benchmarks* existente no *framework* DEAP, como cita Rainville *et al.* (2012). As funções utilizadas para o teste de validação são as funções de *Ackley*, *h1*, *Himmelblau* e *Schwefel*, cujos gráficos são apresentados na Figura 12, e informações no Quadro 1.

Figura 12 – Funções de Benchmark



Fonte: DEAP (2021)

Para o teste com as funções do Quadro 1, considera-se os parâmetros $n_{gen} = 200$, $p_c = 70\%$, $p_m = 30\%$, $ind_{pb} = 20\%$, $\mu = 200$ e $\lambda = 400$. Para as funções *Ackley* e *Schwefel*, considerou-se $N = 5$. O desempenho do RCGA será analisado pela consistência em convergir para as soluções ótimas, considerando uma precisão de seis casas decimais. Para o teste, serão consideradas dez execuções do algoritmo para cada função de *benchmark*.

Quadro 1 – Funções de *Benchmark* para validação do RCGA

Função Ackley	
Função	$f(x) = 20 - 20 \exp \left(-0,2 \sqrt{\frac{1}{N} \sum_{i=1}^N x_i^2} \right) + e - \exp \left(\frac{1}{N} \sum_{i=1}^N \cos(2\pi x_i) \right)$
Tipo	minimização
Faixa	$x_i \in [-15, 30]$
Ótimo Global	$x_i = 0, \forall i \in \{1, \dots, N\}, f(x) = 0$
Função h1	
Função	$f(X) = \frac{\sin(x_1 - \frac{x_2}{8})^2 + \sin(x_2 + \frac{x_1}{8})^2}{\sqrt{(x_1 - 8,6998)^2 + (x_2 - 6,7665)^2 + 1}}$
Tipo	maximização
Faixa	$x_i \in [-100, 100]$
Ótimo Global	$X = (8,6998, 6,7665), f(x) = 2$
Função Himmelblau	
Função	$f(x_1, x_2) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$
Tipo	minimização
Faixa	$x_i \in [-6, 6]$
Ótimo Global	$X_1 = (3, 0, 2, 0), f(X_1) = 0$ $X_2 = (-2, 805118, 3, 131312), f(X_2) = 0$ $X_3 = (-3, 779310, -3, 283186), f(X_3) = 0$ $X_4 = (3, 584428, -1, 848126), f(X_4) = 0$
Função Schwefel	
Função	$f(x) = 418,9828872724339 \cdot N - \sum_{i=1}^N x_i \sin \left(\sqrt{ x_i } \right)$
Tipo	minimização
Faixa	$x_i \in [-500, 500]$
Ótimo Global	$x_i = 420,96874636, \forall i \in \{1, \dots, N\}, f(x) = 0$

Fonte: DEAP (2021)

3.4 OTIMIZAÇÃO TERMOECONÔMICA

3.4.1 Validação do Modelo

Antes de proceder com a otimização termoeconômica via RCGA do sistema térmico de geração de energia por incineração de resíduos sólidos, analisado por Galante (2019), é necessário garantir que o modelo do sistema implementado na linguagem Python

é válido. Para a validação, comparam-se os valores de temperatura (T), pressão (P), vazão mássica (\dot{m}), exergia específica (e), taxa de exergia (\dot{E}), custo médio por unidade de exergia (c) e taxa de custo (\dot{C}) em cada ponto do caso padrão, além do valor de *payback*, sendo permitida uma diferença relativa máxima de 0,5%.

3.4.2 Restrições do Modelo

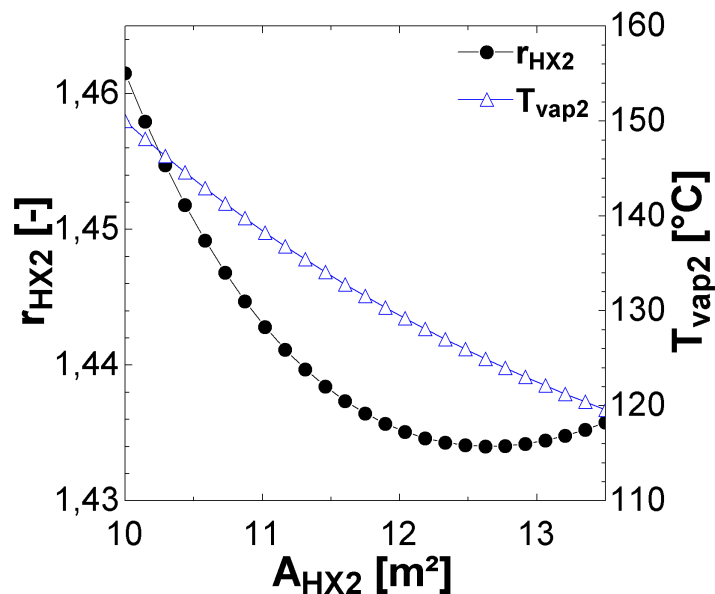
Como mencionado anteriormente, Galante (2019) realiza a otimização do sistema analisando cada equipamento individualmente, e realizando a minimização de seu custo relativo (r_k), calculado pela Equação (20). A minimização de (r_k) é realizada pela variação de algum parâmetro, mas mantendo a taxa de produto ($\dot{E}_{P,k}$) e o custo de combustível do equipamento ($c_{F,k}$) constantes. Como exemplo, a Tabela 1, com as condições para minimização de r_{HX2} , e a Figura 13, com o resultado da minimização de r_{HX2} em função da área de troca térmica de HX2 (A_{HX2}), são apresentadas no presente trabalho, ilustrando o procedimento de otimização deste trocador de calor.

Tabela 1 – Condições para minimização de r_{HX2} em função do produto e combustíveis do equipamento HX2

Variáveis para otimização	Valor fixado
$\dot{E}_{P,HX2} = \dot{E}_{vap3} - \dot{E}_{vap2}$	111.848,412 kJ/h
$c_{F,HX2} = (\dot{C}_{gás6} - \dot{C}_{gás7}) / (\dot{E}_{gás6} - \dot{E}_{gás7})$	48,160E – 6 R\$/kJ

Fonte: Galante (2019)

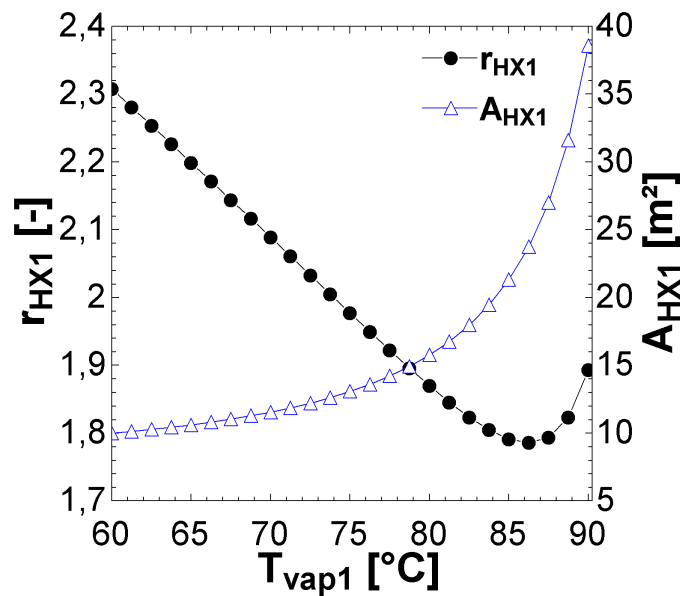
Figura 13 – Resultado para a minimização de r_{HX2} em função da área de troca térmica de HX2



Fonte: Galante (2019)

Em alguns casos, Galante (2019) utiliza como parâmetro de minimização de r_k alguma propriedade termofísica em um ponto específico, deixando características construtivas do equipamento como resultado da variação do valor desta propriedade. A Figura 14 mostra a minimização de r_{HX1} pela variação da temperatura do vapor no ponto vap1, resultando na variação da área de troca térmica deste equipamento. Independentemente da escolha de parâmetro, Galante (2019) mantém fixados os valores de $\dot{E}_{P,k}$ e $\dot{C}_{F,k}$, o que é exigido para que os equipamentos sejam isolados matematicamente do restante do sistema durante o processo de otimização.

Figura 14 – Resultado para a minimização de r_{HX1} em função de T_{vap1}



Fonte: Galante (2019)

As variáveis utilizadas para a minimização do custo relativo dos equipamentos e seus respectivos limites inferiores e superior são mostrados na Tabela 2. Dos equipamentos analisados, somente a otimização da turbina, incinerador e trocadores de calor HX1 e HX5 resultaram em uma redução do período de *payback*. Além disso, a implementação de todos os equipamentos otimizados simultaneamente é conflitante, sendo necessário alguns ajustes. Os pontos ótimos de operação de cada um dos equipamentos analisados são apresentados na Tabela 3 (GALANTE, 2019).

Diferentemente da metodologia utilizada por Galante (2019), a otimização com o RCGA analisa a planta como um todo, e não exige que os valores de $\dot{E}_{P,k}$ e $c_{F,k}$ de cada equipamento sejam mantidos constantes, já que esta condição é utilizada apenas para isolar matematicamente o equipamento do restante do sistema. Portanto, para a otimização do sistema térmico via RCGA, deu-se preferência à utilização de propriedades termofísicas como parâmetros de otimização, deixando as características construtivas dos equipamentos como resultado da variação destes parâmetros. Isso permite que as

características construtivas de mais de um equipamento sejam modificadas pela variação de uma única variável, por exemplo, sem que haja conflito na implementação. Neste cenário, os sistemas individualmente podem não operar em suas condições ótimas, no entanto, essa condição vai de encontro à afirmação de Bejan, Tsatsaronis e Moran (1995), de que a otimização dos equipamentos individualmente não garante a otimização do sistema como um todo. Para a otimização do sistema, é razoável que equipamentos operem em um ponto entre a operação “péssima” e ótima.

Tabela 2 – Variáveis escolhidas por Galante (2019) para a otimização dos equipamentos e seus respectivos limites

Equipamento	Variável	Limite inferior	Limite Superior
Turbina	T_{vap4} [°C]	250	650
HX2	A_{HX2} [m ²]	10	14
Incinerador	T_{ar2} [°C]	125	500
Condensador	A_{cond} [m ²]	10	30
HX5	T_{resf4} [°C]	45	95
HX1	T_{vap1} [°C]	60	90
HX4	T_{ar2} [°C]	125	135

Fonte: Adaptado de Galante (2019)

Tabela 3 – Variáveis que determinam o ponto ótimo de cada um dos equipamentos

Variável	Equipamentos						
	Turb	HX2	Inc	Cond	HX5	HX1	HX4
$\dot{m}_{gás}$ [kg/h]		453,7					463,3
\dot{m}_{vap} [kg/h]	110,34	153,5		145		198,05	
T_{vap1} [°C]				49,6		86,25	
T_{vap2} [°C]		124,8					
T_{vap4} [°C]	532,4						
T_{ar2} [°C]			390				132
T_{resf4} [°C]					87,8		
A_k [°C]		12,65		22,5	12,5	23,74	21

Fonte: Galante (2019)

As propriedades termofísicas escolhidas para a otimização do sistema térmico via RCGA são: temperaturas no ponto vap4 (T_{vap4}), vap2 (T_{vap2}), ar2 (T_{ar2}), vap1 (T_{vap1}), gás5 ($T_{gás5}$), resf4 (T_{resf4}) e vazão mássica de gás de combustão no ponto gás5 ($\dot{m}_{gás5}$). Além das propriedades termofísicas citadas, a eficiência da turbina (η_{turb}) será variada de 55% a 60,2%, mantendo uma potência de saída de 15 kW, conforme implementação em Galante *et al.* (2019). Os parâmetros de otimização escolhidos, e seus respectivos limites inferiores e superiores são apresentados na Tabela 4.

Tabela 4 – Variáveis escolhidas para a otimização via RCGA do sistema térmico em estudo

Variável	Limite inferior	Limite Superior
T_{vap4} [°C]	250	650
T_{vap2} [°C]	100	200
T_{ar2} [°C]	125	500
T_{vap1} [°C]	60	90
$T_{gás5}$ [°C]	900	1400
T_{resf4} [°C]	45	95
$\dot{m}_{gás5}$ [kg/h]	300	700
η_{turb} [%]	55	60,2

Fonte: Autoria própria

Os limites das variáveis de otimização mostrados na Tabela 4 são rearranjados, reescritos como desigualdades do tipo menor ou igual, e adicionadas como restrições ao problema de otimização do sistema térmico de geração de energia por incineração de resíduos sólidos, que é escrito como o seguinte NLP:

$$\begin{aligned}
 \min \quad & f(x) = \text{payback}(T_{vap4}, T_{vap2}, T_{ar2}, T_{vap10}, T_{gás5}, T_{resf4}, \dot{m}_{gás5}, \eta_{turb}), \\
 \text{sujeita a} \quad & g_1(x) = T_{vap4} - 650 \leq 0, \\
 & g_2(x) = 250 - T_{vap4} \leq 0, \\
 & g_3(x) = T_{vap2} - 200 \leq 0, \\
 & g_4(x) = 100 - T_{vap2} \leq 0, \\
 & g_5(x) = T_{ar2} - 500 \leq 0, \\
 & g_6(x) = 125 - T_{ar2} \leq 0, \\
 & g_7(x) = T_{vap1} - 90 \leq 0, \\
 & g_8(x) = 60 - T_{vap1} \leq 0, \\
 & g_9(x) = T_{gás5} - 1400 \leq 0, \\
 & g_{10}(x) = 900 - T_{gás5} \leq 0, \\
 & g_{11}(x) = T_{resf4} - 95 \leq 0, \\
 & g_{12}(x) = 45 - T_{resf4} \leq 0, \\
 & g_{13}(x) = \dot{m}_{gás5} - 700 \leq 0, \\
 & g_{14}(x) = 300 - \dot{m}_{gás5} \leq 0, \\
 & g_{15}(x) = \eta_{turb} - 0,602 \leq 0, \\
 & g_{16}(x) = 0,55 - \eta_{turb} \leq 0, \\
 & g_{17}(x) = \varepsilon_{E,vent} - 1 \leq 0,
 \end{aligned}$$

$$\begin{aligned}
g_{18}(x) &= -\varepsilon_{E,vent} \leq 0, \\
g_{19}(x) &= \varepsilon_{E,inc} - 1 \leq 0, \\
g_{20}(x) &= -\varepsilon_{E,inc} \leq 0, \\
g_{21}(x) &= \varepsilon_{E,pump} - 1 \leq 0, \\
g_{22}(x) &= -\varepsilon_{E,pump} \leq 0, \\
g_{23}(x) &= \varepsilon_{E,turb} - 1 \leq 0, \\
g_{24}(x) &= -\varepsilon_{E,turb} \leq 0, \\
g_{25}(x) &= \varepsilon_{E,cond} - 1 \leq 0, \\
g_{26}(x) &= -\varepsilon_{E,cond} \leq 0, \\
g_{27}(x) &= \varepsilon_{E,poscomb} - 1 \leq 0, \\
g_{28}(x) &= -\varepsilon_{E,poscomb} \leq 0, \\
g_{29}(x) &= \varepsilon_{E,HX3} - 1 \leq 0, \\
g_{30}(x) &= -\varepsilon_{E,HX3} \leq 0, \\
g_{31}(x) &= \varepsilon_{E,HX2} - 1 \leq 0, \\
g_{32}(x) &= -\varepsilon_{E,HX2} \leq 0, \\
g_{33}(x) &= \varepsilon_{E,HX1} - 1 \leq 0, \\
g_{34}(x) &= -\varepsilon_{E,HX1} \leq 0, \\
g_{35}(x) &= \varepsilon_{E,HX4} - 1 \leq 0, \\
g_{36}(x) &= -\varepsilon_{E,HX4} \leq 0, \\
g_{37}(x) &= \varepsilon_{E,HX5} - 1 \leq 0, \\
g_{38}(x) &= -\varepsilon_{E,HX5} \leq 0, \\
g_{39}(x) &= \varepsilon_{E,ciclone} - 1 \leq 0, \\
g_{40}(x) &= -\varepsilon_{E,ciclone} \leq 0, \\
g_{41}(x) &= \varepsilon_{E,FBR} - 1 \leq 0, \\
g_{42}(x) &= -\varepsilon_{E,FBR} \leq 0, \\
g_{43}(x) &= \varepsilon_{E,sys1} - 1 \leq 0, \\
g_{44}(x) &= -\varepsilon_{E,sys1} \leq 0, \\
g_{45}(x) &= \varepsilon_{E,sys2} - 1 \leq 0, \\
g_{46}(x) &= -\varepsilon_{E,sys2} \leq 0,
\end{aligned} \tag{31}$$

As restrições em um determinado problema surgem por limitações de espaço, do equipamento, e dos materiais empregados. Isto é, pode ser necessário restringir dimensões físicas, temperatura, pressão, vazão mássica, etc. Muitas restrições de sistemas térmicos existem por conta de leis de conservação (e.g., massa e energia). Muitas dessas restrições são satisfeitas durante a modelagem e simulação, visto que as equações que governam o modelo

baseiam-se em princípios de conservação. Deste modo, a função objetivo a ser otimizada já considera tais restrições, sendo necessário considerar apenas restrições adicionais que definem os limites do domínio do problema (DINÇER; ROSEN; AHMADI, 2017).

No NLP acima, as restrições g_1 a g_{16} referem-se aos limites inferiores e superiores dos parâmetros de otimização. Já as restrições g_{17} a g_{46} referem-se às eficiências exergéticas dos equipamentos e do sistema, cujos valores devem estar entre 0% e 100%, para que a 2ª Lei da Termodinâmica não seja violada.

Em relação aos limites das restrições, os operadores genéticos SBX e Mutação Polinomial garantem que todos os indivíduos gerados permaneçam dentro dos limites inferiores e superiores estabelecidos, como pode ser observado em Deb e Agrawal (1999), Deb *et al.* (2002) e Deb e Deb (2012). Ainda assim, algumas combinações de parâmetros não solucionam o problema, e a região viável (i.e., domínio do problema) não é conhecida analiticamente devido à complexidade da função objetivo. Nesse cenário, a aplicação de penalidade aos indivíduos inviáveis, com base no nível de violação da região viável, é dificultada por falta de informação. Lee *et al.* (2011), por exemplo, sugere emulação estatística para lidar com restrições ocultas. No presente trabalho, no entanto, por facilidade de implementação, será utilizada penalidade de morte para os indivíduos inviáveis, aplicando-se um valor infinito de *payback* para os mesmos. Apesar da facilidade de implementação, a função de penalidade pode prejudicar o desempenho do RCGA, caso haja poucas soluções viáveis nas iterações iniciais, podendo resultar na estagnação do mesmo em um ótimo local. No presente trabalho, são considerados indivíduos inviáveis aqueles que não solucionam o problema, ou que violam a 2ª Lei da Termodinâmica ao não respeitarem as restrições de eficiência exergética impostas pelas restrições g_{17} a g_{46} , da Equação (31).

A otimização do sistema térmico de geração de energia por incineração de resíduos sólidos via RCGA será implementada utilizando os parâmetros $n_{gen} = 100$, $p_c = 70\%$, $p_m = 30\%$, $ind_{pb} = 20\%$, $\mu = 50$ e $\lambda = 100$. A redução do valor dos parâmetros n_{gen} , μ e λ se deve ao maior custo computacional da função *payback* em relação às funções de *benchmark*. A convergência do algoritmo será analisada por trinta execuções do algoritmo genético, para as seleções $(\mu + \lambda)$ e (μ, λ) . Além do teste de desempenho e convergência do RCGA, as soluções ótimas encontradas serão analisadas termodinamicamente, e comparadas às soluções encontradas por Galante (2019). Ambos os cenários, RSU sem custo e RSU com custo associado a uma fração do custo de incineração, serão considerados. No segundo cenário, considera-se para o custo do RSU, uma fração de 3% do que seria o serviço de incineração. O Apêndice B contém o script da implementação do RCGA em Python, com auxílio do *framework* DEAP.

4 RESULTADOS E DISCUSSÕES

Esta seção tem por finalidade apresentar a validação do modelo do sistema térmico de geração de energia por incineração de resíduos sólidos implementado em Python, e do RCGA utilizado para a otimização termoeconômica deste modelo. A implementação em Python é comparada com a implementação em *Engineering Equation Solver* (EES), considerando o caso padrão apresentado por Galante (2019). O RCGA, por sua vez, é analisado pela taxa de sucesso em otimizar as funções de *benchmark* apresentadas no Quadro 1, exigindo uma variação máxima de $1,0\text{e-}4\%$ entre o valor ótimo conhecido e o valor ótimo encontrado.

4.1 VALIDAÇÃO DO MODELO

Os resultados da implementação em Python e EES da modelagem do sistema térmico de geração de energia por incineração de resíduos sólidos são comparados. A Tabela 5 contém os valores de temperatura, pressão e vazão mássica em pontos de interesse da planta. A coluna mais à direita traz o módulo da maior diferença relativa apresentada em cada estado analisado, comparando-se as diferenças entre as temperaturas, pressões e vazões mássicas. A maior diferença apresentada em todos os pontos comparados refere-se à vazão mássica do estado Água10, com um valor de $0,149\%$, menor que a diferença máxima estabelecida de $0,5\%$, sendo o modelo considerado válido em relação à temperatura, pressão e vazão mássica.

As implementações em Python e EES também são comparadas em relação à exergia específica, taxa de exergia, custo médio por unidade de exergia e taxa de custo, cujos valores são apresentados na Tabela 6. A coluna mais à direita traz o módulo da maior diferença relativa apresentada em cada estado analisado, comparando-se as diferenças entre as exergias específicas, taxas de exergia, custos médio por unidade de exergia e taxas de custo. A maior diferença apresentada em todos os pontos comparados refere-se à taxa de exergia no ponto CBG, com um valor de $0,38\%$, menor que a diferença máxima estabelecida de $0,5\%$, sendo o modelo considerado válido também em relação à exergia específica, taxa de exergia, custo médio por unidade de exergia e taxa de custo.

Quanto ao *payback* do caso padrão, o valor calculado é de 3,889 anos pela implementação em EES, e de 3,887 pela implementação em Python, o que representa uma diferença relativa de $0,048\%$, sendo o modelo considerado válido para ser utilizado como função objetivo do problema de otimização do sistema térmico. Na próxima seção, a validação do RCGA em relação à otimização de funções de *benchmark* é apresentada.

Tabela 5 – Comparação entre valores de temperatura, pressão e vazão mássica

Ponto	Python			EES			Δ [%]
	T [°C]	P [kPa]	\dot{m} [kg/h]	T [°C]	P [kPa]	\dot{m} [kg/h]	
Ar1	25,00	101,33	476,34	25,00	101,32	476,28	0,01
Ar2	125,00	101,33	476,34	125,00	101,32	476,28	0,01
Vent_{in}	125,00	101,33	476,34	125,00	101,32	476,30	0,01
Vent_{out}	125,57	201,32	476,34	125,40	201,30	476,30	0,13
RSU	25,00	101,33	47,83	25,00	101,32	47,82	0,02
CBG	25,00	101,33	1,39	25,00	101,32	1,39	0,11
Gás3	844,10	101,33	502,43	843,90	101,32	502,30	0,03
Gás5	900,00	101,33	525,00	900,00	101,32	525,00	0,00
Gás6	860,59	101,33	525,00	861,00	101,32	525,00	0,05
Gás7	375,90	101,33	525,00	376,20	101,32	525,00	0,08
Gás8	282,78	101,33	525,00	283,10	101,32	525,00	0,11
Gás9	207,58	101,33	525,00	207,80	101,32	525,00	0,11
Gás10	40,00	101,33	449,57	40,00	101,32	449,64	0,02
Água10	40,00	101,33	75,43	40,00	101,32	75,54	0,15
Gás11	39,72	101,33	449,43	39,72	101,32	449,28	0,03
Vap1	60,00	800,00	160,01	60,00	800,00	160,00	0,00
Vap2	150,00	800,00	160,01	150,00	800,00	160,00	0,00
Vap3	170,41	800,00	160,01	170,40	800,00	160,00	0,00
Vap4	250,00	800,00	160,01	250,00	800,00	160,00	0,00
Vap5	59,90	19,86	160,01	59,90	19,84	160,00	0,09
Liq6	59,90	19,86	160,01	59,90	19,84	160,00	0,09
Resf1	25,00	101,33	4.502,70	25,00	101,32	4.497,00	0,13
Resf2	45,00	101,33	4.502,70	45,00	101,32	4.497,00	0,13
Resf3	25,00	101,33	1.249,91	25,00	101,32	1.251,40	0,12
Resf4	45,00	101,33	1.249,91	45,00	101,32	1.251,00	0,09

Fonte: Autoria Própria

Tabela 6 – Comparação entre valores de exergia específica, taxa de exergia, custo médio por unidade de exergia e taxa de custo

Ponto	Python				EES				Δ [%]
	e[kJ/kg]	\dot{E} [kJ/h]	c[R\$/kJ]	\dot{C} [R\$/h]	e[kJ/kg]	\dot{E} [kJ/h]	c[R\$/kJ]	\dot{C} [R\$/h]	
Ar1	48,93	23.309,03	0,00	0,00	48,93	23.309,00	0,00	0,00	0,01
Ar2	63,09	30.052,92	5,06E-05	1,52	63,09	30.051,00	5,06E-05	1,52	0,08
Vent_{in}	63,09	30.052,92	5,06E-05	1,52	63,09	30.051,00	5,06E-05	1,52	0,08
Vent_{out}	63,24	30.122,95	8,20E-05	2,47	63,20	30.107,00	8,21E-05	2,47	0,11
RSU	17.768,55	849.878,27	0,00	0,00	17.769,00	849.731,00	0,00	0,00	0,02
CBG	48.561,28	67.426,81	1,43E-04	9,62	48.561,00	67.685,00	1,43E-04	9,66	0,38
Gás3	747,63	375.629,44	2,21E-05	8,29	747,30	375.387,00	2,21E-05	8,30	0,15
Gás5	804,71	422.471,01	4,81E-05	20,30	804,60	422.389,00	4,82E-05	20,35	0,30
Gás6	763,16	400.660,36	4,81E-05	19,25	763,40	400.790,00	4,82E-05	19,31	0,30
Gás7	334,60	175.665,97	4,82E-05	8,44	334,90	175.807,00	4,82E-05	8,47	0,34
Gás8	276,10	144.952,28	4,81E-05	6,97	276,20	145.028,00	4,82E-05	6,99	0,35
Gás9	237,23	124.545,50	4,81E-05	5,99	237,30	124.579,00	4,82E-05	6,00	0,30
Gás10	141,04	63.409,09	4,81E-05	3,05	141,00	63.352,00	4,82E-05	3,05	0,30
Água10	1,53	115,11	0,00	0,00	1,53	115,40	0,00	0,00	0,26
Gás11	141,07	63.402,49	5,57E-05	3,53	141,00	63.345,00	5,59E-05	3,54	0,30
Vap1	8,67	1.387,19	1,78E-04	0,25	8,67	1.387,00	1,79E-04	0,25	0,40
Vap2	87,91	14.066,98	1,61E-04	2,26	87,97	14.075,00	1,61E-04	2,27	0,35
Vap3	786,70	125.878,61	1,23E-04	15,50	787,00	125.923,00	1,23E-04	15,52	0,14
Vap4	855,98	136.963,01	1,25E-04	17,09	855,70	136.913,00	1,25E-04	17,10	0,19
Vap5	254,38	40.702,47	1,25E-04	5,08	254,30	40.684,00	1,25E-04	5,08	0,19
Liq6	7,85	1.255,81	1,25E-04	0,16	7,85	1.256,00	1,25E-04	0,16	0,19
Resf1	0,00	0,00	3,02E-03	0,00	0,00	3,02E-03	0,00	0,00	0,08
Resf2	2,68	12.086,50	4,88E-04	5,89	2,69	12.081,00	4,88E-04	5,90	0,21
Resf3	0,00	0,00	3,02E-03	0,00	0,00	3,02E-03	0,00	0,00	0,08
Resf4	2,68	3.355,10	1,03E-03	3,46	2,69	3.360,00	1,03E-03	3,47	0,33

Fonte: Autoria Própria

4.2 VALIDAÇÃO DO RCGA

Para a validação do RCGA, o mesmo é executado dez vezes para cada função objetivo e seleção de substituição. A Tabela 7 e a Tabela 8 mostram o resultado das dez execuções do RCGA com as seleções de substituição $(\mu + \lambda)$ e (μ, λ) . Para o teste de validação utilizou-se os parâmetros $n_{gen} = 200$, $p_c = 70\%$, $p_m = 30\%$, $ind_{pb} = 20\%$, $\mu = 200$, e $\lambda = 400$.

Tabela 7 – Resultados da otimização das funções de *Benchmark* com o RCGA, utilizando a seleção $(\mu + \lambda)$

Execução	Função	Ótimo encontrado	Função	Ótimo encontrado
1	<i>Ackley</i>	7,105e-15	<i>h1</i>	2
2		7,105e-15		2
3		3,553e-15		2
4		7,105e-15		2
5		7,105e-15		2
6		1,421e-14		2
7		7,105e-15		2
8		7,105e-15		2
9		3,553e-15		2
10		3,553e-15		2
1	<i>Himmelblau</i>	7,889e-29	<i>Schwefel</i>	4,547e-13
2		2,524e-29		4,547e-13
3		1,262e-29		4,547e-13
4		4,102e-29		4,547e-13
5		1,546e-28		4,547e-13
6		1,578e-29		4,547e-13
7		7,889e-29		4,547e-13
8		1,01e-28		4,547e-13
9		7,889e-28		4,547e-13
10		5,049e-29		4,547e-13

Fonte: Autoria Própria

Os resultados da otimização das funções de *benchmark* com a seleção de substituição $(\mu + \lambda)$, mostrados na Tabela 7, indicam a eficácia do RCGA ao convergir para valores muito próximos aos ótimos globais conhecidos das funções, mostrados no Quadro 1. A maior variação encontrada em todas as execuções foi de 4,5e-11%, sendo o RCGA com a seleção $(\mu + \lambda)$ considerado válido.

Tabela 8 – Resultados da otimização das funções de *Benchmark* com o RCGA, utilizando a seleção (μ, λ)

Execução	Função	Ótimo encontrado	Função	Ótimo encontrado
1	<i>Ackley</i>	8,376e-11	<i>h1</i>	2
2		1,702e-10		2
3		9,07e-12		2
4		7,72e-11		2
5		1,191e-10		2
6		1,451e-11		2
7		5,196e-11		2
8		3,842e-11		2
9		2,232e-11		2
10		1,558e-11		2
1	<i>Himmelblau</i>	1,578e-29	<i>Schwefel</i>	4,547e-13
2		3,155e-30		4,547e-13
3		0		4,547e-13
4		0		4,547e-13
5		2,288e-29		4,547e-13
6		7,967e-29		4,547e-13
7		6,311e-30		4,547e-13
8		0		4,547e-13
9		1,641e-28		4,547e-13
10		8,599e-29		4,547e-13

Fonte: Autoria Própria

Os resultados da otimização das funções de *benchmark* com a seleção de substituição (μ, λ) , mostrados na Tabela 8, indicam a eficácia do RCGA ao convergir para valores muito próximos aos ótimos globais conhecidos das funções, mostrados no Quadro 1. Embora a maior variação apresentada tenha sido maior que nas execuções do RCGA com a seleção $(\mu + \lambda)$, em todas as execuções com a seleção (μ, λ) a variação foi menor que 1,8e-8%, sendo o RCGA considerado válido também para este tipo de seleção de substituição.

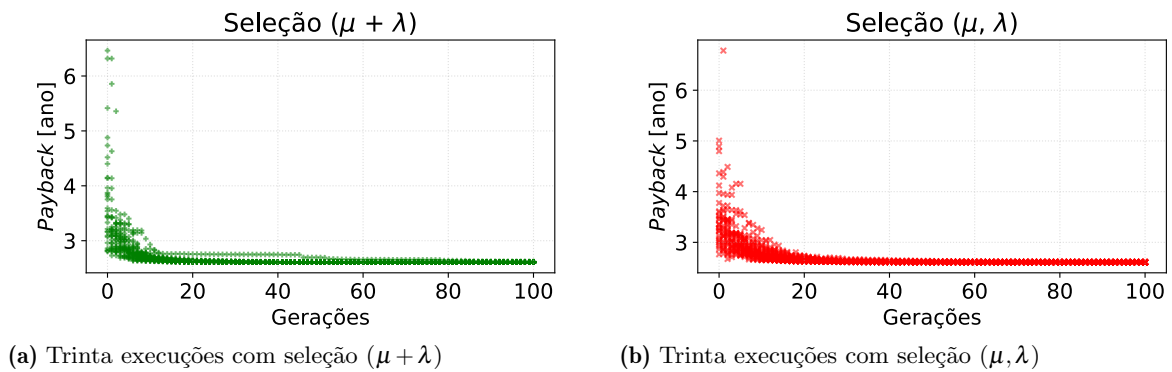
4.3 OTIMIZAÇÃO TERMOECONÔMICA

Com a validação do modelo e do RCGA, a otimização é realizada pela criação e seleção de diversos indivíduos criados pelos operadores genéticos, ao longo das gerações (iterações) do algoritmo. Cada indivíduo é formado por um cromossomo contendo genes, que no caso da codificação real utilizada pelo RCGA, são os próprios parâmetros de otimização. Os parâmetros de otimização de cada indivíduo são utilizados no modelo do sistema, para o qual calcula-se o período de *payaback*, isto é, tempo de recuperação do capital investido. A otimização se dá pela minimização do *payback*, função objetivo do problema.

No presente trabalho, a minimização do *payback* é realizada pelo procedimento apresentado na Figura 11, que é repetido trinta vezes para as seleções de substituição $(\mu + \lambda)$ e (μ, λ) . Além das duas seleções de substituição, considerou-se dois cenários, RSU sem custo e RSU com custo associado à uma fração de 3% do custo pelo serviço de incineração, resultando em cento e vinte execuções do algoritmo. As seleções de substituição serão analisadas com relação às propriedades de *exploration* e *exploitation*.

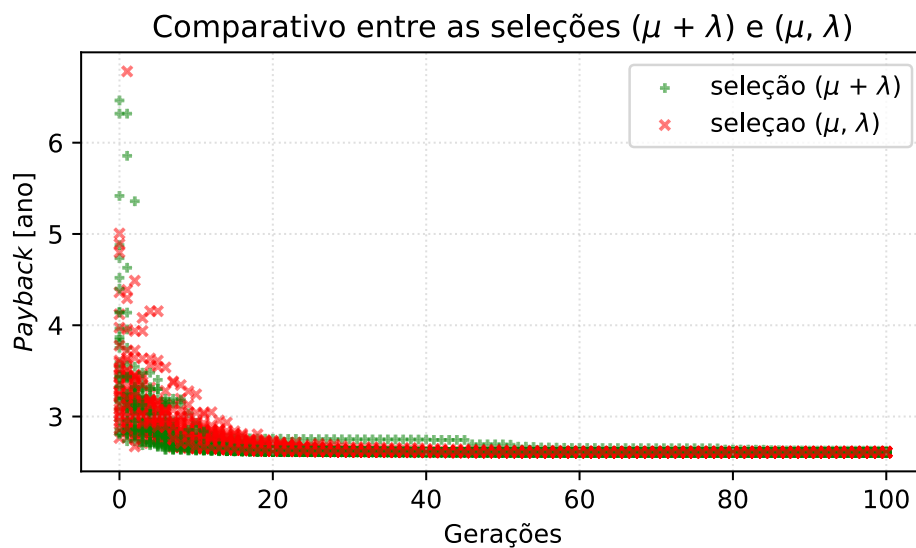
A Figura 15a e a Figura 15b mostram o valor de *payback* das melhores soluções encontradas ao longo das gerações, pelas várias execuções do RCGA, para as seleções $(\mu + \lambda)$ e (μ, λ) , considerando o cenários de RSU sem custo. Esses resultados são sobrepostos e apresentados na Figura 16.

Figura 15 – Melhores soluções encontradas em cada geração, em trinta execuções do RCGA para as soluções $(\mu + \lambda)$ e (μ, λ)



Fonte: Autoria própria

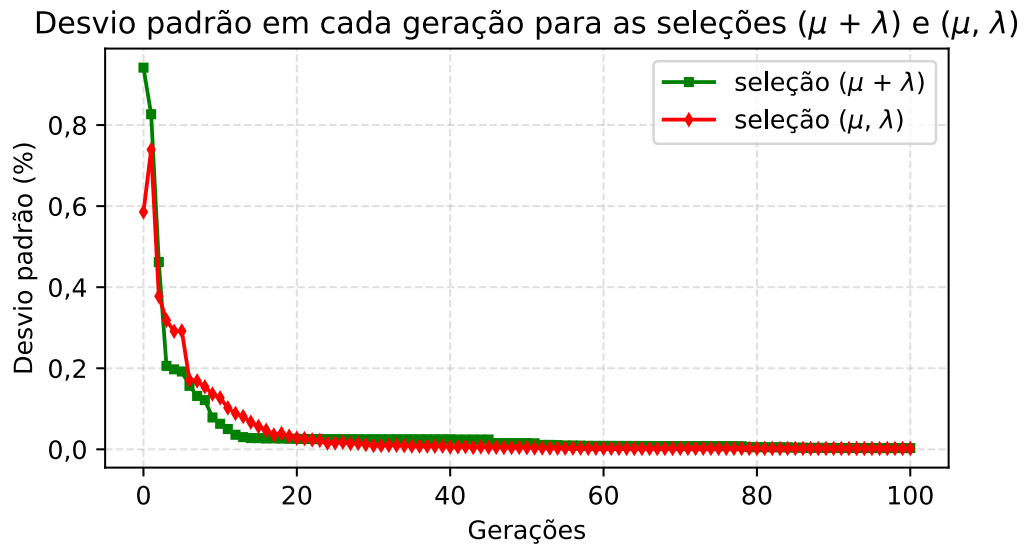
Figura 16 – Melhores soluções encontradas em cada geração, para trinta execuções do RCGA para as soluções $(\mu + \lambda)$ e (μ, λ)



Fonte: Autoria própria

Os resultados da Figura 16 foram considerados inconclusivos para a análise de *exploitation* e *exploration*. Em relação a *exploitation* ambas as seleções de substituição apresentaram uma convergência muito rápida para valores de *payback* próximos a 2,61 anos, já antes da vigésima geração. Em relação a *exploration*, a seleção (μ, λ) aparentou chegar em valores mais distintos de *payback*, o que pode estar relacionado à uma maior distância entre os melhores pontos encontrados encontrados, ou seja, maior diversidade na população. No entanto, essa característica não é observada de forma tão acentuada na Figura 16 para atestar, de fato, uma maior capacidade de *exploitation* pela seleção (μ, λ) . Para uma melhor análise dos melhores pontos encontrados, a Figura 17 traz o desvio padrão das melhores soluções encontradas em cada geração, nas várias execuções do RCGA.

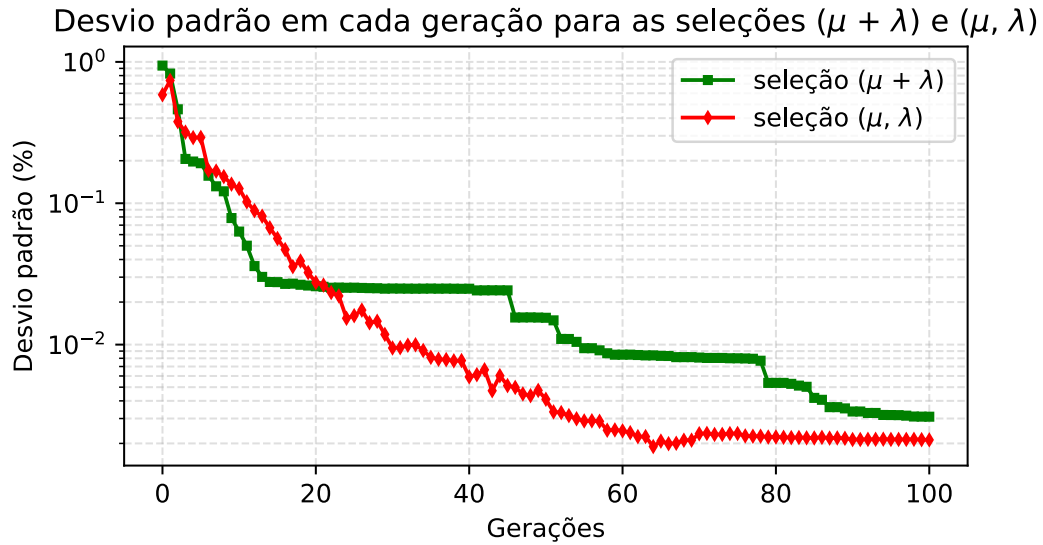
Figura 17 – Desvio padrão do *payback* ao longo das gerações, em trinta execuções do RCGA para cada seleção de substituição



Fonte: Autoria própria

É observável na Figura 17 uma maior distinção dos valores de *payback* para ambas as seleções de substituição, antes da vigésima geração. Após a vigésima geração, pelos pontos apresentados no gráfico, os valores de *payback* aparentam ficar mais próximos entre si, o que pode estar relacionado à diminuição de diversidade na população e o início da convergência para um valor ótimo. Para uma melhor análise, os dados de desvio padrão são apresentados em escala logarítmica na Figura 18.

Figura 18 – Desvio padrão do *payback* ao longo das gerações, em trinta execuções do RCGA para cada seleção de substituição

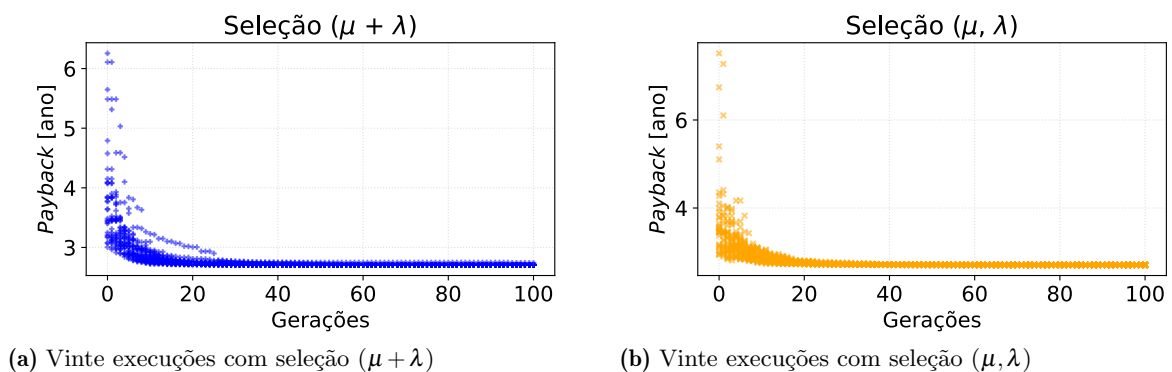


Fonte: Autoria própria

Na Figura 18 fica mais aparente que, antes da vigésima geração, aproximadamente, a seleção (μ, λ) chega em valores de *payback* mais distintos, e após isso o cenário é invertido, com a seleção $(\mu + \lambda)$ apresentando um maior desvio padrão dos valores de *payback* encontrados. Do ponto de vista de *exploitation* e *exploration*, a análise da Figura 17 e Figura 18 foi considerada inconclusiva, sendo recomendada uma quantidade maior de execuções do algoritmo.

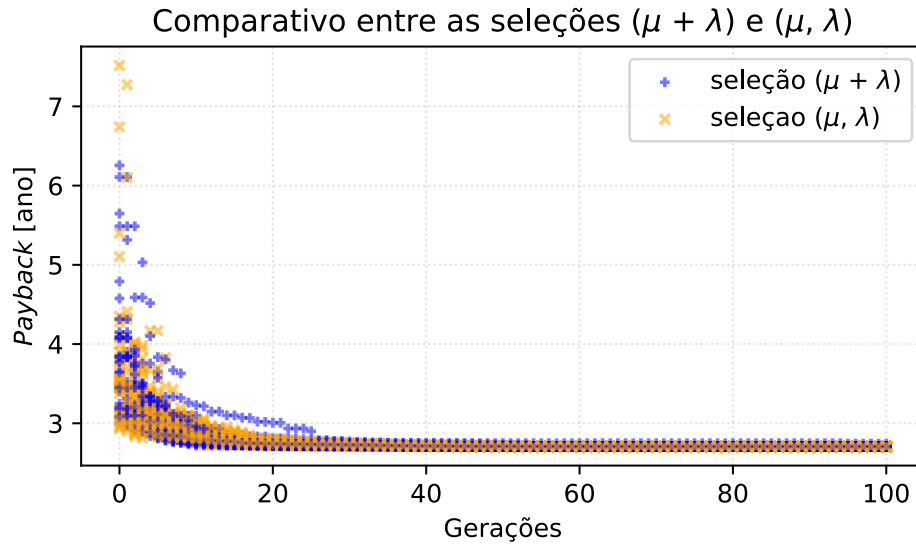
Para o cenário de RSU com custo, a Figura 19a e Figura 19b trazem o valor de *payback* das melhores soluções encontradas pelas várias execuções do RCGA, ao longo das gerações, para as seleções $(\mu + \lambda)$ e (μ, λ) . Esses resultados são sobrepostos e apresentados na Figura 20.

Figura 19 – Melhores soluções encontradas em cada geração do RCGA, para as soluções $(\mu + \lambda)$ e (μ, λ)



Fonte: Autoria própria

Figura 20 – Melhores soluções encontradas em cada geração do RCGA, para as seleções $(\mu + \lambda)$ e (μ, λ)



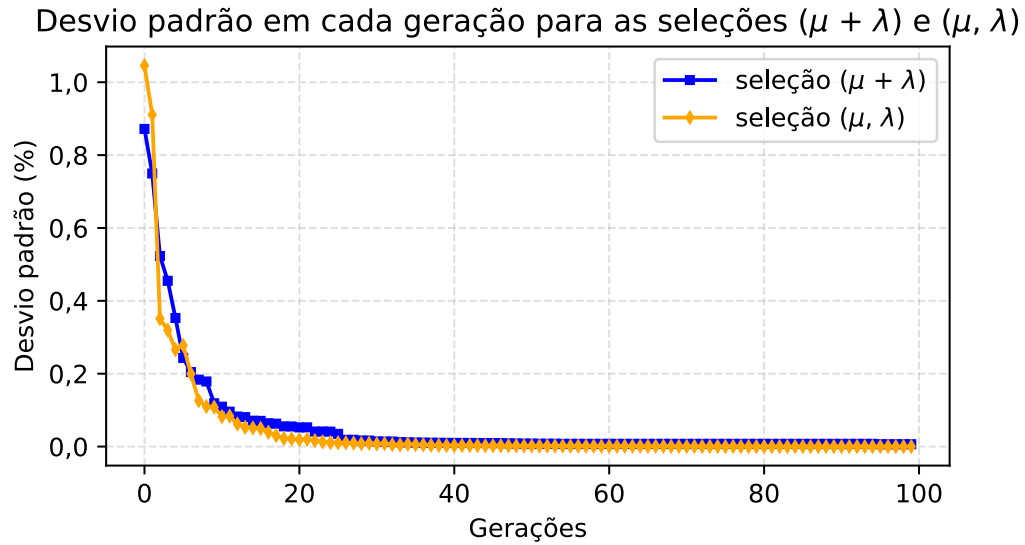
Fonte: Autoria própria

Os resultados da Figura 20 também foram considerados inconclusivos para a análise de *exploitation* e *exploration*. Em relação a *exploitation*, ambas as seleções de substituição apresentaram uma convergência muito rápida para valores de *payback* próximos a 2,71 anos, já antes da vigésima geração. Em relação a *exploration*, a seleção $(\mu + \lambda)$ aparentou chegar em valores mais distintos de *payback*. Esse resultado não vai de encontro ao que consta na literatura. Borne, Popescu e Philip (2014), por exemplo, afirmam que *exploration* está mais relacionada à diversidade da população. Diversidade na população é mais provável de ser observada na seleção (μ, λ) , que aplica menos seleção aos indivíduos da população, se comparada à $(\mu + \lambda)$. Para uma melhor análise dos melhores pontos encontrados, a Figura 21 traz o desvio padrão das melhores soluções encontradas em cada geração, nas várias execuções do RCGA.

É observável na Figura 21 uma maior distinção dos valores de *payback* para ambas as seleções de substituição, antes da vigésima geração. Após a vigésima geração, pelos pontos apresentados no gráfico, os valores de *payback* aparentam ficar mais próximos entre si, o que pode estar relacionado à diminuição de diversidade na população e o início da convergência para um valor ótimo.

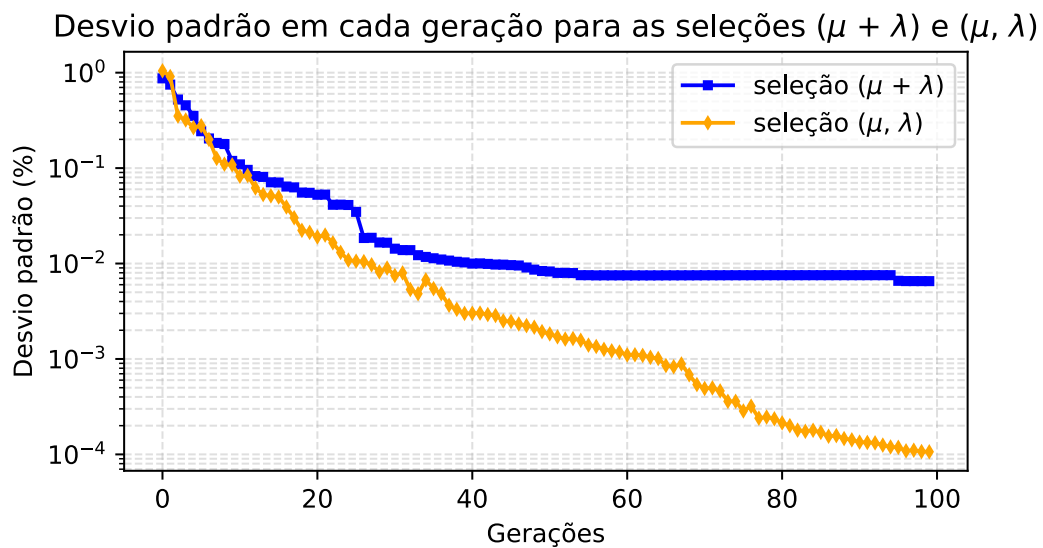
Para uma melhor análise, os dados de desvio padrão são apresentados em escala logarítmica na Figura 22, onde fica evidente a maior diferença entre os valores do *payback* das melhores soluções, em cada geração, encontradas pela seleção $(\mu + \lambda)$. Para uma melhor análise das propriedades de *exploration* e *exploitation* é recomendada a execução do RCGA para outras funções objetivo, sendo considerada inconclusiva na análise do presente trabalho.

Figura 21 – Desvio padrão do *payback* ao longo das gerações, em trinta execuções do RCGA para cada seleção de substituição



Fonte: Autoria própria

Figura 22 – Desvio padrão do *payback* ao longo das gerações, em trinta execuções do RCGA para cada seleção de substituição



Fonte: Autoria própria

Para o cenário de RSU sem custo, a melhor solução é encontrada pela seleção $(\mu + \lambda)$, com um período de *payback* de 2,61 anos. Para o cenário de RSU com custo, a melhor solução é encontrada também pela seleção $(\mu + \lambda)$, com um período de *payback* de 2,71 anos. Considerando o RSU sem custo, a diferença do período de *payback* de 3,06 anos, da otimização de Galante (2019), para 2,61 anos, representa uma redução de 14,71%. Já para o cenário em que considera-se uma fração de 3% do custo do serviço de incineração como custo do RSU, a diferença do período de *payback* de 3,19, da otimização de Galante (2019), para 2,71 anos, representa uma redução de 15,05%. Os valores das variáveis de

otimização das soluções ótimas são mostradas na Tabela 9

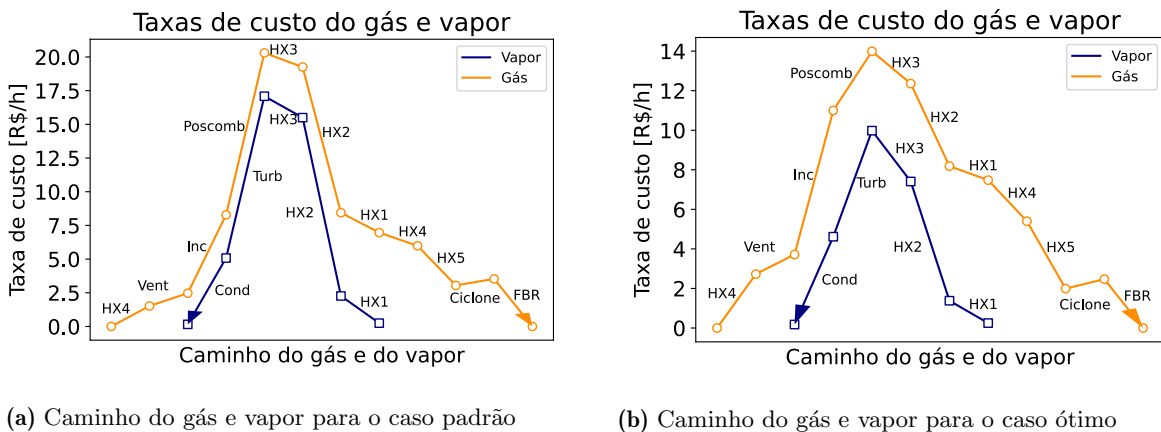
Tabela 9 – Valores das variáveis para as soluções ótimas encontradas

Variável	RSU sem custo	RSU com custo
$T_{\text{vap4}} [^{\circ}\text{C}]$	513,43	516,16
$T_{\text{vap2}} [^{\circ}\text{C}]$	170,40	170,40
$T_{\text{ar2}} [^{\circ}\text{C}]$	260,41	252,58
$T_{\text{vap1}} [^{\circ}\text{C}]$	82,41	90,00
$T_{\text{gás5}} [^{\circ}\text{C}]$	942,58	937,10
$T_{\text{resf4}} [^{\circ}\text{C}]$	45,00	45,00
$\dot{m}_{\text{gás5}} [\text{kg/h}]$	700,00	700,00
$\eta_{\text{turb}} [\%]$	60,20	60,20

Fonte: Autoria própria

Os valores da Tabela 9 mostram um aumento de temperatura nos pontos Vap4, Vap2, Ar2, Vap1, Gás5, um aumento da vazão mássica no ponto Gás5 e da eficiência da turbina, em ambos os cenários analisados (RSU sem custo e RSU com custo). A não variação dos valores da temperatura no ponto Resf4 indica que a mesma pode ser desconsiderada no procedimento de otimização. Para o cenário de RSU sem custo, a variação da taxa de custo de vapor e gás, enquanto estes passam pelos equipamentos do sistema, é mostrada na Figura 23. A Figura 23a traz as informações já apresentadas na Figura 9, mas para a implementação em Python, enquanto a Figura 23b considera o caso ótimo para o cenário de RSU sem custo.

Figura 23 – Taxas de custo do gás e vapor conforme passam pelos equipamentos do sistema



Fonte: Autoria própria

De forma geral, o comportamento de ambos os casos da Figura 23 é similar: a taxa de custo do gás de combustão eleva-se enquanto este passa pelos equipamentos, até o ponto Gás5, a partir de onde o mesmo passa a servir de combustível para os trocadores de calor,

incrementando o valor do vapor do ciclo Rankine. Torna-se evidente, porém a diferença na forma com que as taxas de custos variam conforme passam pelos equipamentos, observadas pelas inclinações das retas. A taxa de custo máxima, em R\$/h, passa de aproximadamente 20,00 no caso padrão para aproximadamente 14,00 no caso ótimo, ambos localizados na entrada do gás no trocador de calor HX3.

A Tabela 10 traz informações da variação de valores de $\varepsilon_{E,k}$ e CI_k , ao comparar-se o caso padrão e o caso ótimo encontrado pela otimização com o RCGA. Os valores são destacados em fonte azul para redução de custo e aumento de eficiência exergética, e em fonte vermelha para aumento de custo e redução de eficiência exergética.

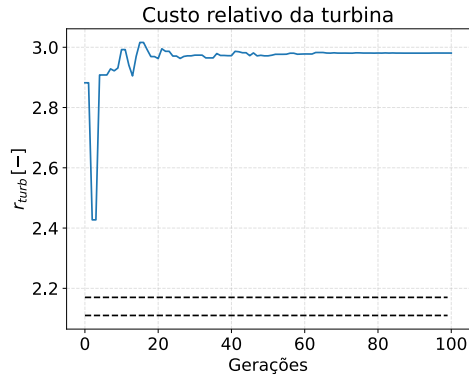
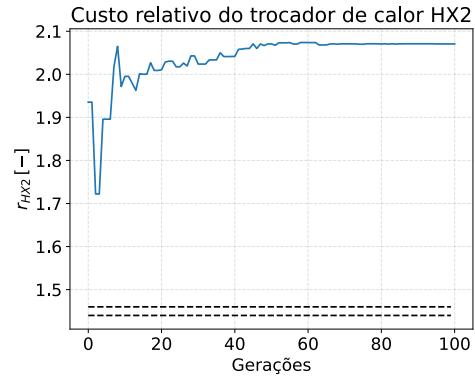
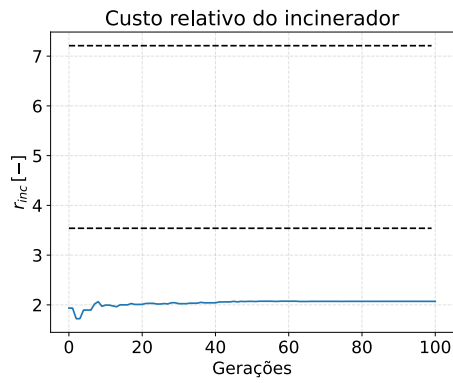
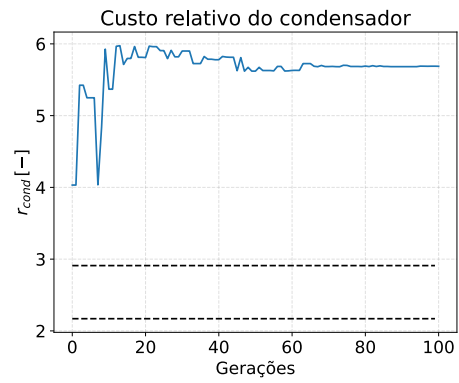
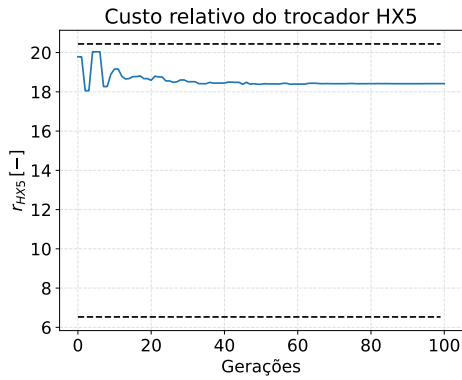
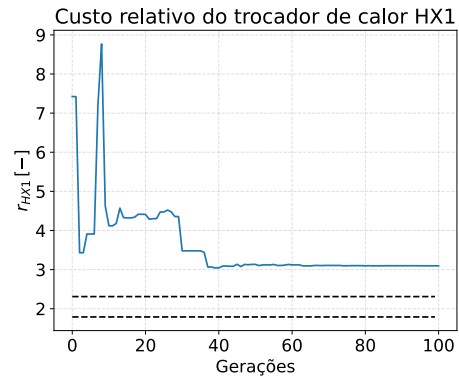
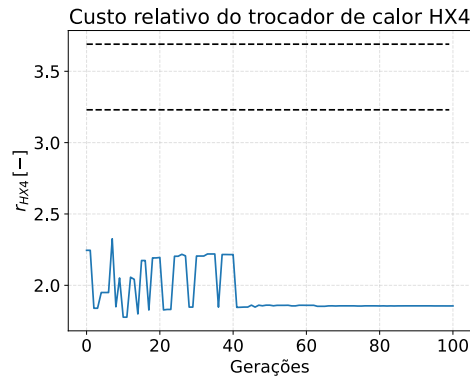
Tabela 10 – Eficiência exergética e capital de investimento dos principais equipamentos do sistema

Equipamento	Caso padrão		Caso ótimo RCGA		$\Delta\varepsilon_{E,k}$ [%]	ΔCI_k [%]
	$\varepsilon_{E,k}$	CI_k [R\$]	$\varepsilon_{E,k}$	CI_k [R\$]		
Turbina	0,58	164.400,00	0,71	164.941,23	22,41	0,33
HX2	0,50	39.870,00	0,47	30.946,81	-6,00	-22,38
Incinerador	0,43	97.500,00	0,48	121.772,66	10,42	19,93
Condensador	0,31	16.000,00	0,17	10.001,45	-45,16	-37,49
Poscomb	0,77	37.500,00	1,00	50.002,14	22,22	33,34
HX5	0,05	8.550,00	0,06	10.158,65	16,67	15,84
HX1	0,41	8.660,00	0,39	7.048,66	-4,87	-18,61
HX4	0,33	8.660,00	0,46	10.829,49	39,39	25,05

Fonte: Autoria própria

Dos resultados apresentados na Tabela 10, destaca-se a eficiência exergética da câmara de pós-combustão em 100%. O sistema exige, segundo Galante (2019), que os gases de combustão cheguem a uma temperatura de 900°C para garantir que não haja liberação de dioxinas. A queima de RSU na câmara de pós-combustão, utilizando gás liquefeito de petróleo (GLP), ocorre somente se a temperatura do gás, após a queima no incinerador, não chegar a 900°C. Na solução ótima encontrada pelo RCGA, a temperatura no ponto Gás3 ($T_{gás3}$), anterior à câmara de pós-combustão, chega a 943°C, tornando a câmara inoperante em relação à queima de RSU com GLP. Verifica-se que, para os equipamentos analisados, o aumento da eficiência exergética interferiu no capital de investimento de forma diretamente proporcional.

Apesar dos valores de r_k não terem sido utilizados no procedimento de otimização com o RCGA, esses valores foram calculados para os melhores indivíduos encontrados em cada geração, na execução do RCGA em que a melhor solução foi encontrada. Os resultados foram plotados na Figura 24, em que os valores de r_k ao longo das gerações são plotados em azul, e as linhas tracejadas em preto referem-se ao valor do r_k calculado para o caso padrão, e o valor do r_k minimizado por Galante (2019).

Figura 24 – Variação de r_{turb} (a) Variação de r_{turb} (b) Variação de r_{HX2} (c) Variação de r_{inc} (d) Variação de r_{cond} (e) Variação de r_{HX5} (f) Variação de r_{HX1} (g) Variação de r_{HX4}

Fonte: Autoria própria

É notável que em alguns casos, os equipamentos operam em condições de r_k maiores que os calculados por Galante (2019) após a otimização, ou até mesmo que os calculados para o caso padrão. Em outros casos, como os do incinerador e do trocador de calor HX4, o r_k calculado ao longo das gerações é menor que o valor mínimo encontrado por Galante (2019), o que pode ter relação com as variáveis utilizadas para a minimização do r_k e para a otimização com o RCGA. Na otimização pelo RCGA, os parâmetros que interferem diretamente no incinerador são: $T_{gás3}$ e $\dot{m}_{gás5}$. No trocador HX4, apenas T_{ar2} tem influência direta, mas é importante ressaltar que todos os equipamentos operam em conjunto, interferindo nos equipamentos anteriores e subsequentes, e que diferentemente da otimização de Galante (2019), os equipamentos não são isolados matematicamente durante o procedimento.

As áreas de troca térmica dos trocadores de calor do sistema, para os casos padrão e ótimo, são apresentadas na Tabela 11. A coluna mais à direita traz os valores da diferença relativa entre as áreas de troca térmica de ambos os casos.

Tabela 11 – Variação das áreas de troca térmica entre os casos padrão e ótimo via RCGA

Equipamento	Caso padrão	Caso ótimo RCGA	Δ [%]
	A_k [m ²]	A_k [m ²]	
HX1	10,00	4,02	-59,80
HX2	10,00	5,61	-43,90
HX3	10,00	39,61	296,10
HX4	10,00	17,64	176,40
HX5	10,00	16,41	164,10
Condensador	10,00	4,57	-54,30

Fonte: Autoria própria

Os resultados apresentados na Tabela 11 mostram que a área de troca térmica de alguns trocadores de calor foram reduzidas no caso ótimo. Isso indica que, para a otimização do sistema como um todo, alguns dos trocadores de calor tiveram seu desempenho, do ponto de vista da transferência de calor, reduzidos. Os trocadores HX1, HX2, e HX3 tiveram suas áreas reduzidas em mais de 40%. Na otimização pela minimização de r_k , Galante (2019) considerou somente o aumento da área de troca térmica, quando utilizou a área como parâmetro direto para essa minimização.

Na Tabela 12 são apresentados os valores de custo dos produtos, fluxo de caixa, CI_{total} e $payback$ do caso padrão e do caso otimizado pelo RCGA, considerando o cenário de RSU sem custo. A coluna mais à direita contém a diferença relativa entre os valores do caso padrão e otimizado. Para identificação, os valores são destacados em fonte azul para a redução de custo e aumento de receita, e em fonte vermelha para o aumento de custo e

redução de receita.

Tabela 12 – Variação de custos do sistema e *payback*

	Caso padrão	Caso otimizado RCGA	Δ [%]
$C_{W,turb}$ [R\$/kWh]	1,43	0,98	-31,46
C_{bio} [R\$/kg]	1,35	1,22	-10,66
C_{resf2} [R\$/m ³]	1,30	1,23	-5,69
C_{resf4} [R\$/m ³]	2,75	1,21	-56,00
$C_{subsídio}$ [R\$/kWh]	0,79	0,35	-55,70
Fluxo de caixa [R\$/h]	36,94	58,22	57,60
CI_{total} [R\$]	551.779,51	583.482,01	5,75
<i>payback</i> [ano]	3,89	2,61	32,90

Fonte: Autoria própria

Os resultados apresentados na Tabela 12 demonstram que a minimização de *payback* se deve ao aumento do fluxo de caixa, já que o CI_{total} aumenta em 5,75%. Esse aumento do capital investido é considerado positivo, já que o fluxo de caixa para essa configuração teve aumento de 57,60% e o período de *payback* foi reduzido em 32,90%.

5 CONCLUSÃO

Este trabalho propôs um Algoritmo Genético de Codificação Real como alternativa à otimização de um sistema térmico de geração de energia por incineração de resíduos sólidos. Este sistema havia sido estudado por Galante (2019), e otimizado pela análise exergoeconômica clássica. Diferentemente dessa análise exergoeconômica clássica, a utilização do Algoritmo Genético de Codificação Real permitiu que o sistema fosse otimizado integralmente e de forma simultânea, evitando o conflito da implementação de algumas configurações. O procedimento de otimização com o Algoritmo Genético de Codificação Real é baseado na seleção natural, que se inicia com soluções candidatas que ao longo de iterações, chamadas gerações, são modificadas e melhoradas por operadores que simulam reprodução, mutação e seleção, conforme observado na natureza. Esta otimização é realizada considerando o *payback*, isto é, o período de recuperação de capital investido, como função objetivo do problema de otimização do sistema térmico analisado.

As principais conclusões do presente trabalho são listadas abaixo:

- A otimização do sistema térmico pelo RCGA, considerando RSU sem custo, resultou em uma redução de período de *payback* de 32,9% em relação ao caso padrão, e de 14,71% em relação ao caso anteriormente otimizado por Galante (2019). Ao considerar-se uma fração de 3% do custo pelo serviço de incineração como custo para o RSU, a redução de *payback* foi de 35,17% em relação ao caso padrão, e de 15,05% em relação ao caso anteriormente otimizado por Galante (2019).
- Em relação à utilização da função de penalidade para lidar com soluções inviáveis, o RCGA não apresentou estagnação em ótimos locais. Em todas as execuções, independente dos pontos de partida, o algoritmo convergiu para valores muito próximos entre si. No entanto, nas primeiras gerações foi observada um maior valor de *payback* médio da população.
- A redução de *payback* se dá pelo aumento do fluxo de caixa. Em comparação do caso padrão com o caso otimizado via RCGA, verifica-se um aumento de 5,75% de capital de investimento total, e um incremento de 57,6% no fluxo de caixa. Pela Equação (16) verifica-se que o aumento de fluxo de caixa incide de forma inversamente proporcional no período de *payback*.
- Para a otimização do sistema como um todo, alguns equipamentos precisam ter seus desempenhos reduzidos. Os equipamentos HX2, condensador e HX1, por exemplo, tiveram suas eficiências exergéticas reduzidas no caso ótimo. Suas áreas de troca térmica também foram reduzidas, implicando numa perda de desempenho do ponto de vista da transferência de calor.
- No caso ótimo, o gás de combustão no ponto gás3 alcança a temperatura de 943°C,

não sendo necessária a queima de RSU com gás liquefeito de petróleo na câmara de pós combustão. Em contrapartida, a vazão mássica de resíduo sólido urbano (\dot{m}_{RSU}) tem um incremento de 33,33%, passando de 525 kg/h para 700 kg/h.

As variáveis de projeto utilizadas no caso ótimo implicam em mudanças estruturais, pois a variação das mesmas afetam as características construtivas de alguns equipamentos do sistema (i.e., área de troca térmica dos trocadores de calor). Como a implementação do caso ótimo interfere diretamente no dimensionamento dos equipamentos dos sistemas, e nas suas capacidades operacionais, torna-se interessante a utilização do procedimento de otimização já na fase de projeto, assim como recomendam Arora (2017) e Deb (2012). No processo de desenvolvimento de produtos e sistemas de engenharia, é comum selecionar um projeto que atende minimamente a alguns requisitos, escolhido entre algumas poucas opções. Nesse sentido, a otimização na fase de projeto pode auxiliar no lançamento de produtos e sistemas de engenharia mais eficientes e mais competitivos no mercado.

Pensando na utilização de resíduo sólido urbano como combustível, torna-se importante o acompanhamento dos níveis de emissões e da composição dos gases de exaustão, sendo válida a análise das implicações ambientais em se aumentar em 33,33% a vazão mássica de resíduo sólido urbano queimado no incinerador.

Quanto às dificuldades e desafios enfrentados no desenvolvimento do presente trabalho, seguem sugestões para trabalhos futuros:

- Utilização de *frameworks* para a simulação do processo de combustão (i.e., Cantera).
- Modelar sistemas de engenharia como problemas de programação não linear, desde as fases mais iniciais, com o objetivo de facilitar o processo de otimização.
- O estudo de outros algoritmos de otimização conhecidos na literatura e a sua utilização para a otimização de problemas de engenharia (i.e, Otimização por Enxame de Partículas (PSO – do inglês *Particle Swarm Optimization*), Recozimento Simulado (*Simulated Annealing*), etc.).
- A implementação de autoadaptação, como a apresentada em Deb, Sindhya e Okabe (2007), baseada na atualização do índice de distribuição de cruzamento (η_c) ao longo das gerações.
- Propor a utilização da planta integrada em grande escala, em municípios que ainda descartam seus resíduos em aterros sanitários.

REFERÊNCIAS

- ANTONIOU, A.; LU, W.-S. **Practical Optimization**. [s.l.]: Springer-Verlag New York Inc., 2007. Citado 2 vezes nas páginas 16 e 17.
- ARORA, J. **Introduction to Optimum Design**. London, UK: Academic Press is an imprint of Elsevier, 2017. ISBN 9780128008065. Citado 4 vezes nas páginas 24, 25, 26 e 79.
- BEJAN, A.; TSATSARONIS, G.; MORAN, M. J. **Thermal Design and Optimization**. [s.l.]: John Wiley & Sons, 1995. ISBN 0471584673. Citado 9 vezes nas páginas 19, 20, 21, 23, 28, 30, 31, 45 e 59.
- BORNE, P.; POPESCU, D.; PHILIP, F. G. **Optimization in Engineering Sciences**. [s.l.]: John Wiley & Sons, Ltd., 2014. Citado 8 vezes nas páginas 18, 37, 40, 42, 43, 44, 53 e 71.
- BRABAZON, A.; O'NEILL, M.; MCGARRAGHY, S. **Natural Computing Algorithms**. Springer Berlin Heidelberg, 2015. ISBN 3662436302. Disponível em: https://www.ebook.de/de/product/22351715/anthony_brabazon_michael_o_neill_sean_mcgarraghy_natural_computing_algorithms.html. Citado 7 vezes nas páginas 34, 35, 36, 40, 43, 50 e 52.
- BRAIMAKIS, K.; KARELLAS, S. Exergetic optimization of double stage organic rankine cycle (ORC). **Energy**, Elsevier BV, v. 149, p. 296–313, apr 2018. Citado na página 30.
- ÇENGEL, Y.; BOLES, M. **Termodinâmica - 7ed**. [s.l.]: Bookman Editora, 2013. ISBN 9788580552010. Citado 3 vezes nas páginas 32, 33 e 47.
- COELLO, C. A. C. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. **Computer Methods in Applied Mechanics and Engineering**, Elsevier BV, v. 191, n. 11-12, p. 1245–1287, jan 2002. Citado na página 27.
- DEAP. **Benchmarks — DEAP 1.3.1 documentation**. 2021. <https://deap.readthedocs.io/en/master/api/benchmarks.html#continuous-optimization>. (Accessed on 11/19/2021). Citado 2 vezes nas páginas 55 e 56.
- DEB, D.; DEB, K. Investigation of mutation schemes in real-parameter genetic algorithms. *In: **Swarm, Evolutionary, and Memetic Computing***. [s.l.]: Springer Berlin Heidelberg, 2012. p. 1–8. Citado 3 vezes nas páginas 43, 53 e 62.
- DEB, K. **Optimization for Engineering Design : Algorithms and Examples**. New Delhi: Prentice Hall of India, 2012. ISBN 9788120346789. Citado 5 vezes nas páginas 23, 24, 25, 26 e 79.
- DEB, K.; AGRAWAL, R. B. Simulated Binary Crossover for Continuous Search Space. 1995. Disponível em: <https://www.semanticscholar.org/paper/Simulated-Binary-Crossover-for-Continuous-Search-Deb-Agrawal/b8ee6b68520ae0291075cb1408046a7dff9dd9ad>. Acesso em: 21 de novembro de 2020. Citado 2 vezes nas páginas 43 e 52.

DEB, K.; AGRAWAL, S. A niched-penalty approach for constraint handling in genetic algorithms. *In: Artificial Neural Nets and Genetic Algorithms*. [s.l.]: Springer Vienna, 1999. p. 235–243. Citado 3 vezes nas páginas 52, 53 e 62.

DEB, K. *et al.* A fast and elitist multiobjective genetic algorithm: NSGA-II. **IEEE Transactions on Evolutionary Computation**, Institute of Electrical and Electronics Engineers (IEEE), v. 6, n. 2, p. 182–197, apr 2002. Citado 2 vezes nas páginas 53 e 62.

DEB, K.; SINDHYA, K.; OKABE, T. Self-adaptive simulated binary crossover for real-parameter optimization. *In: Proceedings of the 9th annual conference on Genetic and evolutionary computation - GECCO '07*. [s.l.]: ACM Press, 2007. Citado na página 79.

DINÇER, I.; ROSEN, M. A.; AHMADI, P. **Optimization of Energy Systems**. WILEY, 2017. ISBN 111889443X. Disponível em: https://www.ebook.de/de/product/29245495/ibrahim_dincer_marc_a_rosen_pouria_ahmadi_optimization_of_energy_systems.html. Citado na página 62.

EIBEN, A. **Introduction to evolutionary computing**. New York: Springer, 2003. ISBN 9783540401841. Citado na página 37.

EIBEN, A.; SCHIPPERS, C. On evolutionary exploration and exploitation. **Fundamenta Informaticae**, IOS Press, v. 35, n. 1–4, p. 35–50, 1998. ISSN 01692968. Citado na página 40.

ESHELMAN, L. J.; SCHAFFER, J. D. Real-coded genetic algorithms and interval-schemata. *In: Foundations of Genetic Algorithms*. [s.l.]: Elsevier, 1993. p. 187–202. Citado na página 41.

FOGEL, L.; OWENS, A.; WALSH, M. **Artificial Intelligence Through Simulated Evolution**. [s.l.]: Wiley, 1966. Citado na página 36.

FORTIN, F.-A. *et al.* DEAP: Evolutionary algorithms made easy. **Journal of Machine Learning Research**, v. 13, p. 2171–2175, jul 2012. Citado na página 54.

FREUND, R. M. Penalty and barrier methods for constrained optimization. **Massachusetts Institute of Technology, Cambridge, MA**, 2004. Citado na página 27.

FRIEDBERG, R. M. A learning machine: Part i. **IBM Journal of Research and Development**, IBM, v. 2, n. 1, p. 2–13, jan 1958. Citado na página 36.

FRIEDMAN, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. **Journal of the American Statistical Association**, Informa UK Limited, v. 32, n. 200, p. 675–701, dec 1937. Citado na página 30.

GALANTE, R. M. **Análise e Otimização Termoeconômica de Sistemas de Geração de Energia por Incineração de Resíduos com Filtro Biológico de Emissões**. Curitiba - PR, Brasil: [s.n.], 2019. Citado 28 vezes nas páginas 9, 20, 21, 22, 23, 27, 29, 30, 31, 32, 33, 34, 45, 46, 47, 48, 49, 50, 56, 57, 58, 59, 62, 63, 72, 74, 76 e 78.

GALANTE, R. M. *et al.* Clean energy from municipal solid waste (MSW). *In: ASME 2019 13th International Conference on Energy Sustainability*. [s.l.]: American Society of Mechanical Engineers, 2019. Citado na página 59.

GOLDBERG, D. **Genetic algorithms in search, optimization, and machine learning**. Reading, Mass: Addison-Wesley Publishing Company, 1989. ISBN 0201157675. Citado na página 37.

GOLDBERG, D. Real-coded Genetic Algorithms, Virtual Alphabets, and Blocking. *t*, 1991. Disponível em: <https://www.semanticscholar.org/paper/Real-coded-Genetic-Algorithms%2C-Virtual-Alphabets%2C-Goldberg/ce4e27373a77c850b04decc7382443096b8a76fb>. Acesso em: 21 de novembro de 2020. Citado na página 41.

HAUPT, R. L.; HAUPT, S. E. **Practical genetic algorithms**. New York: Wiley, 1998. ISBN 9780471188735. Citado 2 vezes nas páginas 21 e 37.

HERRERA, F.; LOZANO, M.; VERDEGAY, J. Tackling real-coded genetic algorithms: Operators and tools for behavioural analysis. **Artificial Intelligence Review**, Springer Science and Business Media LLC, v. 12, n. 4, p. 265–319, 1998. Citado 2 vezes nas páginas 41 e 42.

HOLLAND, J. **Adaptation in natural and artificial systems : an introductory analysis with applications to biology, control, and artificial intelligence**. Ann Arbor: University of Michigan Press, 1975. ISBN 9780472084609. Citado 2 vezes nas páginas 36 e 37.

INCROPERA, F. **Fundamentos de transferência de calor e de massa**. Rio de Janeiro: LTC, 2008. ISBN 8521615841. Citado na página 47.

KHAN ACADEMY. **Darwin, evolution, & natural selection (article) | Khan Academy**. 2016. Disponível em: <https://www.khanacademy.org/science/ap-biology/natural-selection/natural-selection-ap/a/darwin-evolution-natural-selection>. Acesso em: 17 de novembro de 2020. Citado 2 vezes nas páginas 34 e 35.

KRAMER, O. **Genetic Algorithm Essentials**. Springer-Verlag GmbH, 2017. ISBN 3319521551. Disponível em: https://www.ebook.de/de/product/28192209/oliver_kramer_genetic_algorithm_essentials.html. Citado 5 vezes nas páginas 16, 37, 39, 42 e 44.

LEE, H. K. H. *et al.* Optimization subject to hidden constraints via statistical emulation. **Pacific Journal of Optimization**, v. 7, n. 3, p. 467–478, 2011. Disponível em: <https://scholar.google.com/scholar?cluster=6376843206533956665>. Citado na página 62.

LUENBERGER, Y. Y. D. G. **Linear and Nonlinear Programming**. Springer International Publishing, 2016. ISBN 3319374397. Disponível em: https://www.ebook.de/de/product/27486787/david_g_luenberger_yinyu_ye_linear_and_nonlinear_programming.html. Citado na página 27.

MICHALEWICZ, Z. **Genetic Algorithms + Data Structures = Evolution Programs**. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996. ISBN 9783662033159. Citado 3 vezes nas páginas 21, 38 e 41.

MITCHELL, M. **An introduction to genetic algorithms**. Cambridge, Mass: MIT Press, 1996. ISBN 9780262631853. Citado na página 37.

MITCHELL, M.; TAYLOR, C. E. Evolutionary computation: An overview. *Annual Reviews*, v. 30, n. 1, p. 593–616, nov 1999. Citado na página 39.

MOHAMMADKHANI, F.; KHALILARYA, S.; MIRZAEI, I. **Exergoeconomic analysis and optimization of internal combustion engine cogeneration system using genetic algorithm**. 2011. Conference: 1st International Conference on Emerging Trends in Energy Conservation-ETEC p. Disponível em: https://www.researchgate.net/publication/318094769_Exergoeconomic_analysis_and_optimization_of_internal_combustion_engine_cogeneration_system_using_genetic_algorithm. Acesso em: 23 de novembro de 2020. Citado na página 29.

NAZARI, N.; HEIDARNEJAD, P.; PORKHIAL, S. Multi-objective optimization of a combined steam-organic rankine cycle based on exergy and exergoeconomic analysis for waste heat recovery application. **Energy Conversion and Management**, Elsevier BV, v. 127, p. 366–379, nov 2016. Citado na página 29.

PATEL, V. K.; SAVSANI, V. J.; TAWHID, M. A. **Thermal System Optimization**. [s.l.]: Springer International Publishing, 2019. Citado 5 vezes nas páginas 18, 20, 28, 29 e 30.

PRADO, G. P.; GALANTE, R. M. Real-coded genetic algorithm for double stage organic rankine cycle exergy optimization. **ABCM**, 2020. Citado na página 30.

RADCLIFFE, N. Equivalence Class Analysis of Genetic Algorithms. **Complex Systems**, v. 5, n. 2, Mar 2003. Disponível em: https://www.researchgate.net/publication/2848344_Equivalence_Class_Analysis_of_Genetic_Algorithms. Acesso em: 21 de novembro de 2020. Citado na página 41.

RAINVILLE, F. D. *et al.* GA Mutations. abr. 2021. Disponível em: <https://github.com/DEAP/deap/blob/master/deap/tools/mutation.py>. Citado na página 54.

RAINVILLE, F.-M. D. *et al.* DEAP. In: **Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference companion - GECCO Companion '12**. [s.l.]: ACM Press, 2012. Citado na página 55.

RAYWARD-SMITH. **Modern Heuristic Search Methods**. [s.l.]: John Wiley & Sons, 1996. ISBN 0471962805. Citado na página 17.

RECHENBERG, I. **Evolutionsstrategie; Optimierung technischer Systeme nach Prinzipien der biologischen Evolution**. Stuttgart-Bad Cannstatt: Frommann-Holzboog, 1973. ISBN 9783772803734. Citado na página 36.

SALHI, S. **Heuristic Search**. [s.l.]: Springer International Publishing, 2017. Citado na página 18.

SCHWEFEL, H.-P. Kybernetische evolution als strategie der experimentellen forschung in der stromungstechnik. **Technical University of Berlin**, 1965. Citado na página 36.

SIVANANDAM, S.; DEEPA, S. N. **Introduction to Genetic Algorithms**. Springer-Verlag GmbH, 2007. ISBN 354073189X. Disponível em: https://www.ebook.de/de/product/6693320/s_n_sivanandam_s_n_deepa_introduction_to_genetic_algorithms.html. Citado 4 vezes nas páginas 21, 41, 42 e 52.

SORSA, A.; PELTOKANGAS, R.; LEIVISKA, K. Real-coded genetic algorithms and non-linear parameter identification. In: **2008 4th International IEEE Conference Intelligent Systems**. [s.l.]: IEEE, 2008. Citado na página 42.

STORN, R.; PRICE, K. Differential Evolution: A Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous Spaces. **J. Global Optim.**, v. 23, Jan 1995. Disponível em: https://www.researchgate.net/publication/246648860_Differential_Evolution_A_Simple_and_Efficient_Adaptive_Scheme_for_Global_Optimization_Over_Continuous_Spaces. Citado na página 36.

WIRSANSKY, E. **Hands-On Genetic Algorithms with Python**. Packt Publishing, 2020. ISBN 1838557741. Disponível em: https://www.ebook.de/de/product/38585387/eyal_wirsansky_hands_on_genetic_algorithms_with_python.html. Citado na página 40.

WRIGHT, A. H. Genetic algorithms for real parameter optimization. *In: **Foundations of Genetic Algorithms***. [s.l.]: Elsevier, 1991. p. 205–218. Citado na página 41.

YANG. **Engineering Optimization**. [s.l.]: John Wiley & Sons, 2010. Citado 5 vezes nas páginas 17, 18, 21, 29 e 34.

Apêndices

**APÊNDICE A – Real-Coded Genetic Algorithm For Double Stage Organic Rankine
Cycle Exergy Optimization**

ENC-2020-0487

REAL-CODED GENETIC ALGORITHM FOR DOUBLE STAGE ORGANIC RANKINE CYCLE EXERGY OPTIMIZATION

Guilherme de Paula Prado**Renan Manozzo Galante**

Federal University of Technology – Parana (UTFPR), Câmpus Guarapuava, Avenida Professora Laura Pacheco de Bastos, 800. CEP 85053-525. Guarapuava- PR - Brasil

guilhermeprado@alunos.utfpr.edu.br

rmgalante@utfpr.edu.br

Abstract. *The organic Rankine cycles (ORC) have an important role for moderate/low heat source temperatures, being an alternative for common steam Rankine cycles. Several studies are conducted in using ORC for generate power and electrical energy from waste heat recovery. With aim to improve the ORC working fluid selection and different setups are analyzed by various authors. In this present paper, a real-coded genetic algorithm (RCGA) is presented as an alternative method to a brute force algorithm in a double stage ORC (DS ORC) optimization. The optimization is performed by minimizing the exergetic efficiency for 28 combinations of working fluids and design variables. The proposed RCGA proved to be an alternative tool for thermal design and optimization.*

Keywords: *DS ORC, real-coded genetic algorithm, exergy, optimization*

1. INTRODUCTION

1.1 Organic rankine cycle

The ORCs have an important role in situations where the temperature and/or the thermal power available from the energy source is limited and neither gas nor steam cycles are technically/economically viable (Macchi, 2017).

As in a common steam Rankine cycle (SRC), an organic Rankine cycle (ORC) thermal plant is essentially composed by four components: boiler, expander, pump and condenser. The ORC operates similarly to the SRC except by the using of an organic fluid as working fluid instead of water. In the ORC the working fluid is heated in the boiler to produce vapor which drive an expander connected to a generator, generating power by converting shaft work. The vapor leaves the expander with low pressure and is condensed to saturated liquid in the condenser. The saturated liquid pressure is increased in the pump that also conducts the working fluid to the boiler completing the cycle (Ahmadi *et al.*, 2020).

Accordingly with Hromadka and Martinek (2017), an organic working fluid which is useful for the ORCs is characterized by its lower boiling temperature if compared with water. The lower boiling temperature of the organic working fluids allows the ORCs to operate with moderate/lower heat source temperatures from 80°C to 300°C. The ORCs present several technical advantages: to be used with low temperature heat sources, minimal stress of the turbine, lower requirement of material mechanical properties of components, simple design of the turbine and no erosion of the rotor blades by moisture during the expansion.

Some applications and improvement of the ORCs are widely studied. Macchi (2017) reviewed geothermal ORC for electricity production, Oyekale *et al.* (2020) investigated the optimization of a hybrid solar-biomass ORC based on exergetic and exergoeconomic analysis. Some improvements in the ORC consider the working fluid selection: Braimakis and Karellas (2017) investigated four working fluids in a regenerative ORC thermoeconomic optimization and Thurairaja *et al.* (2019) evaluated the performance of an ORC for around a hundred working fluids.

In order to improve the ORCs, the setup is also studied. The multistage concept has been developed and examined for several authors. Dubberke *et al.* (2018) presented a testing procedure of an experimental cascade two-stage organic Rankine cycle (CORG). Braimakis and Karellas (2018) presented an optimization study of a double stage ORC (DS ORC) serially-connected by maximizing the exergy efficiency for different working fluids combinations.

In Braimakis and Karellas (2018), the exergy optimization is performed by a brute force (exhaustive search) optimization algorithm. A set of variables are assumed as decision variables and are varied from a minimum to a maximum value in a iterative loop. The optimization procedure is repeated from different values of heat source temperature ranging from 100 to 300°C in increments of 20 K and the feasibility of the values is checked.

The aim of the present paper is to present a real-coded genetic algorithm (RCGA) as an alternative method to optimize the same DS ORC examined by Braimakis and Karellas (2018) instead of a brute force algorithm. The comparison

between the values obtained with the RCGA and the brute force algorithm can validate the proposed RCGA as a method for thermal systems optimization. The optimization with the RCGA is performed considering some assumptions of the Braimakis and Karellas (2018) modeling. The modeling and optimization are described in methodology section.

It is not uncommon to use genetic algorithms to optimize thermodynamic systems. Baghernejad and Yaghoubi (2011) performed an exergoeconomic optimization of an integrated solar combined cycle system using genetic algorithm. Bian *et al.* (2014) employed genetic algorithm for parametric optimization of ORC. Hayat *et al.* (2017) used genetic algorithm for a dual-objective optimization of an ORC. The next subsection provides an overview about RCGAs.

1.2 Real-coded genetic algorithms

Kramer (2017) describes genetic algorithms (GA) as biologically-inspired methods for optimization that mimics the natural selection translating the biological concept of the evolution into algorithm recipes.

An artificial evolution process begins with a first generation formed by randomly or manually initialized solutions. Then the evolutionary cycle starts recombining two or more solutions with the crossover operator. The recombined solutions can be mutated for a mutation operator. The best solutions generated after crossover and mutation are selected (or tend to be select) for the following generation (Kramer, 2017).

Genetic algorithms were first introduced as optimization methods by Holland (1975), in United States. The first genetic algorithm has a binary representation, low probability mutation, fitness proportionate selection, with emphasis in recombination to generate new solution candidates. This GA is generally referred as simple genetic algorithm (SGA) or the "canonical GA" (Eiben, 2003).

Binary coded GAs proved inefficient when applied to high precision, multidimensional or continuous problems if compared with RCGAs. The real coding is favorable to be applied to variables in the continuous domain. In RCGAs the variables (floating-point) are the genes that form the chromosome and are modified by different genetic operators (Chambers, 2001).

Michalewicz (1996) compared GAs coded with binary and floating-point representations and concluded that the floating-point representation is faster, more consistent and provides higher precision. The floating-point representation also is described as easier for designing other operators based on problem specific knowledge as it is closer to the problem space.

Herrera *et al.* (1998) reviewed RCGAs and its features, and also revised different genetic operators and mechanisms. Sorsa *et al.* (2008) used a RCGA in parameter identification of the macroscopic chemostat model and revised crossover and mutation operators. The RCGA used in the present paper will be better explained and detailed in methodology section.

2. METHODOLOGY

The present paper aim to optimize the DS ORC studied by (Braimakis and Karellas, 2018) using a RCGA instead of a brute force (exhaustive search) optimization algorithm. In (Braimakis and Karellas, 2018) a brute force algorithm is performed to maximize the exergetic efficiency by varying a set of variables from a minimum to a maximum bound in a iterative loop. As previously seem, the optimization procedure is repeated from different values of heat source temperature ranging from 100 to 300°C in increments of 20 K and the feasibility of the values is checked. The optimization is performed for 28 combinations of working fluids. In the present paper, the optimization with a RCGA is proposed as an alternative of exergy optimization for the same DS ORC and working fluid combinations seemed in (Braimakis and Karellas, 2018) assuming the heat source also as a optimization variable. The modeling, some assumptions and optimization process is detailed in the next subsections.

2.1 Modeling

Braimakis and Karellas (2018) modeled the DS ORC as two single ORC cycles serially connected, operating at a higher and a lower temperature, respectively, as shown in Fig. 1. As in Braimakis and Karellas (2017) a heat source consisting of pressurized hot water is assumed. The high temperature (HT) and low temperature (LT) cycles have exactly the same configuration. For the system modeling, steady state operation is assumed, pressure drop and heat losses are neglected.

As in Braimakis and Karellas (2018), the objective function used for the optimization of the DS ORC is the exergy (second law) efficiency, given by Eq. (1).

$$\eta_{ex} = \frac{P_{e,net}}{\dot{E}_{hs}} \quad (1)$$

In Eq. (1), $P_{e,net}$ is the sum of the net electricity output of the HT and LT cycles, and the \dot{E}_{hs} is the exergetic rate of the heat source stream at the DS ORC inlet, given by Eq. (2) and Eq. (3), respectively:

$$P_{e,net} = P_{e,net,HT} + P_{e,net,LT} \quad (2)$$

$$\dot{E}_{hs} = \dot{m}_{hs} [(h_{hs,in} - h_{hs,ref}) - T_0 (s_{hs,in} - s_{hs,ref})] \quad (3)$$

The net electricity produced in each cycle is:

$$P_{e,net} = \dot{m}_{wf} \left[\eta_m \eta_G \eta_{is,exp} \Delta h_{exp,is} \frac{\Delta h_{pump,is}}{\eta_M \eta_{is,pump}} \right] \quad (4)$$

In Eq. (4), η_m , η_G and η_M refers to mechanical efficiency of the expander and the efficiency of the generator and pump motor, respectively. Δh_{is} is referent to the isentropic enthalpy drop in the expander and pump (Braimakis and Karellas, 2018).

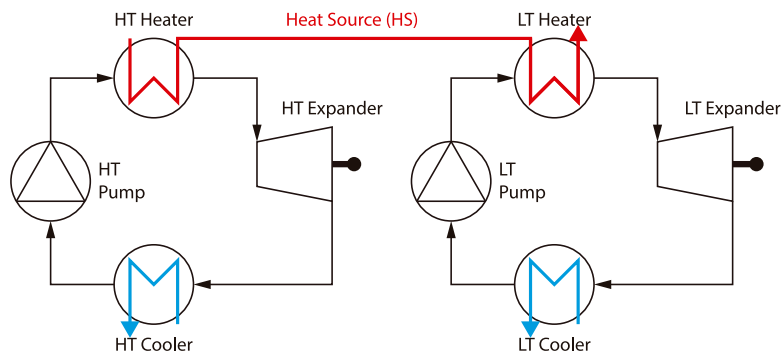


Figure 1: double stage ORC representational diagram, adapted from Braimakis and Karellas (2018).

Braimakis and Karellas (2018) investigate a selection of working fluids to operate the DS ORC. Seven working fluids are considered for HT and LT cycles. All possible permutations are examined considering that the HT working fluid must have its critical temperature greater than or equal to the LT working fluid critical temperature, resulting in 28 combinations. In the present paper the same assumptions for the working fluids selection are considered. Table 1 presents the working fluids examined and their properties.

Table 1: Working fluids properties.

Fluids	Properties	
	T_{crit} (°C)	p_{crit} (kPa)
Toluene	318.6	41.3E+2
Cyclohexane	280.5	40.8E+2
Cyclopentane	238.6	45.2E+2
Pentane	196.6	33.7E+2
Butane	152.0	38.0E+2
R1234ze	109.4	36.4E+2
R1234yf	94.70	33.8E+2

As in Braimakis and Karellas (2018), the thermophysical properties are also computed using CoolProp Library, that is developed and presented by Bell *et al.* (2014).

2.2 Optimization variables and assumptions

Braimakis and Karellas (2018) optimized the exergy efficiency of a DS ORC using a brute force algorithm with different pair of working fluids, ranging the heat source from 100 to 300 °C, using the following parameters as variables optimization: evaporation pressures in HT and LT stages, evaporator pinch point and condenser temperature in the HT stage. The main system assumptions and optimization variables are shown in Tab. 2, with optimization variables in bold.

In addition to variables shown in Tab. 2, in the present paper, the heat source temperature are also optimized between 100 and 300 °C. The calculation of the expander inlet temperature follows the methodology presented in (Braimakis and Karellas, 2017). Briefly, the expander inlet temperature is the maximum temperature in each stage, and the outlet vapour at the evaporator is superheated until the maximum specific entropy value of the saturation line is reached. The maximum temperature calculation procedure is presented in Tab. 3 depending on working fluid classification (dry or wet fluid).

Table 2: Main system assumptions and optimization variables (Braimakis and Karellas, 2018).

Efficiencies	
Eletromechanical efficiency ($\eta_m \eta_G$)	0.95
Pump motor efficiency (η_M)	0.85
Expander isentropic efficiency ($\eta_{exp, is}$)	0.75
Pump isentropic efficiency ($\eta_{pump, is}$)	0.70
Heat exchangers	
Pinch point in HT evaporator (PP_{HT})	to be optimized (5 to 40 K)
Pinch point in LT evaporator	5 K
Cooling water temperature increase in the condensers	10 K
Pressure and temperature global limits	
HT evaporation pressure (p_{HT})	to be optimized
LT evaporation pressure (p_{LT})	to be optimized
HT condensation temperature ($T_{cond, HT}$)	to be optimized (40 to 140 °C)
Minimum HT and LT condensation temperature (°C)	$\max(40, T_{sat}(p = 5 \text{ kPa}))$
Minimum evaporation pressure (HT and LT)	$1.5 p_{cond}$
Maximum evaporation pressure (HT and LT)	$\min(40E+2 \text{ kPa}, 1.4 p_{crit})$

Table 3: Maximum cycle temperature calculation procedure (Braimakis and Karellas, 2018).

Subcritical ORC	Dry fluids
	if $p < p_{sat}(s_{v, max})$, $T_{max} = T_{evap} + 5 \text{ K}$
	if $p > p_{sat}(s_{v, max})$, $T_{max} = T(p=p_{max}, s=s_{v, max})$
	Wet fluids
	$s_{v, max} = s_{sat}(T = T_{cond})$
	$T_{max} = T(p=p_{max}, s=s_{v, max})$
Supercritical ORC	$T_{max} = T(p=p_{max}, s=s_{v, max})$
All configurations	$p > p_{crit} + 2$, $p < p_{crit} - 2$

The classification of the fluids as a wet fluid or a dry fluid depends on the range of temperature operation. A same fluid can be classified and follows the Braimakis and Karellas (2017) methodology, assuming that the condensation temperature is the minimum temperature in each cycle. Basically, the maximum saturated vapor entropy $s_{v, max}$ is computed for each cycle and compared with the saturated vapor entropy at condensation temperature, $s_v(T = T_{cond})$; if is equal, the working fluid is classified as wet; otherwise, the fluid is classified as dry. Figure 2 shows the classification of butane depending on its minimum temperature of operation.

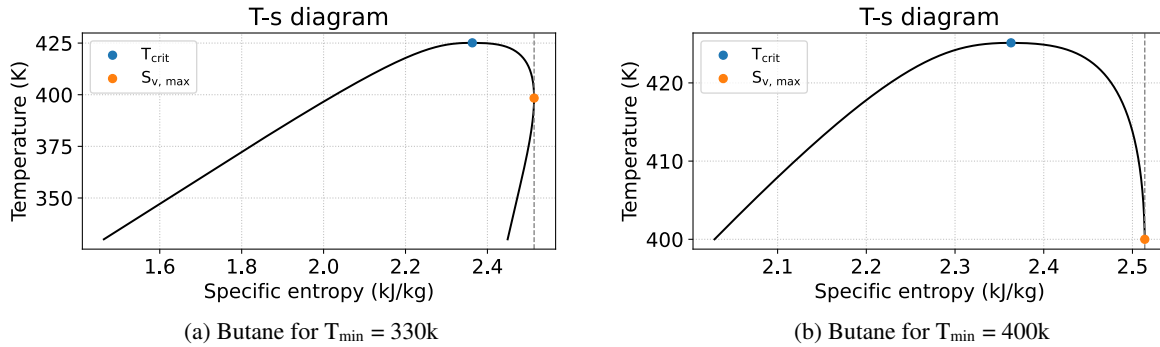


Figure 2: Butane as a dry fluid in (a) and a wet fluid in (b).

2.3 Optimization with the RCGA

The optimization procedure with the RCGA starts by creating individuals which its chromosomes contain the optimization variables as its genes. An individual chromosome is represented by a vector with floating-point numbers. In

Eq. (5), X is an individual and $x_1, x_2, x_3, \dots, x_n$ are genes represented by floating-point numbers that form the chromosome.

$$X = \langle x_1, x_2, x_3, \dots, x_n \rangle \quad (5)$$

As previously seen, in Braimakis and Karellas (2018) the optimization variables are: $PP_{HT}, p_{HT}, p_{LT}, T_{cond, HT}$. The present work also will consider $T_{hs, in}$ as a optimization variable. So, for the problem optimization, each individual has five genes and can be represented as in Eq. (6).

$$individual = \langle PP_{HT}, p_{HT}, p_{LT}, T_{cond, HT}, T_{hs, in} \rangle \quad (6)$$

For each individual, the optimization variables (genes) are randomly set in the ranges of the Tab. 4:

Table 4: Optimization variables ranges, adapted from Braimakis and Karellas (2018).

5	\leq	PP_{HT}	\leq	40 K
1.5 p_{cond}	\leq	p_{HT}	\leq	$\min(40E+2 \text{ kPa}, 1.4 p_{crit})$
1.5 p_{cond}	\leq	p_{LT}	\leq	$\min(40E+2 \text{ kPa}, 1.4 p_{crit})$
40	\leq	$T_{cond, HT}$	\leq	140 °C
100	\leq	$T_{hs, in}$	\leq	300 °C

The RCGA of the present paper uses the $(\mu + \lambda)$ evolution strategy that are better described for Beyer and Schwefel (2002). A population size (n_{pop}), generally fixed, that define the numbers of individuals is set. An initial random population with n_{pop} individuals is generated. The individuals of the population are randomly created as previously seen. A flowchart that represents the RCGA used in this paper is shown in Fig. 3. The RCGA is developed in Python3.7 with DEAP library that is presented in Fortin *et al.* (2012).

After the initial population is randomly generated, each individual is evaluated based in its fitness. The fitness is computed by using the genes of the individual as variables in DS ORC simulation and calculating Eq. (1). As the aim of optimization is to maximize the exergetic efficiency, the individuals with higher fitness (higher value of exergetic efficiency) are considered better than the others with lower fitness.

With the evaluation of the initial generation, it is possible to select parents based on individual fitness. Parents are individuals selected to reproduce with each other (pairwise) using the crossover operator, generating new individuals referred as offspring. In the RCGA of the present paper, the parent selection follows the methodology seem in (Deb *et al.*, 2002).

With a set of parents of size μ selected, the crossover operator is applied, generating a offspring of size λ . There are a lot of crossover operators for floating-point number representation, many of them is seem in Herrera *et al.* (1998) and Sorsa *et al.* (2008). In the present paper, the RCGA uses the Simulated Binary Crossover (SBX) presented in Deb and Agrawal (1994). For the SBX, a crossover probability (ρ_c) and a non-negative number (n) are set. The probability crossover, ρ_c , defines how often the crossover occurs and (n) defines how close offspring will be to parents. Generally, the case $\mu < \lambda$ is set but the cases $\mu > \lambda$ and $\mu = \lambda$ are also possible.

The mutation operator is applied after crossover. The operator used in this paper is the Polynomial Mutation Operator that is studied in Deb and ayan Deb (2014). For the polynomial mutation, a user-defined (η_m) and a mutation probability (ρ_c) is set. The mutation probability defines how often the mutation will occur and η_m defines how different an individual will be after mutation. The crossover and mutation operators described before are bounded accordingly with Tab. 4, ensuring that the offspring values after crossover and mutation do not exceed the established limits.

After crossover and mutation operations, the new population is selected. In this selection the offspring is evaluated and merged with parents forming a set (Parents + Offspring) named as mating pool, then the n_{pop} best individuals are selected from this mating pool and replace the previously population, setting a new generation. All the process of selecting parents, generating, and mutating offspring is repeated until the number of generations is smaller than a set maximum number of generations. (n_{gen}). When the maximum number of generation is reached, the process is repeated one last time, the best individual of all generations is chosen and its genes are set as optimal values for the problem. The RCGA runs repeatedly for all 28 fluids combinations and the evolution of each pair of fluid is plotted. The graphs and the results will be presented in the next section.

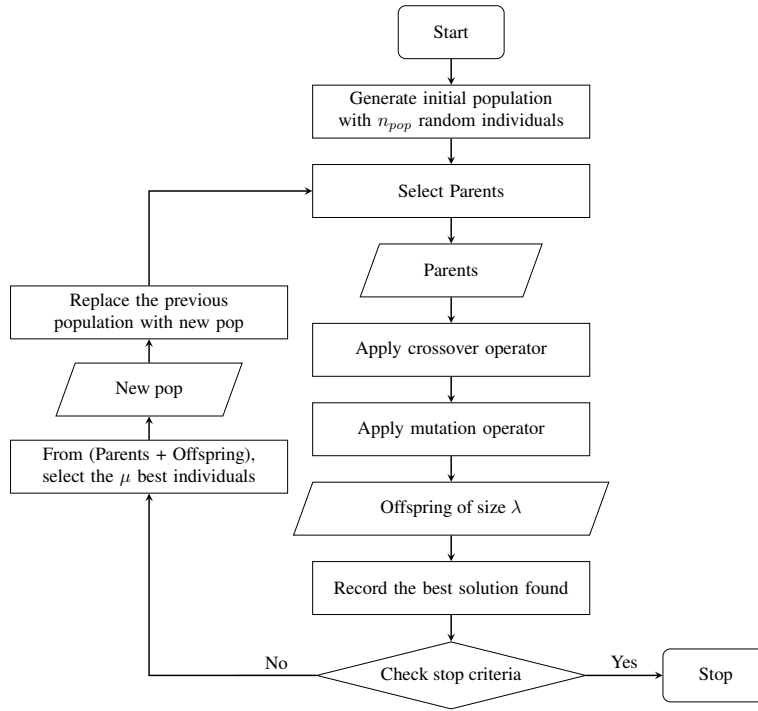


Figure 3: Flowchart of the optimization procedure with the RCGA.

3. RESULTS

As mentioned earlier, the fluid of the HT stage must have higher or equal critical temperature than the fluid of the LT stage. The RCGA is executed at once for each fluid pair, resulting in 28 runs of the RCGA, with the following parameters: $n_{pop} = 15$, $\mu = n_{pop}$, $\lambda = 30$, $\rho_c = 80\%$, $n = 0.5$, $\rho_m = 20\%$, $\eta_m = 2$, $n_{gen} = 50$.

For each run, the evolution over generations is plotted and the results are shown in Fig. 4 and 5. The $(\mu + \lambda)$ evolution strategy with the adopted parent selection apply elitism to RCGA, meaning that the best individuals of each population is preserved along the generations thus the best individual of the last generation is assumed as the optimal value.

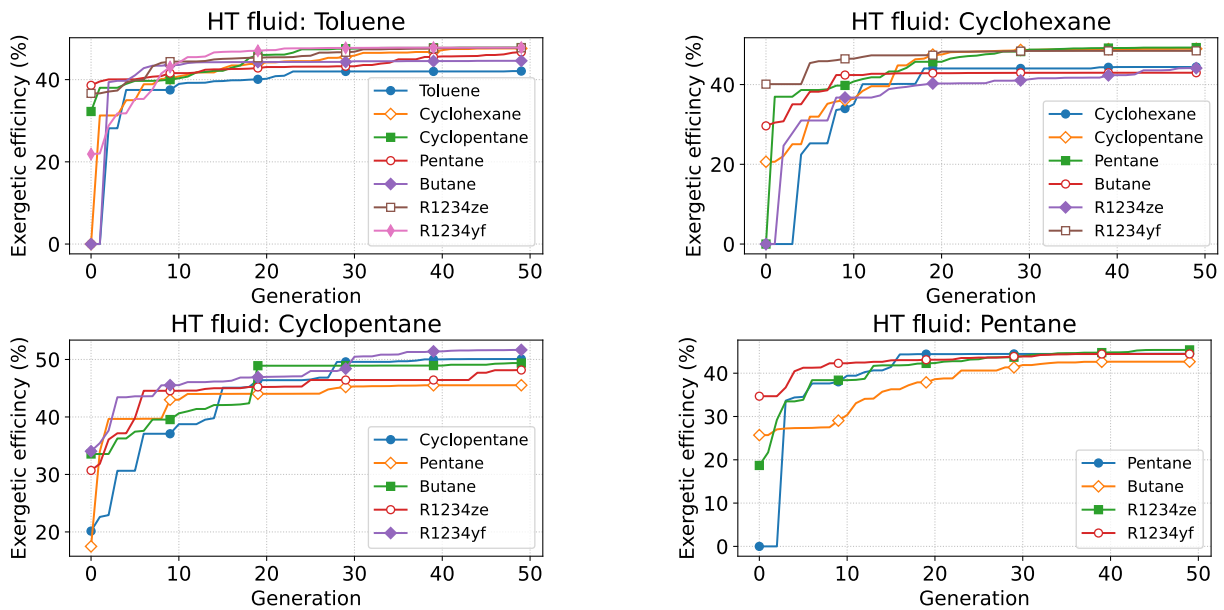


Figure 4: RCGA evolution over generations.

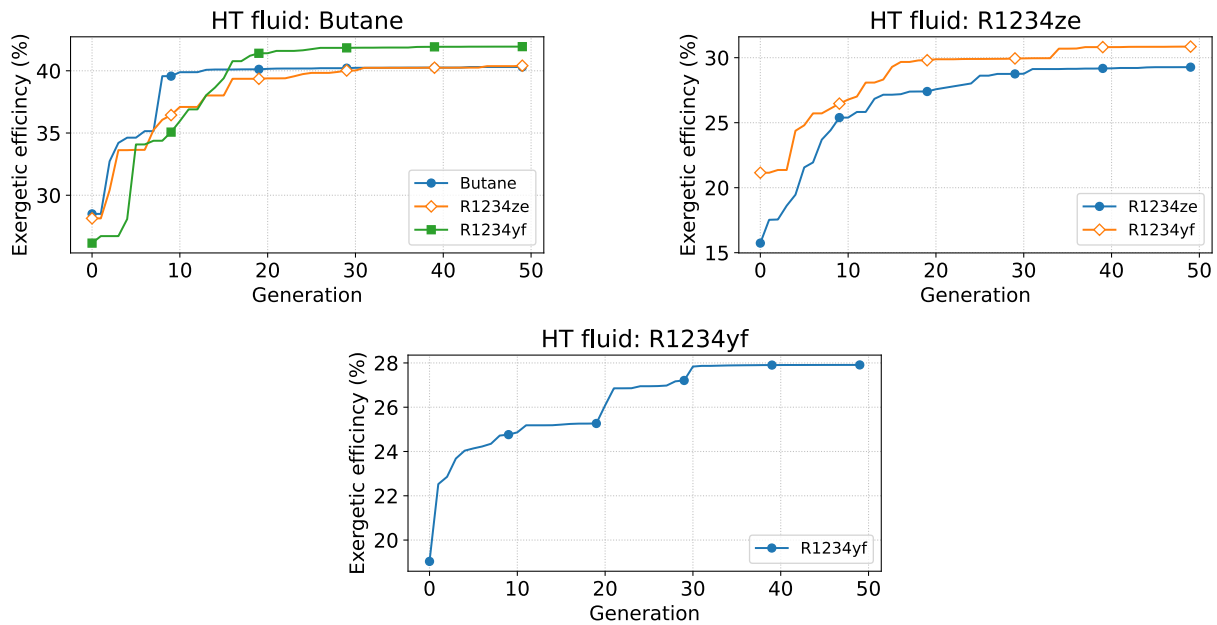


Figure 5: RCGA evolution over generations.

As expected and how it is possible to be observed in Fig. 4 and Fig. 5, the RCGA starts with a random population that generally have low value of exergetic efficiency, sometimes with all its individuals as infeasible. Along the generations, the individuals of the populations evolve by crossover and mutation operators, improving the value of exergetic efficiency until the max number of generation is reached.

As Braimakis and Karellas (2018) optimized the DS ORC for different fluid pairs ranging the heat source temperature from 100 to 300°C and its values is presented graphically, the results cannot be compared accurately, thus approximated values are assumed and presented in Tab 5. Table 5 shows a comparison between the values of optimal exergy efficiency found with RCGA of the present work and the approximated values found by the brutal force (exhaustive search) performed by Braimakis and Karellas (2018).

As shown in Tab. 5, the genetic algorithm found exergetic efficiency similarly to Braimakis and Karellas (2018), however for some pair of fluids the optimal exergetic efficiency is higher (e.g, Butane-R1234yf) and for others (majority) it is lower (e.g, Toluene-Toluene). This behavior can be explained by the non-deterministic characteristic of the GAs. As a heuristics method, GA must be executed several times to ensure an optimal value sufficiently closer of the global optimum, sometimes it is necessary to try different GA parameters to avoid that the individuals get stuck into a local optimum, and in the present work the RCGA is executed for each fluid pairs only once. Despite the cases of lower exergy efficiency found, the RCGA shows itself as a tool and an alternative for designing and optimizing thermal systems like DS ORCs. Figure 6 presents the evolution of the RCGA for the fluid pair Toluene-Toluene, showing that the optimal found can be improved if the RCGA is executed more than once. With five runs the exergetic efficiency is improved from 42.1% to 47.6%.

The Cyclopentane-R1234yf DS ORC has the higher exergetic efficiency value in Tab. 5. The optimal parameters of Cyclopentane-R1234yf DS ORC is shown in Tab. 6 and its pinch point analysis for HT and LT stages are shown in Fig. 7. By analyzing the Fig. 7 it is noted that the heat source utilization by the LT stage is insignificant and does not contribute to the exergetic efficiency. For these cases the choice for a single stage organic Rankine cycle (SS ORC) is recommended. Accordingly with Braimakis and Karellas (2018) this occurs because of the proximity between the heat source temperature at inlet and the optimal heat source temperature of the HT stage working fluid. For these cases a SS ORC configuration operating at high evaporating pressure, with high thermal efficiency is favorable.

In the present paper, the majority of working fluids optimization lead to insignificant heat source utilization by the LT stage if compared with the HT stage. This can occur because of the heat source temperature at HT stage inlet was considered as a optimization variable, leading to temperatures closer to the optimal heat source temperature of the HT stage. Braimakis and Karellas (2018) states that the DS ORC configuration operating with two working fluids is recommended for the cases which the heat source temperature is significantly lower than or between its critical temperatures. To confirm these statements a Cyclopentane-Butane DS ORC is optimized via RCGA with the heat source temperature fixed at 240°C, a value between Cyclopentane and R1234yf critical temperatures. As the critical temperature of R1234yf is lower than the minimum heat source admitted, a Cyclopentane-Butane DS ORC is optimized via RCGA with the heat source temperature fixed at 120°C, a value lower than Cyclopentane and Butane critical temperatures. The pinch point analysis of the system for the two cases is presented in Fig. 8.

Table 5: Comparison between the optimal values obtained by the RCGA of the present work and the approximated values obtained by the Brutal Force performed by Braimakis and Karellas (2018)

Fluid pair	RCGA (%)	Brutal force (%)	Fluid pair	RCGA (%)	Brutal force (%)
Toluene-Toluene	42.1	47	Cyclopentane-Pentane	45.5	49
Toluene-Cyclohexane	47.6	48	Cyclopentane-Butane	49.4	49
Toluene-Cyclopentane	47.8	48	Cyclopentane-R1234ze	48.2	49
Toluene-Pentane	46.7	48	Cyclopentane-R1234yf	51.7	49
Toluene-Butane	44.6	48	Pentane-Pentane	44.5	44
Toluene-R1234ze	47.7	48	Pentane-Butane	42.7	44
Toluene-R1234yf	47.8	48	Pentane-R1234ze	45.4	44
Cyclohexane-Cyclohexane	44.4		Pentane-R1234yf	44.4	44
Cyclohexane-Cyclopentane	49	48	Butane-Butane	40.3	40
Cyclohexane-Pentane	49.3	48	Butane-R1234ze	40.4	40
Cyclohexane-Butane	43	48	Butane-R1234yf	41.9	40
Cyclohexane-R1234ze	44.1	48	R1234ze-R1234ze	29.3	29
Cyclohexane-R1234yf	48.5	48	R1234ze-R1234yf	30.9	32
Cyclopentane-Cyclopentane	50.1	49	R1234yf-R1234yf	27.9	29

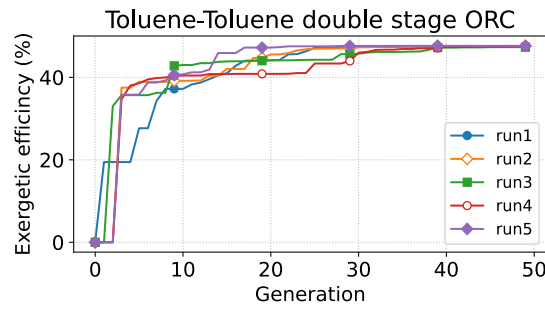
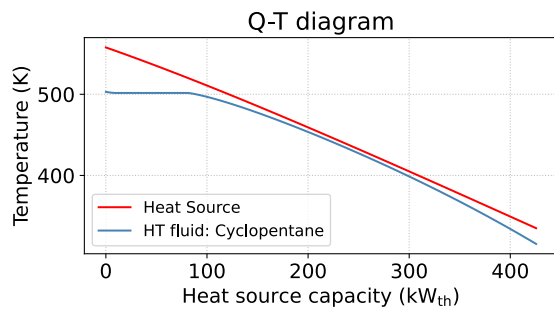


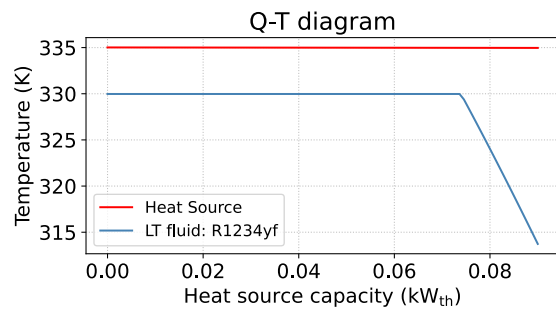
Figure 6: With aim to improve the exergetic efficiency of Toluene-Toluene DS ORC, the RCGA is executed 5 times.

Table 6: Parameters of the optimized Cyclopentane-R1234yf DS ORC.

Parameter	Value
Pinch point in HT evaporator (PP_{HT})	5.02 K
Heat source temperature at inlet ($T_{hs,in}$)	300 °C
HT condensation temperature ($T_{cond, HT}$)	40 °C
HT evaporation pressure(p_{HT})	3983 kPa
LT evaporation pressure(p_{LT})	1528 kPa



(a) HT stage heat source utilization



(b) LT stage heat source utilization

Figure 7: Heat source utilization by double stage ORC (Cyclopentane-R1234yf).

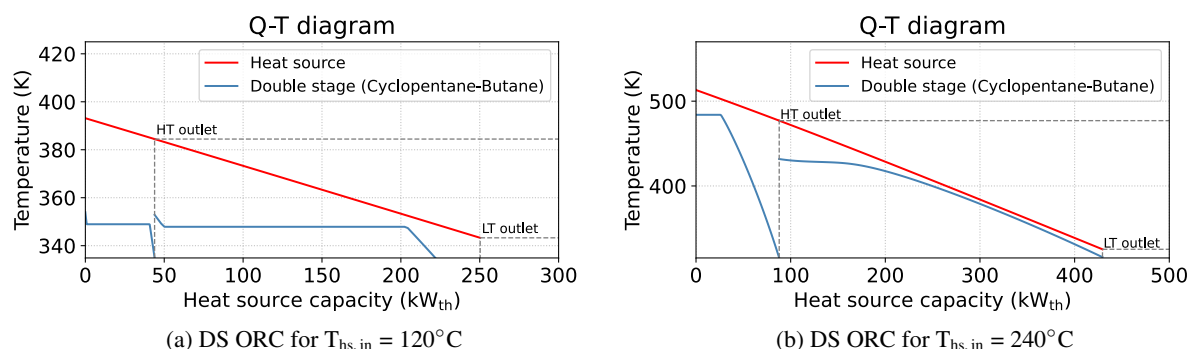


Figure 8: Heat source utilization by double stage ORC (Cyclopentane-Butane) with heat source temperatures lower than working fluids critical temperatures in (a) and between working fluids critical temperatures in (b).

4. CONCLUSIONS

The present paper proposed and performed an exergetic efficiency optimization of DS ORCs operating with 28 different pairs of working fluid using a RCGA instead of a brute force (exhaustive search) as in Braimakis and Karellas (2018). The RCGA was executed only once for each pair of fluids and found similarly results with the values found by Braimakis and Karellas (2018) that performed an exhaustive search optimization. By the heuristic and non-deterministic characteristic of the GAs, the RCGA founds lower values of exergetic efficiency for half the fluids pairs, on an average of 4.3% lower values, when compared to the exhaustive search approximate results. A possible solution for RCGA lower values is shown when running Toluene-Toluene DS ORC case five more times, given to this heuristic methodology enough opportunities to reach a good value of exergetic efficiency. The RCGA proved to be an alternative tool for thermal design and optimization. The RCGA tends to require less processing but may need to be executed several times to guarantee that a value sufficiently close to the global optimal is found. The several running dependence can be minimized with evolution strategies, parameter tuning and self-adaption which are suggested as possibilities for future works and are described in Beyer and Schwefel (2002), Chebbi and Chaouachi (2015), Saravanan *et al.* (1995), respectively.

As presented by Braimakis and Karellas (2018) the DS ORC operating with two working fluids is favorable in cases which the heat source temperature is significantly lower or between the critical temperature of these two fluids. On other hand if the heat source temperature is very close to the optimal heat source temperature of HT working fluid, a SS ORC configuration is recommend. These statements are shown by Braimakis and Karellas (2018) with an exhaustive search optimization and by the RCGA optimization of this present paper. As the ORCs are highly recommended for waste heat recovery applications and in these cases the ORC configuration must be suitable for the available heat source, the DS ORC can be an alternative, if working fluids with higher critical temperature are chosen or if the heat source temperature is between the working fluids critical temperatures. Due to the importance of ORCs in the recovery of residual heat, it would be interesting to use an exhaust gas composition as heat source instead of pressurized hot water, in order to model, design and optimize a system closer to reality.

5. ACKNOWLEDGMENTS

The authors are grateful for the support of Konstantinos Braimakis, who besides being one of the authors of the article on which the present work is based, made the implementation of the model possible.

6. REFERENCES

- Ahmadi, A., Assad, M.E.H., Jamali, D., Kumar, R., Li, Z., Salameh, T., Al-Shabi, M. and Ehyaei, M., 2020. "Applications of geothermal organic rankine cycle for electricity production". *Journal of Cleaner Production*, Vol. 274, p. 122950. doi:10.1016/j.jclepro.2020.122950.
- Baghernejad, A. and Yaghoubi, M., 2011. "Exergoeconomic analysis and optimization of an integrated solar combined cycle system (ISCCS) using genetic algorithm". *Energy Conversion and Management*, Vol. 52, No. 5, pp. 2193–2203. doi:10.1016/j.enconman.2010.12.019.
- Bell, I.H., Wronski, J., Quoilin, S. and Lemort, V., 2014. "Pure and pseudo-pure fluid thermophysical property evaluation and the open-source thermophysical property library coolprop". *Industrial & Engineering Chemistry Research*, Vol. 53, No. 6, pp. 2498–2508. doi:10.1021/ie4033999. URL <http://pubs.acs.org/doi/abs/10.1021/ie4033999>.
- Beyer, H.G. and Schwefel, H.P., 2002. *Natural Computing*, Vol. 1, No. 1, pp. 3–52. doi:10.1023/a:1015059928466.

- Bian, S., Wu, T. and Yang, J.F., 2014. "Parametric optimization of organic rankine cycle by genetic algorithm". *Applied Mechanics and Materials*, Vol. 672-674, pp. 741–745. doi:10.4028/www.scientific.net/amm.672-674.741.
- Braimakis, K. and Karellas, S., 2017. "Integrated thermoeconomic optimization of standard and regenerative ORC for different heat source types and capacities". *Energy*, Vol. 121, pp. 570–598. doi:10.1016/j.energy.2017.01.042.
- Braimakis, K. and Karellas, S., 2018. "Exergetic optimization of double stage organic rankine cycle (ORC)". *Energy*, Vol. 149, pp. 296–313. doi:10.1016/j.energy.2018.02.044.
- Chambers, L., 2001. *The practical handbook of genetic algorithms: applications*. Chapman & Hall/CRC, Boca Raton, Fla. ISBN 978-1584882404.
- Chebbi, O. and Chaouachi, J., 2015. "Effective parameter tuning for genetic algorithm to solve a real world transportation problem". In *2015 20th International Conference on Methods and Models in Automation and Robotics (MMAR)*. pp. 370–375.
- Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T., 2002. "A fast and elitist multiobjective genetic algorithm: NSGA-II". *IEEE Transactions on Evolutionary Computation*, Vol. 6, No. 2, pp. 182–197. doi:10.1109/4235.996017.
- Deb, K. and Agrawal, R.B., 1994. "Simulated binary crossover for continuous search space". Technical report.
- Deb, K. and ayan Deb, 2014. "Analysing mutation schemes for real-parameter genetic algorithms". *International Journal of Artificial Intelligence and Soft Computing*, Vol. 4, No. 1, p. 1. doi:10.1504/ijaisc.2014.059280.
- Dubberke, F.H., Linnemann, M., Abbas, W.K., Baumhögger, E., Priebe, K.P., Roedder, M., Neef, M. and Vrabec, J., 2018. "Experimental setup of a cascaded two-stage organic rankine cycle". *Applied Thermal Engineering*, Vol. 131, pp. 958–964. doi:10.1016/j.applthermaleng.2017.11.137.
- Eiben, A., 2003. *Introduction to evolutionary computing*. Springer, New York. ISBN 9783540401841.
- Fortin, F.A., De Rainville, F.M., Gardner, M.A., Parizeau, M. and Gagné, C., 2012. "DEAP: Evolutionary algorithms made easy". *Journal of Machine Learning Research*, Vol. 13, pp. 2171–2175.
- Hayat, N., Ameen, M.T., Tariq, M.K., Shah, S.N.A. and Naveed, A., 2017. "Dual-objective optimization of organic rankine cycle (ORC) systems using genetic algorithm: a comparison between basic and recuperative cycles". *Heat and Mass Transfer*, Vol. 53, No. 8, pp. 2577–2596. doi:10.1007/s00231-017-1992-9.
- Herrera, F., Lozano, M. and Verdegay, J., 1998. *Artificial Intelligence Review*, Vol. 12, No. 4, pp. 265–319. doi:10.1023/a:1006504901164.
- Holland, J., 1975. *Adaptation in natural and artificial systems : an introductory analysis with applications to biology, control, and artificial intelligence*. University of Michigan Press, Ann Arbor. ISBN 9780472084609.
- Hromadka, A. and Martinek, Z., 2017. "Overview of the organic rankine cycles and their current utilization: Verification of several current ORCs utilization by the software dymola". In *2017 18th International Scientific Conference on Electric Power Engineering (EPE)*. IEEE. doi:10.1109/epe.2017.7967272.
- Kramer, O., 2017. *Genetic algorithm essentials*. Springer, Cham, Switzerland. ISBN 9783319521565.
- Macchi, E., 2017. "Theoretical basis of the organic rankine cycle". In *Organic Rankine Cycle (ORC) Power Systems*, Elsevier, pp. 3–24. doi:10.1016/b978-0-08-100510-1.00001-6.
- Michalewicz, Z., 1996. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer Berlin Heidelberg, Berlin, Heidelberg. ISBN 9783662033159.
- Oyekale, J., Petrollese, M., Heberle, F., Brüggemann, D. and Cau, G., 2020. "Exergetic and integrated exergoeconomic assessments of a hybrid solar-biomass organic rankine cycle cogeneration plant". *Energy Conversion and Management*, Vol. 215, p. 112905. doi:10.1016/j.enconman.2020.112905.
- Saravanan, N., Fogel, D.B. and Nelson, K.M., 1995. "A comparison of methods for self-adaptation in evolutionary algorithms". *Biosystems*, Vol. 36, No. 2, pp. 157–166. doi:10.1016/0303-2647(95)01534-r.
- Sorsa, A., Peltokangas, R. and Leiviska, K., 2008. "Real-coded genetic algorithms and nonlinear parameter identification". In *2008 4th International IEEE Conference Intelligent Systems*. IEEE. doi:10.1109/is.2008.4670495.
- Thurairaja, K., Wijewardane, A., Jayasekara, S. and Ranasinghe, C., 2019. "Working fluid selection and performance evaluation of ORC". *Energy Procedia*, Vol. 156, pp. 244–248. doi:10.1016/j.egypro.2018.11.136.

7. RESPONSIBILITY NOTICE

The authors are solely responsible for the printed material included in this paper.

APÊNDICE B – RCGA Script

```

from deap import base, tools, creator
import multiprocessing
import numpy as np
import payback

# individual variables
var = { 'T_vap4': [250, 650],
        'T_vap2': [100, 200],
        'T_ar2': [125, 500],
        'T_vap1': [60, 90],
        'T_gas5': [900, 1400],
        'T_resf4': [45, 95],
        'm_gas5_h': [300, 700],
        'eta_turb': [0.55, 0.602]}

bound_low = [var[i][0] for i in var]
bound_up = [var[i][1] for i in var]

# types creation
creator.create("FitnessMin", base.Fitness, weights=(-1.0, ))
creator.create("Individual", list, fitness=creator.FitnessMin)

# individual creation function
def attr_float(bound_low, bound_up):
    return [np.random.uniform(low, up) for low, up in \
            zip(bound_low, bound_up)]

# toolbox
toolbox = base.Toolbox()
toolbox.register("individual", tools.initCycle,
creator.Individual, ind_function, n=1)
toolbox.register("population", tools.initRepeat, list,
toolbox.individual)

# genetic operators
toolbox.register("evaluate", payback.payback)

```

```

toolbox.register("mate", tools.cxSimulatedBinaryBounded,
eta=0.15, low=var_min, up=var_max)
toolbox.register("mutate", tools.mutPolynomialBounded,
eta=0.1, low=var_min, up=var_max, indpb=0.3)

def main():
    NGEN, CXPB, MUTPB = 100, 0.7, 0.3
    mu, lambda_ = 50, 100

    # computing statistics
    stats = tools.Statistics(key=lambda ind: ind.fitness.values)
    stats.register("min", np.min)
    stats.register("avg", np.mean)
    stats.register("max", np.max)

    # define hof
    hof = tools.HallOfFame(1)

    # define logbook outputs
    logbook = tools.Logbook()
    logbook.header = ['gen'] + (stats.fields if stats else [])

    # evaluate the entire population
    fitnesses = toolbox.map(toolbox.evaluate, pop)
    for ind, fit in zip(pop, fitnesses):
        ind.fitness.values = fit

    # update hof with best individual found
    hof.update(pop)

    record = stats.compile(pop) if stats is not None else {}
    logbook.record(gen=0, nevals=len(pop), **record)
    print(logbook.stream)

    # begin the generational process
    for g in range(NGEN):

        # select the next generation individuals

```

```

offspring = tools.selRandom(pop, lambda_)
# clone the selected individuals
offspring = list(map(toolbox.clone, offspring))

# apply crossover on selected individuals
for child1, child2 in zip(offspring[::2], offspring[1::2]):
    if np.random.random() < CXPB:
        toolbox.mate(child1, child2)
        del child1.fitness.values
        del child2.fitness.values

# apply mutation on the offspring
for mutant in offspring:
    if np.random.random() < MUTPB:
        toolbox.mutate(mutant)
        del mutant.fitness.value

# evaluate the individuals with an invalid fitness
invalid_ind = [ind for ind in offspring if \
                not ind.fitness.valid]
fitnesses = toolbox.map(toolbox.evaluate, invalid_ind)
for ind, fit in zip(invalid_ind, fitnesses):
    ind.fitness.values = fit

# update hof with the best of offspring
hof.update(offspring)

# select the next generation pop if mu plus lambda
pop[:] = tools.selBest(pop + offspring, mu)

# select the next generation pop if mu comma lambda
# pop[:] = tools.selBest(pop, mu)

# update the statistics with the new pop
record = stats.compile(pop) if stats is not None \
    else {}
logbook.record(gen=g, **record)
print(logbook.stream)

```

```

return pop, logbook, hof

if __name__ == "__main__":
    # enabling multiprocessing
    pool = multiprocessing.Pool()
    toolbox.register("map", pool.map)
    main()
    pool.close()

    # print the best solution found
    print(f """Best solution found: {hof.items[0]}
           with {hof.items[0].fitness.values}
           years of payback""")

    # extract statistics
    minFitnessValues, meanFitnessValues = \
    logbook.select("min", "avg")

```