Matheus José Peres Miguel

Otimização de treliças através de algoritmos evolutivos modernos

Projeto de Graduação

Projeto de Graduação apresentado ao Departamento de Engenharia Mecânica da PUC-Rio

Orientador: Helon Vicente Hultmann Ayala
Coorientador: Anderson Pereira

Rio de Janeiro

Dezembro de 2019

AGRADECIMENTOS

A Deus, por me dar vida e saúde para concluir o curso de graduação em Engenharia Mecânica.

Aos meus pais, Rosana e Marcio, pelo esforço, dedicação e tempo investidos no meu desenvolvimento acadêmico.

Ao meu irmão, Marcio, pelas orientações, conselhos e auxílios.

À minha namorada, Marcele, pela paciência e apoio.

Ao ProUni, por me permitir estudar em uma universidade cujos custos são acessíveis a uma restrita parcela da população.

Ao FESP, pelos apoios que contribuíram com a minha permanência equilibrada na universidade.

A todos os professores da PUC-Rio, pela dedicação ao ensino.

Ao meu orientador e coorientador deste trabalho, Helon e Anderson, pelas informações, sugestões e críticas construtivas.

Aos meus amigos formandos, Daniel, Raphael, Thiago, Marcos, Petrus e João, pela companhia durante o curso e pela amizade criada.

A todos os amigos e familiares que apoiaram direta e indiretamente para minha formação.

Resumo

Otimização de treliças através de algoritmos evolutivos modernos.

A otimização de treliças sob restrições de frequências naturais permite que um projetista controle as frequências selecionadas de forma a melhorar as características dinâmicas de uma estrutura. Por isso, a otimização de treliças com restrições de frequências vem recebendo muitos esforços nas últimas décadas. Por outro lado, para reduzir a massa da estrutura, são geradas mudanças bruscas nas frequências. Com isso, métodos tradicionais baseados em gradientes apresentaram dificuldades na otimização das treliças, convergindo muitas vezes para ótimos locais. Deste modo, muitos pesquisadores vêm aplicando algoritmos evolutivos nesta tarefa de otimizar treliças com restrições de frequências. Assim, este trabalho utilizou 4 algoritmos, *Genetic Algorithm* (GA), *Simulated Annealing* (SA), *Particle Swarm Optimization* (PSO) e *Heuristic Kalman algorithm* (HKA), para otimizar dois modelos de treliças que são *benchmarks* típicos na literatura, a treliça plana de 10 barras e a treliça espacial de 72 barras. A partir deste estudo, foram gerados gráficos estatísticos dos desempenhos desses algoritmos em cada problema, que também foram comparados com resultados da literatura que utilizaram outros algoritmos com este mesmo objetivo. Dos algoritmos estudados, o HKA obteve o melhor valor médio da função objetivo para ambas as treliças. Além disso, alguns algoritmos puros deste trabalho obtiveram resultados melhores que algoritmos aprimorados recentes na literatura. Com isso, foi verificado o *no free lunch* (NFL) *theorem*, que afirma que não existe algoritmo melhor que outro quando avaliados em uma média de todos os problemas possíveis.

Palavras-chave

Otimização; treliças; restrições; algoritmos evolutivos; inteligência artificial; GA; PSO; SA; HKA.

Abstract

Truss optimization through modern evolutionary algorithms.

Truss optimization under natural frequency constraints allows a designer to control the selected frequencies in order to improve the dynamic characteristics of a structure. Therefore, the truss optimization with frequency constraints has been receiving multiple efforts in the last decades. On the other hand, to decrease the truss mass, abrupt frequency changes are generated. Thereby, traditional gradient-based methods present difficulty in truss optimization, often converging to local optima. Thus, many researches have been applying evolutionary algorithms to this task of optimizing trusses with frequency constraints. Therefore, this work has used 4 algorithms, Genetic Algorithm (GA), Simulated Annealing (SA), Particle Swarm Optimization (PSO) e Heuristic Kalman algorithm (HKA), in order to optimize two truss models that are typical benchmarks in the literature, the 10-bar planar truss and the 72-bar spatial truss. From the present study, statistical graphs were generated for the performance of these algorithms in each problem, which were also compared to results in the literature that have used other algorithms for this same purpose. Among the studied algorithms, HKA has obtained the best average value for the objective function for both trusses. Furthermore, some pure algorithms from the present work have obtained better results than recent improved algorithms in the literature. Thus, the no free lunch (NFL) theorem was verified, that affirms that there is no better algorithm than other when evaluated in an average of all possible problems.

Keywords

Optimization; truss; constraints; evolutionary algorithms; artificial intelligence; GA; PSO; SA; HKA.

Sumário

Lista de figuras

Lista de tabelas

1
Introdução

A otimização de treliças sob restrições de frequências naturais permite que um projetista controle as frequências selecionadas de modo a melhorar as características dinâmicas da estrutura [16]. Assim, a otimização de treliças com restrições de frequência vem recebendo atenção considerável nas últimas décadas. Grandes esforços têm sido colocados nesse campo e conquistas marcantes vem sendo realizadas. Na maioria dos problemas de baixa frequência, a resposta de uma estrutura é primordialmente função de suas frequências naturais e geometria. Assim, a capacidade de manipular essas variáveis pode melhorar a performance da estrutura e evitar o fenômeno de ressonância [13].

Todavia, na otimização da massa da estrutura, os modos de vibração podem se alterar, por exemple, de um modo de flexão para um modo axial ou de torção, e isso pode levar a mudanças nocivas nas frequências, causando dificuldade na convergência. Métodos baseados na sensibilidade (gradiente) da frequência às variáveis de projeto obtiveram dificuldade ao lidar com vários autovalores repetidos (devido à simetria da estrutura), uma vez que o gradiente não é formalmente definido. Além disso, por causa dessa natureza baseada em gradiente, pode haver convergência para um ótimo local [14].

Assim, minimizar a massa da estrutura com várias restrições de frequências naturais adicionam complexidade ao problema e normalmente levam a divergência. Vários pesquisadores aplicaram diversos métodos com esse objetivo [4]. Com isso, algoritmos evolutivos vêm sendo utilizados massivamente para otimizar esse tipo de treliças. Dois modelos tradicionalmente utilizado nesses estudos nos últimos anos são a treliça de 10 barras e a treliça de 72 barras [1, 4–9, 13, 14]. Este trabalho tem o objetivo de dar continuidade a esse estudo, aplicando diferentes algoritmos evolutivos na otimização de treliças de 10 barras e de treliças de 72 barras.

Um projeto ótimo de estruturas de treliças sujeitas a comportamento dinâmico tem sido um campo de estudo desafiador, e uma área ativa de pesquisa. O projeto de treliça ótima sujeita a limites de frequências naturais

é uma ferramenta importante para a melhoria do comportamento dinâmico de uma treliça. Essas frequências naturais precisam ser impostas de forma a evitar ressonância mediante excitação externa. Além disso, estruturas de engenharia precisam ser o mais leves possível. Todavia, a minimização de massa conflita com os limites de frequência e aumenta a complexidade do problema [6].

Dessa maneira, um método de otimização eficiente se faz necessário para projetar treliças sujeitas a restrições de frequências naturais e, por essa razão, esforços contínuos vêm sendo realizados por pesquisadores com relação a isso [7]. Assim, muitos pesquisadores vêm aplicando em treliças algoritmos estocásticos de inteligência artificial baseados em populações evolutivas. Com isso, ao longo dos últimos anos, o problema da treliça de 10 barras vem tendo a massa reduzida a cada publicação que propõe um novo algoritmo mais efetivo [1, 4–9, 13, 14].

1.1 Revisão bibliográfica

O livro de Simon [2] apresentou abordagens de soluções de problemas de otimização. Em particular, esse trabalho abordou algoritmos evolutivos para otimização. Algoritmos inspirados na biologia e em populações de indivíduos foram apresentados como forma de resolução de problemas por inteligência computacional. Foram exibidos algoritmos clássicos como *Genetic Algorithm* (GA), *Simulated Annealing* (SA), *Particle Swarm Optimization* (PSO).

Toscano et al. [3] apresentaram o *Heuristic Kalman algorithm* (HKA) em 2009. O princípio desse algoritmo é considerar o processo de otimização explicitamente como um processo de medição projetado para produzir uma estimativa do ótimo. Em especial, esse método possui bom desempenho em problemas não convexos. Nesse trabalho foram utilizadas funções de teste para comparar o HKA com o GA, PSO e SA. Esses testes mostraram que o HKA obteve melhor desempenho na maioria das funções testadas. HKA também já foi utilizado em algoritmo de solução de problemas de otimização multiobjectivo por Ayala et al. em 2017 [11].

Em Coello [10], foi apresentada uma nova metodologia de lidar com restrições em algoritmos evolutivos. A Coevolução de populações foi utilizada para adaptar fatores de penalização nas funções de *fitness* de algoritmos genéticos aplicados em otimizações. Porém, por conta da limitação intrínseca de funções de penalização para lidar com restrições de equações, somente foram consideradas restrições em forma de inequações.

Nos últimos anos, diversos grupos de pesquisa vêm aplicando algoritmos evolutivos na otimização de estruturas de treliças com restrições de frequências [1, 4–9, 13, 14]. Em 2005, Lingyun et al. aplicaram GA para otimizar uma treliça de 10 barras [13]. Nesse trabalho, foram geradas treliças de 10 barras com massa de 542,75 kg, 1% acima da treliça de 10 barras mais leve encontrada até então por Sedagharti et al. em 2002 [15], que utilizara elementos finitos para obter uma treliça de 537,01 kg.

Depois disso, outros algoritmos estocásticos foram utilizados em busca de treliças ainda mais leves e que obedecessem às restrições de frequências naturais. Em 2011, foi aplicado PSO para obtenção de uma treliça de 10 barras que ainda era mais pesada que a treliça mais leve até então, porém somente 0,2% mais pesada, com 537,98 kg [14]. Logo após, em 2012, Miguel et al., utilizando *Harmony Search* (HS) encontraram uma treliça de 10 barras com 531,28 kg, que foi a primeira treliça obtida por algoritmo evolutivo e que era mais leve que qualquer outra treliça já encontrada anteriormente, inclusive por elementos finitos [1]. Em [1], também foi utilizado *Firefly Algorithm* (FA).

Daí em diante, outros algoritmos foram aplicados no problema de treliças com esse mesmo objetivo: minimizar a massa respeitando as restrições de frequência. Em 2016, foi utilizado *Symbiotic Organisms Search* (SOS) [4] e *Multi-Class Teaching-Learning-Based Optimization* (MC-TLBO) [5]. em 2018, foi utilizado *Improved Symbiotic Organisms* Search (ISOS) [7], *Craziness based Particle Swarm Optimization* (CRPSO) [8] e Hypotrochoid Spiral Optimization algorithm (HSPO) [9]. Em 2019, foi utilizado *Enhanced Artificial Coronary Circulation System Algorithm* (EACCS) [6]. Em [6], Kaveh et al. chegaram a uma treliça de 10 barras com 524,6001 kg. Em 2018, Lieu já havia utilizado *Adaptive Hybrid Evolutionary*

Firefly Algorithm (AHEFA) para obter a treliça de 10 barras mais leve até agora através de algoritmos estocásticos, com 524,4516 kg.

1.2 Objetivos

De forma geral, este trabalho tem o objetivo de otimizar dois *benchmarks* de treliças, a treliça plana de 10 barras e a treliça espacial de 72 barras, através dos algoritmos evolutivos modernos GA, PSO, SA e HKA. Especificamente, objetiva-se executar cada algoritmo 30 vezes na otimização de cada algoritmo, gerar gráficos de comparação do desempenho de cada algoritmo ao longo do número de avaliações da função objetivo, gerar *box plots* com os resultados estatísticos indicadores da mediana, do desvio padrão e dos *outliers* e comparar os resultados obtidos neste trabalho com outros trabalhos da literatura que utilizaram diferentes algoritmos e métodos na otimização desses *benchmarks*.

2
Metodologia para o desenvolvimento do trabalho

Neste trabalho, foram otimizadas a treliça plana de 10 barras e a treliça espacial de 72 barras, que são benchmarks típicos na literatura. Para isso, foram utilizados 4 algoritmos evolutivos diferentes: GA, PSO, SA e HKA. Além disso, foi utilizado o método de punição para tratamento das restrições dos problemas, baseado no trabalho de Coello [10].

Tanto para a treliça plana de 10 barras quanto para a treliça espacial de 72 barras, existe na literatura divergência quanto ao módulo de elasticidade E. Embora o valor original dos problemas seja $E = 10^7\ psi = 6{,}89 \times 10^{10}\ N/m^2$ [15, 16, 1], em muitos trabalhos recentes o valor considerado é $E = 6{,}98 \times 10^{10}\ N/m^2$ [4, 5, 6, 7, 8, 9, 17, 18, 19].

Por conta desta divergência, foram verificados os resultados de todas as referências, e somente os trabalhos que utilizaram, na prática, $E = 6{,}98 \times 10^{10}$ foram considerados a nível de comparação dos resultados. Esta constatação já fora feita na literatura por Hosseinzadeh [20], que gerou os resultados para os dois casos de módulo de elasticidade e comparou com os trabalhos que utilizaram os mesmos valores em cada caso.

2.1
Treliça de 10 barras

O primeiro modelo de treliça adotado no projeto é da treliça plana de 10 barras. A Tabela 1 mostra os parâmetros da treliça, que são normalmente adotados na literatura e serão utilizados para resolver este problema [1, 4–9, 13, 14].

A Figura 1 mostra o modelo da treliça com as dimensões, a identificação de cada barra e os 4 pontos de massa concentrada adicionada nos nós (m_a) circulados.

Parâmetro (unidade)	Valor
Módulo de Elasticidade $E\ (N/m^2)$	$6{,}98 \times 10^{10}$
Densidade do material $\rho\ (kg/m^3)$	2770
Massa adicionada nos nós $m_a(kg)$	454
Intervalo de tolerância da área de sessão (m^2)	$0{,}645 \times 10^{-4} \leq A \leq 50 \times 10^{-4}$
Restrições nas 3 primeiras frequências (Hz)	$\omega_1 \geq 7,\ \ \omega_2 \geq 15,\ \ \omega_3 \geq 20$

Tabela 1: Parâmetros da treliça plana de 10 barras

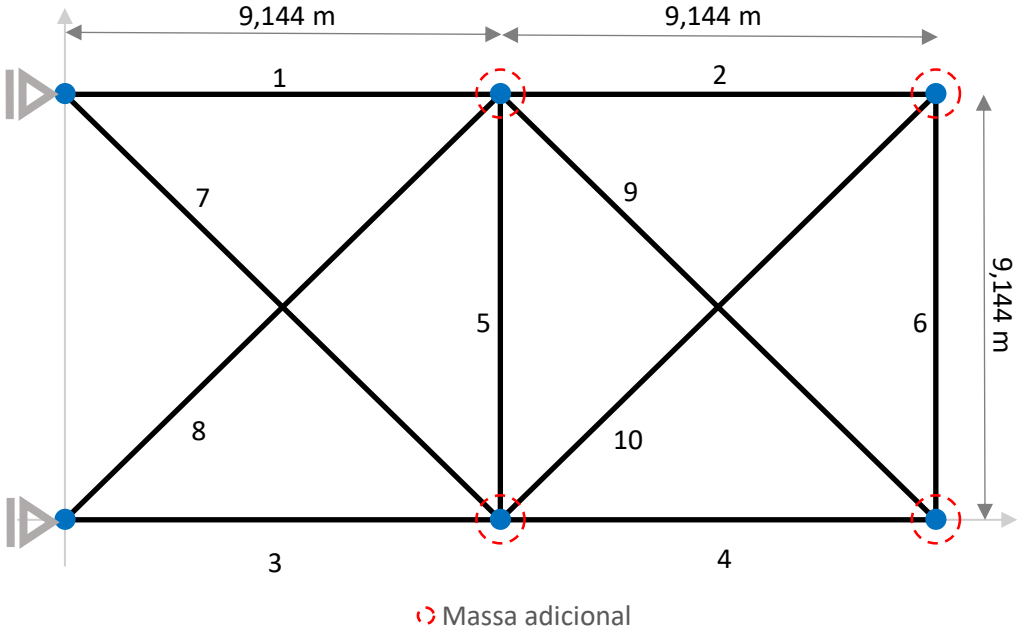

Figura 1: Modelo da treliça plana de 10 barras

O problema de minimização de peso da treliça de 10 barras sujeita a restrições de frequência pode ser formulado da seguinte forma apresentada abaixo.

$$\min_{x}\ f(\boldsymbol{x}) = \sum_{e=1}^{10} \rho l_e\, A_e(\boldsymbol{x}) \qquad (1)$$

Sujeito a:

$$
\begin{aligned}
c_1(x) &= 1 - \frac{f_1}{7} \leq 0 \\
c_2(x) &= 1 - \frac{f_2}{15} \leq 0 \\
c_3(x) &= 1 - \frac{f_3}{20} \leq 0 \\
x_l &\leq x \leq x_u
\end{aligned}
\tag{2}
$$

Com:

$$(K - \lambda_i M)v_i = 0 \tag{3}$$

onde f_1, f_2 e f_3 são as três primeiras frequências naturais da estrutura:

$$f_i = \frac{\sqrt{\lambda_i}}{2\pi}, i = 1, \dots 12 \tag{4}$$

O problema de autovalor para obtenção das frequências naturais é apresentado abaixo:

$$(K - \lambda M)\, v = 0 \tag{5}$$

onde K é a matriz de rigidez, M é a matriz de massa, λ é o autovalor associado ao autovetor v. As matrizes K e M são obtidas pela contribuição das matrizes de rigidez locais $k_{e,d}$ e $m_{e,d}$, respectivamente, onde $d = 2{,}3$ é a dimensão do problema. Assim, a matriz de rigidez local $k_{e,d}$ é definida da seguinte forma:

$$k_{e,2} = \frac{E_e A_e}{L_e} \begin{bmatrix} c_x^2 & c_x c_y & -c_x^2 & -c_x c_y \\ c_x c_y & c_y^2 & -c_x c_y & -c_y^2 \\ -c_x^2 & -c_x c_y & c_x^2 & c_x c_y \\ -c_x c_y & -c_y^2 & c_x c_y & c_y^2 \end{bmatrix} \tag{6}$$

$$k_{e,3} = \frac{E_e A_e}{L_e}\begin{bmatrix} c_x^2 & c_x c_y & c_x c_z & -c_x^2 & -c_x c_y & -c_x c_z \\ c_x c_y & c_y^2 & c_y c_z & -c_x c_y & -c_y^2 & -c_y c_z \\ c_x c_z & c_y c_z & c_z^2 & -c_x c_z & -c_y c_z & -c_z^2 \\ -c_x^2 & -c_x c_y & -c_x c_z & c_x^2 & c_x c_y & c_x c_z \\ -c_x c_y & -c_y^2 & -c_y c_z & c_x c_y & c_y^2 & c_y c_z \\ -c_x c_z & -c_y c_z & -c_z^2 & c_x c_z & c_y c_z & c_z^2 \end{bmatrix} \quad (7)$$

onde c_x, c_y e c_z são os cossenos diretores:

$$c_x = \frac{(a_{2,x} - a_{1,x})}{l_e}$$
$$c_y = \frac{(a_{2,y} - a_{1,y})}{l_e} \quad (8)$$
$$c_z = \frac{(a_{2,z} - a_{1,z})}{l_e}$$

onde $\boldsymbol{a}_1$ e $\boldsymbol{a}_2$ são as coordenadas cartesianas iniciais e finais de cada elemento. De maneira similar, a matriz de massa local $\boldsymbol{m}_{e,d}$ é definida da seguinte forma:

$$m_{e,2} = \frac{\rho A_e l_e}{6}\begin{bmatrix} 2 & 0 & 1 & 0 \\ 0 & 2 & 0 & 1 \\ 1 & 0 & 2 & 0 \\ 0 & 1 & 0 & 2 \end{bmatrix} \quad (9)$$

$$m_{e,3} = \frac{\rho A_e l_e}{6}\begin{bmatrix} 2 & 0 & 0 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 & 1 & 0 \\ 0 & 0 & 2 & 0 & 0 & 1 \\ 1 & 0 & 0 & 2 & 0 & 0 \\ 0 & 1 & 0 & 0 & 2 & 0 \\ 0 & 0 & 1 & 0 & 0 & 2 \end{bmatrix} \quad (10)$$

Esta formulação das frequências naturais também é válida para a treliça espacial de 72 barras.

2.2
Treliça de 72 barras

O segundo modelo de treliça adotado no projeto é da treliça espacial de 72 barras. A Figura 2 mostra o modelo da treliça espacial de 72 barras com as dimensões, a identificação de cada barra e os 4 pontos de massa concentrada adicionada nos nós (m_a) circulados.

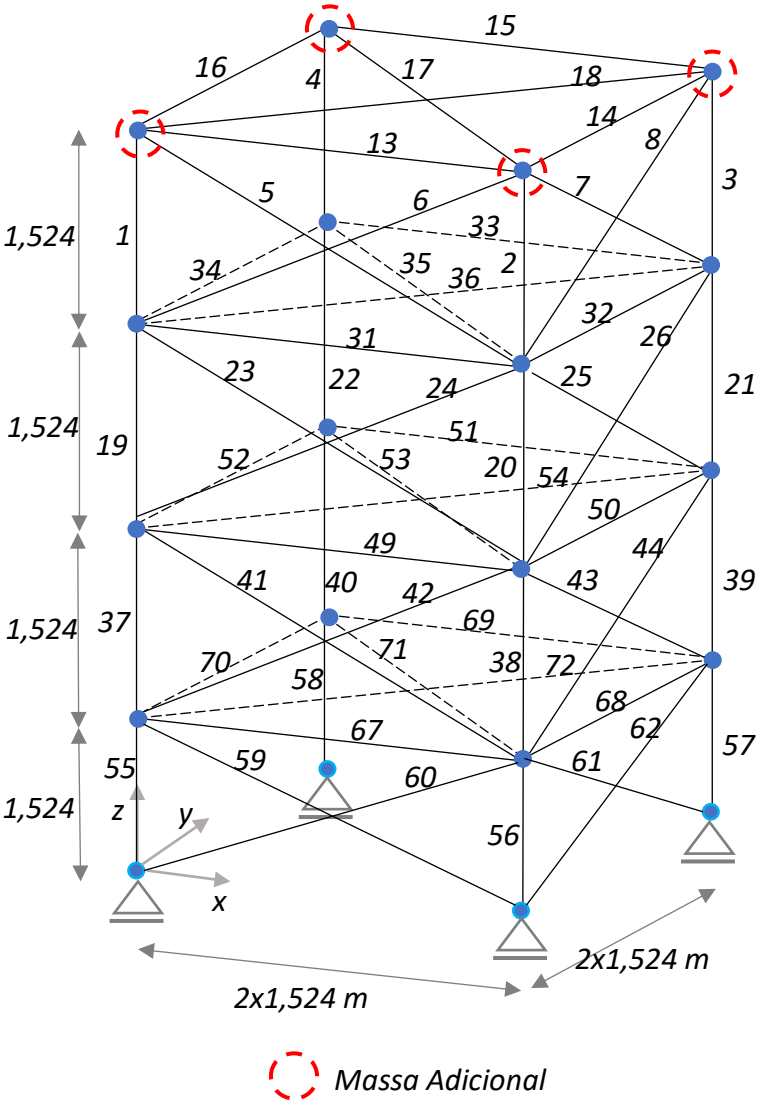

Figura 2: Modelo da treliça espacial de 72 barras

A Tabela 2 mostra os parâmetros da treliça, que são normalmente adotados na literatura e serão utilizados para resolver este problema [1, 4–9, 13, 14].

Parâmetro (unidade)	Valor
Módulo de Elasticidade $E\ (N/m^2)$	$6{,}98 \times 10^{10}$
Densidade do material $\rho\ (kg/m^3)$	2770
Massa adicionada nos nós $m_a(kg)$	2270
Intervalo de tolerância da área de sessão (m^2)	$0{,}645 \times 10^{-4} \leq A \leq 50 \times 10^{-4}$
Restrições nas 3 primeiras frequências (Hz)	$\omega_1 = 4, \omega_3 \geq 6$

Tabela 2: Parâmetros da treliça espacial de 72 barras

O problema de minimização de peso da treliça de 72 barras sujeita a restrições de frequência pode ser formulado da seguinte forma:

$$\min_{\boldsymbol{x}} \ f(\boldsymbol{x}) = \sum_{e=1}^{72} \rho l_e \, A_e(\boldsymbol{x}) \tag{11}$$

Sujeito a:

$$\begin{gathered} c_1(\boldsymbol{x}) = 1 - \frac{f_1}{4} \leq 0 \\ c_2(\boldsymbol{x}) = 1 - \frac{f_3}{6} \leq 0 \\ \boldsymbol{x}_l \leq \boldsymbol{x} \leq \boldsymbol{x}_u \end{gathered} \tag{12}$$

Com:

$$(\boldsymbol{K} - \lambda_i \boldsymbol{M})\boldsymbol{v}_i = \boldsymbol{0} \tag{13}$$

onde f_1 e f_3 são a primeira e terceira frequência natural da estrutura.

2.3 *Genetic Algorithm* (GA)

O GA é o mais antigo, mais conhecido e mais amplamente utilizado algoritmo evolutivo. GA é a simulação da seleção natural que pode resolver problemas de otimização. O GA pode resolver sistemas muito mais amplos do que otimizadores de funções [2].

As características do GA incluem um sistema baseado na biologia, com uma população de indivíduos que usualmente têm a capacidade de reproduzir. Além disso, os indivíduos têm uma vida finita, podem sofrer

mutação e sua habilidade de sobreviver está diretamente ligada à sua habilidade de reproduzir. O GA simula essas características da seleção natural. Assim, são gerados indivíduos cuja adaptação é avaliada por uma função objetivo. Os indivíduos mais bem avaliados são aqueles com mais probabilidade de reprodução e colaboram para aproximar o problema da solução ótima [2].

Com isso, os principais parâmetros utilizados na solução de um problema por GA são o número de indivíduos, a probabilidade de mutação dos indivíduos, a probabilidade de reprodução (cruzamento) dos indivíduos, e o desvio padrão para mutação. A Figura 3 mostra um pseudocódigo do GA utilizado neste trabalho.

$Pais \leftarrow \{população\ gerada\ aleatoriamente\}$
$Enquanto\ não\ \{critério\ de\ parada\}$
    $Calcule\ o\ fitness\ de\ cada\ pai\ na\ população$
    $Filhos \leftarrow \emptyset$
    $Enquanto\ |filhos| < |pais|$
        $Use\ o\ fitness\ para\ selecionar\ probabilisticamente\ um\ par\ de\ pais$
        $Acasale\ os\ pais\ para\ criar\ filhos\ c1\ e\ c2$
        $Filhos \leftarrow Filhos \cup \{c1, c2\}$
    $loop$
    $Mute\ aleatoriamente\ alguns\ filhos$
    $Pais \leftarrow Filhos$
$Próxima\ geração$

Figura 3: Pseudocódigo do GA utilizado neste trabalho (Fonte: [2]).

2.4 *Particle Swarm Optimization* (PSO)

Assim como o GA, o PSO também se inspira na inteligência de sistemas naturais. Neste caso, o PSO se baseia na inteligência coletiva de seres vivos. Um exemplo está nas formigas, que tem uma extraordinária inteligência coletiva, que não reside em cada indivíduo, mas sim no grupo como um todo. Essa característica pode ser observada em bandos de animais que evitam predadores, buscam comida e procuram viajar mais rapidamente. Nesses grupos, a colaboração permite que cada indivíduo

possa se alimentar, se reproduzir e manter sua vida de maneira mais eficiente [2].

O PSO se baseia em três principais conceitos. O primeiro deles é a inércia, que se baseia no fato de que tendemos a nos apegarmos aos velhos costumes que comprovadamente tiveram sucesso no passado. O segundo conceito é a influência da sociedade, que se baseia no fato de que nós ouvimos sobre o sucesso de outros e nós tentamos imitar suas atitudes. O terceiro e último conceito é a influência da vizinhança, que se baseia no fato de que nós costumamos aprender mais com o exemplo daqueles que estão próximos de nós [2].

Com isso, os principais parâmetros utilizados na solução de problemas através de PSO são o número de partículas, a velocidade máxima, a aceleração influenciada pelos melhores locais, a aceleração influenciada pelos melhores globais, a inércia máxima (inicial) e a inércia mínima (final). A Figura 4 mostra um pseudocódigo do GA utilizado neste trabalho.

$Inicialize\ uma\ população\ aleatória\ de\ indivíduos\ \{x_i\}, i \in [1, N]$
$Inicialize\ o\ vetor\ velocidade\ de\ cada\ individuo\ v_i, i \in [1, N]$
$Inicialize\ a\ melhor\ posição\ até\ o\ momento\ de\ cada\ indivíduo: b_i \leftarrow x_i,$
$i \in [1, N]$
$Defina\ o\ tamanho\ da\ vizinhança\ \sigma < N$
$Defina\ as\ acelerações\ de\ influência\ máximas\ \phi_{1,máx}\ e\ \phi_{2,máx}$
$Defina\ a\ velocidade\ máxima\ v_{máx}$
$Enquanto\ não\ \{critério\ de\ parada\}$
  $Para\ cada\ indivíduo\ x_i, i \in [1, N]$
    $H_i \leftarrow \{\sigma\ vizinhos\ mais\ próximos\ de\ x_i\}$
    $h_i \leftarrow mínimo_x\{f(x): x \in H_i\}$
    $Crie\ um\ vetor\ aleatório\ \phi_1\ com\ \phi_1(k) \sim U[0, \phi_{1,máx}]\ para\ k \in [1, n]$
    $Crie\ um\ vetor\ aleatório\ \phi_2\ com\ \phi_2(k) \sim U[0, \phi_{2,máx}]\ para\ k \in [1, n]$
    $v_i \leftarrow v_i + \phi_1 \circ \{b_i - x_i\} + \phi_2 \circ \{h_i - x_i\}$
    $Se\ |v_i| > v_{máx}\ então$
      $v_i \leftarrow v_i . v_{máx}/|v_i|$
    $Fim\ se$
    $x_i \leftarrow x_i + v_i$
    $b_i \leftarrow mínimo\{f(x_i), f(b_i)\}$
  $Próximo\ indivíduo$
$Próxima\ geração$

Figura 4: Pseudocódigo do PSO utilizado neste trabalho (Fonte: [2]).

2.5
Simulated Annealing (SA)

Assim como o GA e o PSO, o SA também se inspira na inteligência de sistemas da natureza. Neste caso, o SA se baseia no comportamento do resfriamento e da cristalização de substâncias químicas. Todavia, diferente dos outros algoritmos apresentados, o SA não envolve uma população de indivíduos, mas sim um único indivíduo [2].

Na natureza, um exemplo deslumbrante da habilidade da natureza em otimização são as redes cristalinas. Uma rede cristalina é um arranjo de átomos ou moléculas em um líquido ou sólido. Em altas temperaturas materiais cristalinos não demonstram muita estrutura, a temperatura alta dá ao material muita energia, que contribui para muita vibração e desordem. Entretanto, conforme a temperatura diminui, os materiais cristalinos se acomodam em estruturas mais ordenadas. Além disso, a estrutura cristalina não é sempre a mesma, pois um material que é aquecido e resfriado diversas vezes se acomoda em estados de equilíbrio diferentes cada vez, mas cada estado tende a ter baixa energia. A Figura 5 mostra uma comparação entre uma estrutura com alta entropia (esquerda) e uma estrutura com baixa entropia (direita) [2].

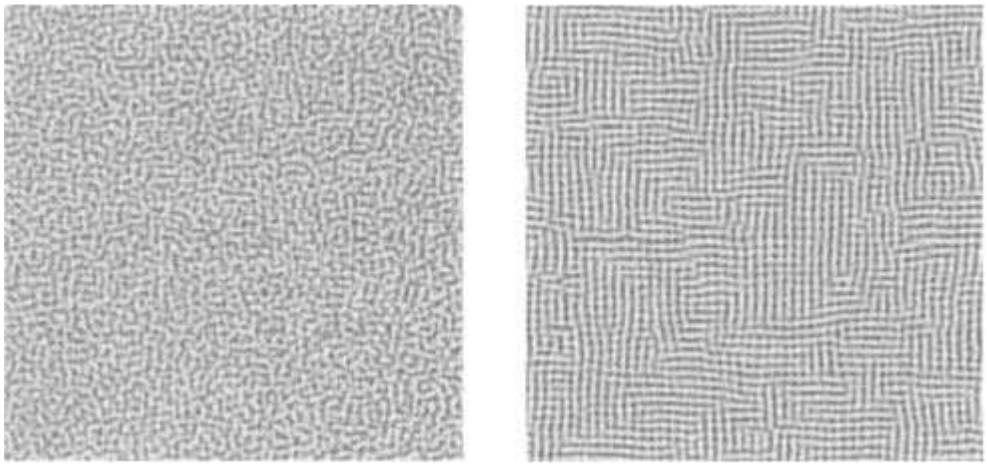

Figura 5: Visão conceitual de um processo de recozimento (Fonte: [2])

O SA se baseia na mecânica estatística, que estuda o comportamento de diversas partículas interagindo, como os átomos em um gás. De forma quantitativa, suponha que $E(s)$ seja a energia de uma configuração

especifica s. A probabilidade de o sistema de átomos estar na configuração s é dada por:

$$P(s) = \frac{\exp\left[-\frac{E(s)}{kT}\right]}{\sum_w \exp\left[-\frac{E(w)}{kT}\right]} \quad (14)$$

Onde k é a constante de Boltzmann, T é a temperatura do sistema no equilíbrio e a soma no denominador é dada por todas as w possíveis configurações [2].

Agora suponha que o sistema está em uma configuração q e escolhemos aleatoriamente uma configuração r que é candidata para a configuração do sistema no próximo passo. Se $E(r) < E(q)$, então aceitamos r como a configuração no próximo passo com probabilidade um [2].

$$P(r|q) = 1 \quad se\ E(r) < E(q) \quad (15)$$

Entretanto, se $E(r) \geq E(q)$, então movemos para r com probabilidade que é proporcional à energia relativa de q e r, ou seja, existe uma probabilidade diferente de zero de o sistema mover para a configuração com energia maior [2].

$$P(r|q) = \exp[E(q) - E(r)/(kT)] \quad se\ E(r) < E(q) \quad (16)$$

Com isso, os principais parâmetros usados na solução de problemas através de SA são o número de amostras para determinação da temperatura inicial, o coeficiente de redução da temperatura e o desvio padrão para geração de uma nova solução candidata. A Figura 6 mostra um pseudocódigo do SA utilizado neste trabalho.

$T = temperatura\ inicial > 0$
$\alpha(T) = função\ de\ refriamento: \alpha(T) \in [0, T]\ para\ todo\ T$
$Inicialize\ uma\ solução\ candidata\ x_0\ a\ partir\ de\ uma\ amostra\ inicial$
$Enquanto\ não\{critério\ de\ parada\}$
    $Gere\ uma\ solução\ candidata\ x$
    $Se\ f(x) < f(x_0)$
        $x_0 \leftarrow x$
    $Senão$
        $r \leftarrow U[0,1]$
        $Se\ r < \exp\left[\frac{f(x_0) - f(x)}{kT}\right] então$
            $x_0 \leftarrow x$
        $Fim\ se$
    $Fim\ se$
    $T \leftarrow \alpha(T)$
$Próxima\ iteração$

Figura 6: Pseudocódigo do SA utilizado neste trabalho (Fonte: [2]).

2.6 *Heuristic Kalman Algorithm* (HKA)

Diferente dos outros algoritmos, o HKA não tem uma forte inspiração em sistemas observáveis na natureza. Ele se baseia na analogia entre o procedimento de otimização e os processos de medição e estimativa da estimativa de estado [11].

O princípio de funcionamento do HKA é mostrado na Figura 7. O procedimento é iterativo, onde k é a k-ésima iteração no algoritmo. O algoritmo é baseado em 5 passos, que são resumidos abaixo [11].

- O primeiro passo é a inicialização da média m_k e do desvio padrão Σ_k no instante $k = 0$.

$$m_0 = \begin{bmatrix} \mu_1 \\ \vdots \\ \mu_n \end{bmatrix}, \mu_i = \frac{\overline{x_i} + \underline{x_i}}{2}, i = 1,2,\dots,n \tag{17}$$

$$\Sigma_0 = \begin{bmatrix} \sigma_1^2 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_n^2 \end{bmatrix}, \sigma_i = \frac{\overline{x_i} + \underline{x_i}}{6}, i = 1,2,\dots,n \tag{18}$$

onde $\overline{x_i}$ e $\underline{x_i}$ são os limites superior e inferior da i-ésima dimensão da variável de decisão x.

- O segundo passo é a geração da gaussiana de acordo com a média e o desvio padrão. Com isso são gerados N novas soluções de acordo com a distribuição normal $\mathcal{N}$.

$$x_k^i = \mathcal{N}(m_k, \Sigma_k), \forall i \tag{19}$$

- O terceiro passo é o processo de medição, onde as soluções na k-ésima iteração são ordenadas de acordo com o valor da função objetivo. A variância é então calculada como:

$$V_k = \frac{1}{N_\xi}\left[\sum_{i=1}^{N_\xi}(x_{1,k}^i - \xi_{1,k})^2 \ , \dots, \sum_{i=1}^{N_\xi}(x_{n,k}^i - \xi_{n,k})^2 \right]^T \tag{20}$$

Onde N_ξ é o número de melhores candidatos e $\xi_k = \frac{1}{N_\xi}\sum_{i=1}^{N_\xi} x_k^i$ é a medição.

- O quarto passo é calculado o estimador de Kalman.

$$L_k = \Sigma_k(\Sigma_k + diag(V_k))^{-1} \tag{21}$$

$$W_k = (vec^d((I - L_k)\Sigma_k))^{1/2} \tag{22}$$

$$a_k = \frac{\alpha.\min\left(1, \left(\frac{1}{N}\sum_{i=1}^{n}\sqrt{v_{i,k}}\right)^2\right)}{\min\left(1, \left(\frac{1}{N}\sum_{i=1}^{n}\sqrt{v_{i,k}}\right)^2\right) + \max(w_i, k)} \tag{23}$$

Onde vec^d retorna um vetor com os componentes da diagonal da matriz, $S_k = (vec^d(\Sigma_k))^{1/2}$ e I é a identidade.

- O quinto passo é a atualização da média e do desvio padrão de acordo com o estimador de Kalman calculado no passo anterior.

$$m_{k+1} = \mathrm{m}_k + \mathrm{L}_k(\xi_k - \mathrm{m}_k) \tag{24}$$

$$S_{k+1} = \mathrm{S}_k + \mathrm{a}_k(\mathrm{W}_k - \mathrm{S}_k) \tag{25}$$

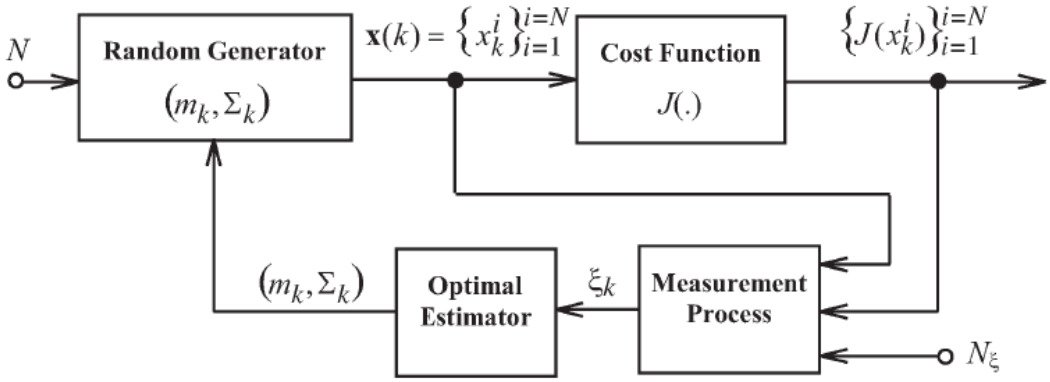

Figura 7: Princípio do HKA (Fonte: [3])

Com isso, os principais parâmetros utilizados na solução de problemas através de HKA são o número de pontos, o número de melhores candidatos e o coeficiente de desaceleração utilizado no estimador de Kalman. A Figura 8 mostra um pseudocódigo do HKA utilizado neste trabalho.

Inicialize o número de pontos N
Inicialize o número de melhores candidatos N_ξ
Inicialize o coeficiente de desaceleração α
Inicialize m_0 e Σ_0 conforme primeiro passo
Enquanto não$\{$*critério de parada*$\}$
    Para cada candidato $x_i, i \in [1, N]$
        $[Gerador\ Gaussiano]\ Calcule: x_k^i = \mathcal{N}(m_k, \Sigma_k), \forall i$
        $[Processo\ de\ medição] Calcule: V_k\ e\ \xi_k$
        $[Estimador\ de\ Kalman]\ Calcule: L_k, W_k\ e\ a_k$
        $[Atualize\ média\ e\ desvio\ padrão]\ Calcule: m_{k+1}\ e\ S_{k+1}$
    Próximo candidato
Próxima iteração

Figura 8: Pseudocódigo do HKA utilizado neste trabalho (Fonte: [12]).

2.7
Tratamento de restrições por punição

O tratamento das restrições dos problemas foi baseado no trabalho de Coello [10]. Este algoritmo de tratamento de restrições na otimização de problemas de engenharia é baseado na coevolução de duas populações: uma população de indivíduos que correspondem às treliças a serem otimizadas (P1) e outra população para evoluir os fatores de penalização responsáveis por inserir as restrições no problema de otimização (P2), conforme representado na Figura 9.

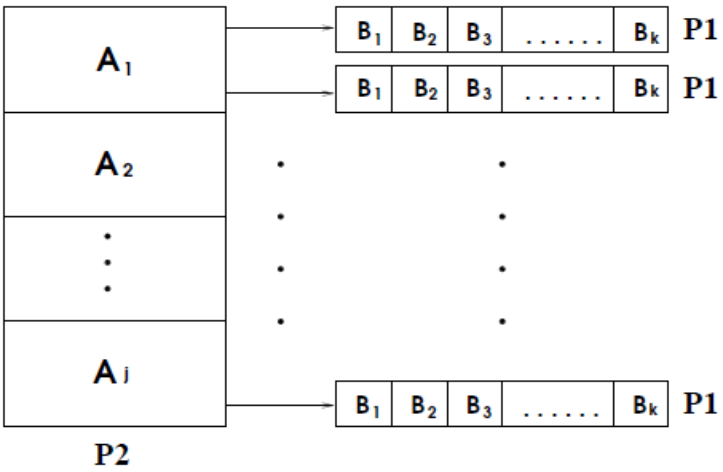

Figura 9 – Representação gráfica do método baseado em GA de lidar com restrições (Fonte: [10])

O objetivo é:

$$Minimizar\ f(X) \tag{26}$$

Sujeito às restrições:

$$g_i(X) \leq 0; \quad i = 1, \ldots, p \tag{27}$$

Para isso, a expressão usada para representar o novo *fitness* de um indivíduo φ_i, assumindo minimização da função objetivo, é dada pela função objetivo corrigida pela influência das restrições:

$$\varphi_i = f_i(X) + (S_i \times w_1 + Q_i \times w_2) \tag{28}$$

Onde $f_i(X)$ é a função objetivo avaliada no cromossomo do indivíduo i; w_1 e w_2 são os fatores de penalização; Q_i é a quantidade de restrições violadas; e S_i é a acumulação da quantidade violada em relação a cada restrição do problema.

$$S_i = \sum_{i=1}^{p} g_i(X) \quad \forall g_i(X) > 0 \tag{29}$$

Este método de lidar com restrições foi adotado devido ao seu bom desempenho com algoritmos evolutivos, uma vez que não restringe totalmente a diversidade de soluções. Além disso, este algoritmo possui alta eficiência, uma vez que apenas acrescenta punições na função objetivo.

3
Procedimento experimental

Foram aplicados os quatro algoritmos apresentados neste trabalho, GA, PSO, SA e HKA, na otimização de dois benchmarks típicos dentre os problemas de treliça, a treliça de 10 barras e a treliça de 72 barras. Em cada um dos problemas, foram executados os algoritmos evolutivos 30 vezes. Além disso, foram utilizadas sementes aleatórias padrão para garantir que fossem gerados números aleatórios homogêneos e previsíveis na execução de cada grupo de testes de cada algoritmo.

Foram estabelecidos limites para o número de avaliações da função objetivo em cada otimização. Para otimizar a treliça plana de 10 barras, foi imposto um limite de 8000 avaliações da função objetivo. Já para a treliça espacial de 72 barras, esse limite foi de 12000 avaliações da função objetivo. Esses valores foram escolhidos de forma a tornar os resultados deste trabalho comparáveis com os resultados da literatura com relação ao número de avaliações da função objetivo [6] e [17].

Para o ajuste dos parâmetros, foi utilizado o método apresentado por Simon [2]. Inicialmente foram ajustados os coeficientes de penalização w_1 e w_2 variando w_1 com w_2 fixo e depois variando w_2 com w_1 fixo.

Para o GA, primeiro, selecionamos o tamanho da população e o número de indivíduos para avaliação; segundo, ajustamos a probabilidade de mutação e o seu desvio padrão; terceiro ajustamos a probabilidade de cruzamento, quando aplicável.

Para o PSO, primeiro selecionamos o número de partículas, depois foram ajustados a velocidade máxima, a aceleração para influência do melhor local, a aceleração para influência do melhor global, a inércia máxima e a inércia mínima, nesta ordem.

Para o SA, foi selecionado o número de amostras para a determinação da temperatura inicial, depois foram ajustados o coeficiente de redução de temperatura e o desvio padrão, nesta ordem.

Para o HKA, foi selecionado o número de pontos, e em seguida foram ajustados o número de melhores candidatos e o coeficiente de desaceleração, nesta ordem, conforme [12].

Esses parâmetros são apresentados da Tabela 3 até a Tabela 10.

Parâmetros do GA – 10 barras	Valor
Coeficiente de penalização para número de restrições violadas, w_2	200
Coeficiente de penalização para soma das quantidades violadas, w_1	200
Número de indivíduos	40
Número de gerações	200
Número de indivíduos para seleção por torneio	15
Probabilidade de cruzamento (*crossover*)	0,9
Probabilidade de mutação	0,95
Desvio padrão para mutação	0,016

Tabela 3: Parâmetros utilizados no GA para o problema da treliça de 10 barras

Parâmetros do GA – 72 barras	Valor
Coeficiente de penalização para número de restrições violadas, w_2	200
Coeficiente de penalização para soma das quantidades violadas, w_1	200
Número de indivíduos	40
Número de gerações	300
Número de indivíduos para seleção por torneio	15
Probabilidade de cruzamento (*crossover*)	0,9
Probabilidade de mutação	0,95
Desvio padrão para mutação	0,04

Tabela 4: Parâmetros utilizados no GA para o problema da treliça de 72 barras

Parâmetros do PSO – 10 barras	Valor
Coeficiente de penalização para número de restrições violadas, w_2	200
Coeficiente de penalização para soma das quantidades violadas, w_1	200
Número de partículas	50
Número máximo de iterações	160
Velocidade máxima	8
Aceleração para influência do melhor local	2
Aceleração para influência do melhor global	2
Inércia máxima (inicial)	0,6
Inércia mínima (final)	0,3

Tabela 5: Parâmetros utilizados no PSO para o problema da treliça de 10 barras

Parâmetros do PSO – 72 barras	Valor
Coeficiente de penalização para número de restrições violadas, w_2	250
Coeficiente de penalização para soma das quantidades violadas, w_1	250
Número de partículas	50
Número máximo de iterações	240
Velocidade máxima	12
Aceleração para influência do melhor local	2
Aceleração para influência do melhor global	2
Inércia máxima (inicial)	0,7
Inércia mínima (final)	0,2

Tabela 6: Parâmetros utilizados no PSO para o problema da treliça de 72 barras

Parâmetros do SA – 10 barras	Valor
Coeficiente de penalização para número de restrições violadas, w_2	300
Coeficiente de penalização para soma das quantidades violadas, w_1	300
Número de amostras para determinação da temperatura inicial	1000
Coeficiente de redução da temperatura	0,8
desvio padrão para a geração de uma nova solução candidata	0,01
Número de iterações	7000

Tabela 7: Parâmetros utilizados no SA para o problema da treliça de 10 barras

Parâmetros do SA – 72 barras	Valor
Coeficiente de penalização para número de restrições violadas, w_2	300
Coeficiente de penalização para soma das quantidades violadas, w_1	300
Número de amostras para determinação da temperatura inicial	1000
Coeficiente de redução da temperatura	0,7
desvio padrão para a geração de uma nova solução candidata	0,014
Número de iterações	1100

Tabela 8: Parâmetros utilizados no SA para o problema da treliça de 72 barras

Parâmetros do HKA – 10 barras	Valor
Coeficiente de penalização para número de restrições violadas, w_2	200
Coeficiente de penalização para soma das quantidades violadas, w_1	200
Número de pontos	100
Número de melhores candidatos	5
Coeficiente de desaceleração	0,5

Tabela 9: Parâmetros utilizados no HKA para o problema da treliça de 10 barras

Parâmetros do HKA – 72 barras	Valor
Coeficiente de penalização para número de restrições violadas, w_2	200
Coeficiente de penalização para soma das quantidades violadas, w_1	200
Número de pontos	100
Número de melhores candidatos	5
Coeficiente de desaceleração	0,3

Tabela 10: Parâmetros utilizados no HKA para o problema da treliça de 72 barras

Com os parâmetros ajustados, foram executados os testes 30 vezes para cada algoritmo em cada um dos problemas. Este número de execuções foi utilizado para garantir que cada algoritmo fosse submetido a diferentes cenários, gerando valores de média e de desvio padrão consistentes. Além disso, foram utilizadas sementes aleatórias do MATLAB para garantir valores aleatórios controlados entre cada algoritmo diferente. Com isso, foram gerados gráficos mostrando o desempenho de cada algoritmo para um dado número de avaliações da função objetivo e as estatísticas das saídas dos testes.

4
Resultados

A partir do procedimento experimental apresentado, foram gerados os resultados de desempenho e estatística de cada algoritmo. A Tabela 11 mostra as áreas de seção das melhores treliças planas de 10 barras otimizadas por cada algoritmo. Além disso, são exibidos a melhor massa, as frequências, o número de avaliações da função objetivo, a massa média e o desvio padrão.

Variáveis de projeto	Kaveh (2012)	Ho-Huu (2016)	Lieu (2018)	Tejani (2018)	Kaveh (2019)	Este trabalho (2019)			
A_i (cm^2)	CSS–BBBC	IDE	AHEFA	ISOS	EACCSA	GA	PSO	SA	HKA
A_1	35,2740	35,0606	35,1714	35,2654	35,0354	35,8803	35,1242	36,7443	36,0332
A_2	15,4630	14,6851	14,7203	14,6803	15,0132	14,9778	13,5765	12,6058	14,4582
A_3	32,1100	35,0688	35,1074	34,4273	35,4505	34,4350	34,5766	34,2700	34,2692
A_4	14,0650	14,8095	14,6986	14,9605	15,0603	14,3828	15,4560	16,1964	14,7792
A_5	0,6450	0,6451	0,6451	0,6450	0,6475	1,6247	0,7116	1,8859	0,6458
A_6	4,8800	4,5578	4,5593	4,5927	4,5224	4,5625	4,6568	5,0303	4,5722
A_7	24,0460	23,5271	23,7330	23,3417	23,4493	24,0008	21,9693	21,1412	23,4786
A_8	24,3400	23,7998	23,6795	23,8236	23,4182	23,9575	25,7323	27,0703	24,1181
A_9	13,3430	12,5038	12,3987	12,8497	12,1051	12,3495	12,7456	13,6109	12,4956
A_{10}	13,5430	12,4599	12,4231	12,5321	12,7158	12,2328	12,6524	10,8362	12,2872
Melhor massa (kg)	529,0900	524,4627	524,4516	524,7341	524,6001	527,9830	525,5239	530,6081	524,6064
ω_1 (Hz)	7,0000	7,0000	7,0000	7,0001	7,0000	7,0011	7,0001	7,0017	7,0001
ω_2 (Hz)	16,1190	16,1853	16,1920	16,1703	16,2094	17,2119	16,1974	17,3015	16,1912
ω_3 (Hz)	20,0750	20,0000	20,0000	20,0024	20,0000	20,0152	20,0002	20,0003	20,0008
Avaliações da F.O.	4000	6260	5860	4000	8000	8000	8000	8000	8080
Massa média (kg)	–	525,6162	525,1623	530,0286	525,0068	535,2290	539,1398	539,7227	528,0036
Desvio Padrão (kg)	–	2,3041	1,9155	3,4763	0,3030	3,5067	12,0500	12,4153	3,4281

Tabela 11: Resultados da literatura e deste trabalho na otimização da treliça de 10 barras

De acordo com a Tabela 11, o AHEFA [18] obteve a melhor massa, 524,4516 kg. Neste trabalho, o HKA foi o algoritmo que mais se aproximou neste parâmetro, com 524,6064 kg.

A Figura 10 mostra a evolução do desempenho de cada algoritmo na otimização da treliça plana de 10 barras ao longo da quantidade de

avaliações da função objetivo. Para gerar cada curva, foram calculados, para cada número de avaliações da função objetivo, o valor médio da função objetivo avaliada no melhor indivíduo até o momento em cada uma das 30 execuções, além do desvio padrão entre os valores da função objetivo de cada execução.

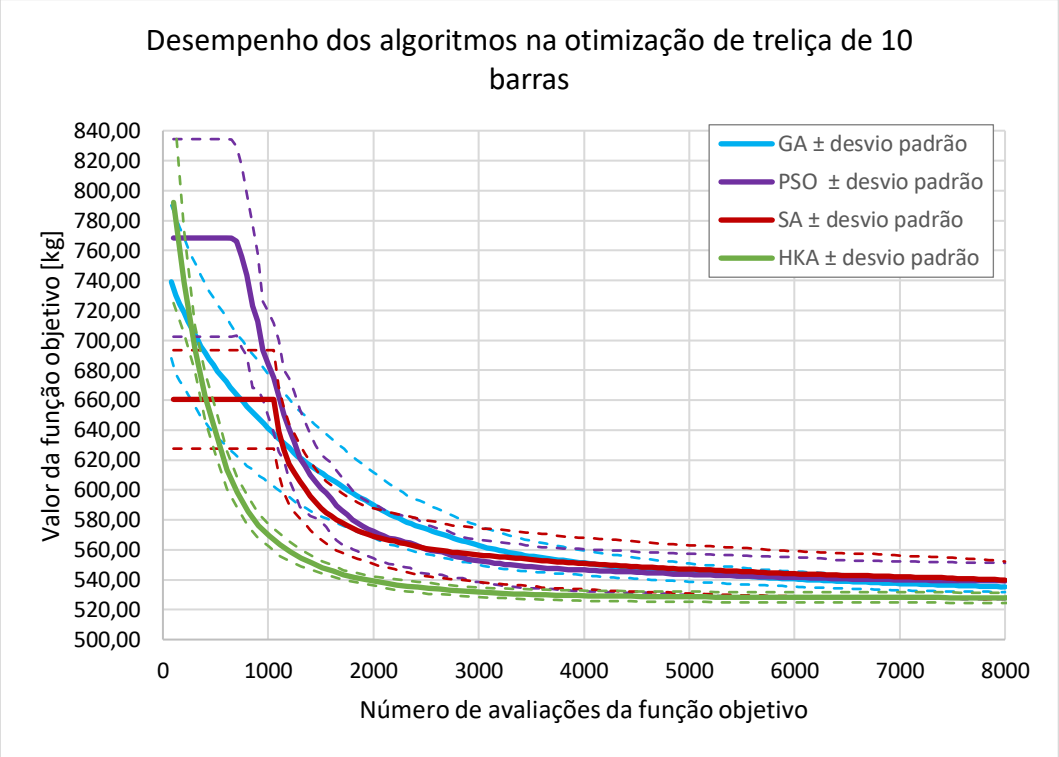

Figura 10: Desempenho do GA, PSO, SA e HKA na otimização da treliça de 10 barras

De acordo com a Figura 10, o HKA foi o algoritmo que obteve a convergência mais rápida e também o valor médio mínimo.

Nas primeiras iterações, não houve melhora no SA e no PSO. Para o SA, esse comportamento ocorre devido ao fato de este algoritmo ser baseado em um único indivíduo. Assim, se faz necessária uma amostra inicial, de 1000 treliças neste caso, para determinação da temperatura inicial. Já para o PSO, não houve melhora nas primeiras iterações devido à relação entre a diversidade da população inicial e o valor máximo de velocidade tolerado, que neste caso é constante ao longo das iterações.

A Figura 11 apresenta o *box plot* com as estatísticas dos valores finais da função objetivo em cada um dos quatro algoritmos. Nela, a linha

vermelha representa a mediana, os limites inferior e superior da caixa azul representam o 25º quartil e o 75º percentil, respectivamente, os limites superior e inferior tracejados representam os valores mais extremos desconsiderando os outliers, que são as cruzes vermelhas.

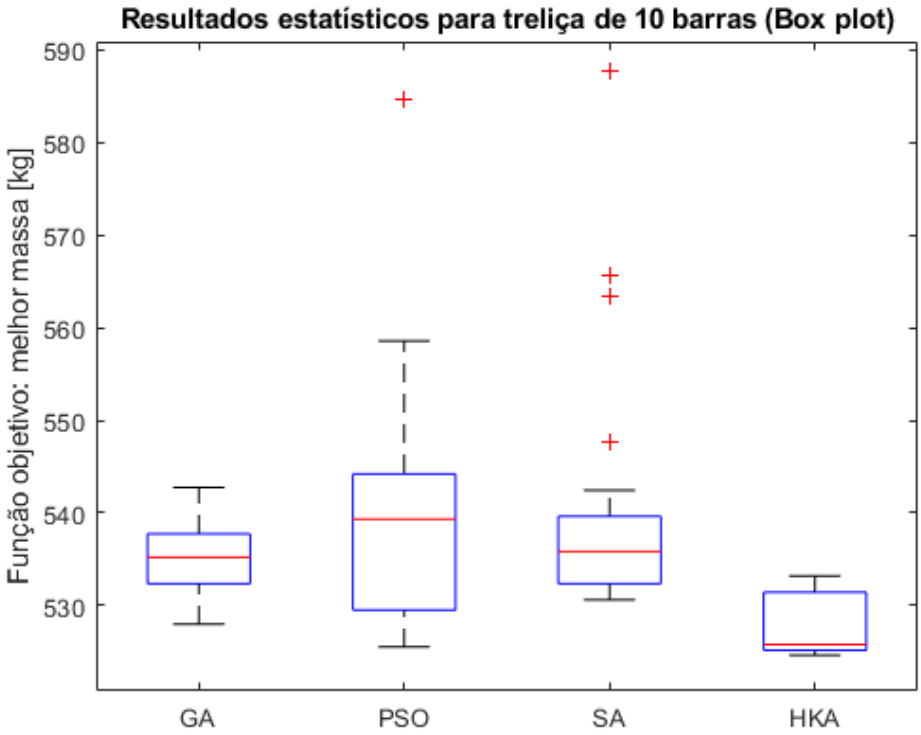

Figura 11: Box plot estatístico do GA, PSO, SA e HKA na otimização da treliça de 10 barras

De acordo com o *box plot* da treliça plana de 10 barras, o HKA obteve a melhor mediana, porém o GA obteve a menor faixa entre o 25º e o 75º. Neste caso, o PSO obteve a pior (maior) mediana e o maior desvio padrão entre os valores finais da função objetivo.

Para uma análise qualitativa dos resultados, é possível esboçar a treliça ótima com os diâmetros das barras em escala. A Figura 12 mostra a treliça plana de 10 barras otimizada através do HKA. Pelos resultados obtidos dos outros algoritmos, podemos observar que todos apresentam uma mesma ordem de grandeza para a espessura das barras, o que leva a uma treliça com as mesmas características quando realizada uma análise qualitativa.

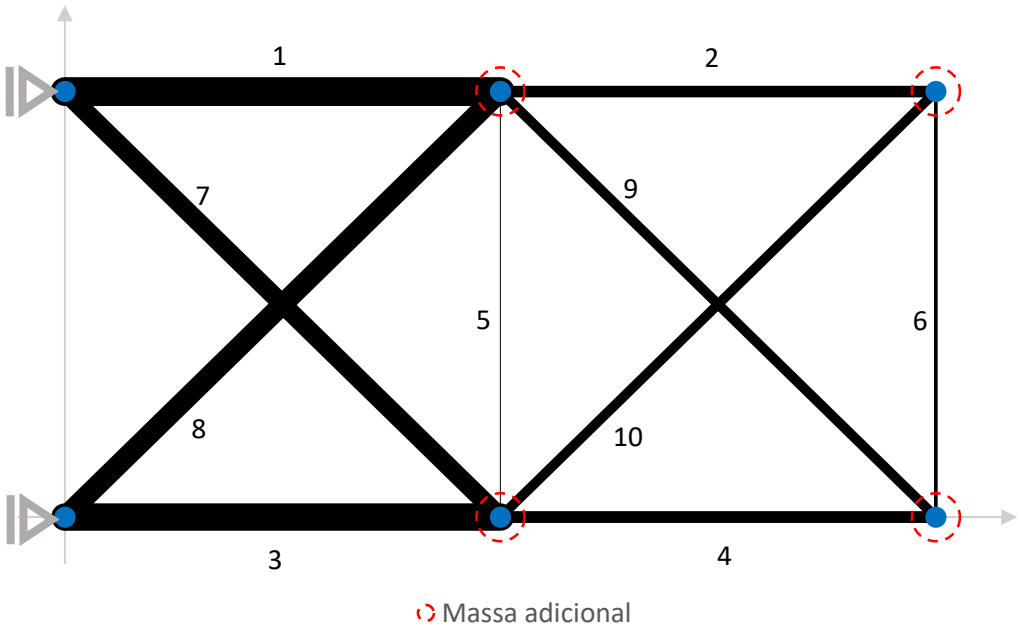

Figura 12: Treliça plana de 10 barras em escala da solução ótima obtida por HKA

De acordo com o esboço em escala da treliça de 10 barras, existe uma grande diferença proporcional entre as treliças, por exemplo a treliça número 1 possui um diâmetro mais de 7 vezes maior que a treliça número 5. Outro observação qualitativa que podemos notar da Figura 12 é que existe forte simetria entre a metade inferior e a metade superior da treliça.

A Tabela 12 mostra as áreas de seção das melhores treliças planas de 72 barras otimizadas por cada algoritmo. Além disso, são exibidos a melhor massa, as frequências, o número de avaliações da função objetivo, a massa média e o desvio padrão.

Assim como na treliça plana de 10 barras, o AHEFA [18] também obteve a melhor massa mínima, igual a 324,2376 kg, para a treliça espacial de 72 barras. Neste trabalho, o melhor resultado obtido foi do PSO, igual a 324,3094 kg. Embora a melhor massa tenha sido obtida através do PSO, a melhor média das massas foi obtida pelo HKA.

A Figura 13 mostra a evolução do desempenho de cada algoritmo na otimização da treliça espacial de 72 barras ao longo da quantidade de avaliações da função objetivo. Para gerar cada curva, foram calculados, para cada número de avaliações da função objetivo, o valor médio da função objetivo avaliada no melhor indivíduo até o momento em cada uma

das 30 execuções, além do desvio padrão entre os valores da função objetivo de cada execução.

Variáveis de projeto	Kaveh (2012)	Ho-Huu (2016)	Lieu (2018)	Tejani (2018)	Kaveh (2019)	Este trabalho (2019)			
A_i (cm^2)	CSS–BBBC	IDE	AHEFA	ISOS	ACCS	GA	PSO	SA	HKA
A_1 - A_4	2,8540	3,5863	3,5612	3,3563	4,1886	3,8950	3,2791	3,5604	3,5677
A_5 - A_{12}	8,3010	7,8278	7,8736	7,8726	7,9958	8,3975	7,8017	7,8469	7,7499
A_{13} - A_{16}	0,6450	0,6450	0,6450	0,6450	0,6813	0,7241	0,6510	1,5449	0,6503
A_{17} - A_{18}	0,6450	0,6450	0,6451	0,6450	0,7430	0,9001	0,6457	2,3071	0,6588
A_{19} - A_{22}	8,2020	8,1052	7,9710	8,5798	6,4211	7,8710	8,2113	7,8725	8,2160
A_{23} - A_{30}	7,0430	7,8788	7,8928	7,6566	7,6720	7,7870	7,8879	7,8155	7,8521
A_{31} - A_{34}	0,6450	0,6451	0,6450	0,7417	0,7656	0,6405	0,6454	0,7186	0,6763
A_{35} - A_{36}	0,6450	0,6450	0,6451	0,6450	0,7014	0,8187	0,6450	0,7593	0,6619
A_{37} - A_{40}	16,3280	12,5157	12,5404	13,0864	14,0034	11,5950	12,8016	12,9832	12,9683
A_{41} - A_{48}	8,2990	8,0102	7,9639	8,0764	8,1338	7,6497	8,0738	8,2392	8,0617
A_{49} - A_{52}	0,6450	0,6450	0,6459	0,6450	0,6612	0,9046	0,6450	1,7674	0,6474
A_{53} - A_{54}	0,6450	0,6452	0,6462	0,6937	0,6810	1,4619	0,6450	1,0923	0,6607
A_{55} - A_{58}	15,0480	16,9997	17,1323	16,2517	17,1666	18,1013	16,9347	16,7245	16,5012
A_{59} - A_{66}	8,2680	8,0362	8,0216	8,1703	7,9807	7,9874	7,9911	7,9161	8,1192
A_{67} - A_{70}	0,6450	0,6451	0,6450	0,6450	0,7097	0,6225	0,6450	1,7565	0,6553
A_{71} - A_{72}	0,6450	0,6453	0,6451	0,6450	0,6532	1,0015	0,6450	1,4821	0,6546
Melhor massa (kg)	327,5070	324,2441	324,2376	325,0682	325,1956	330,0699	324,3094	342,7601	324,8469
ω_1 (Hz)	4,0000	4,0000	4,0000	4,0000	4,0000	4,0016	4,0002	4,0027	4,0014
ω_3 (Hz)	6,0040	6,0000	6,0000	6,0008	6,0112	6,0016	6,0001	6,0037	6,0023
Avaliações da F.O.	4000	11620	8860	4000	12000	12000	12000	12000	12019
Massa média (kg)	--	324,3379	324,4109	329,4699	326,3036	335,4298	327,8126	350,6325	325,4641
Desvio Padrão (kg)	--	0,1023	0,2420	2,6642	0,6500	2,4183	11,2197	3,7128	1,2613

Tabela 12: Resultados da literatura e deste trabalho na otimização da treliça de 72 barras

O HKA obteve a convergência mais rápida para a treliça de 72 barras, assim como na treliça de 10 barras. Todavia, para este caso da treliça de 72 barras, o PSO obteve um resultado mais próximo do HKA do que na solução do problema da treliça de 10 barras. Além disso, o SA obteve um resultado mais distante dos outros algoritmos na treliça de 72 barras do que no caso da treliça de 10 barras.

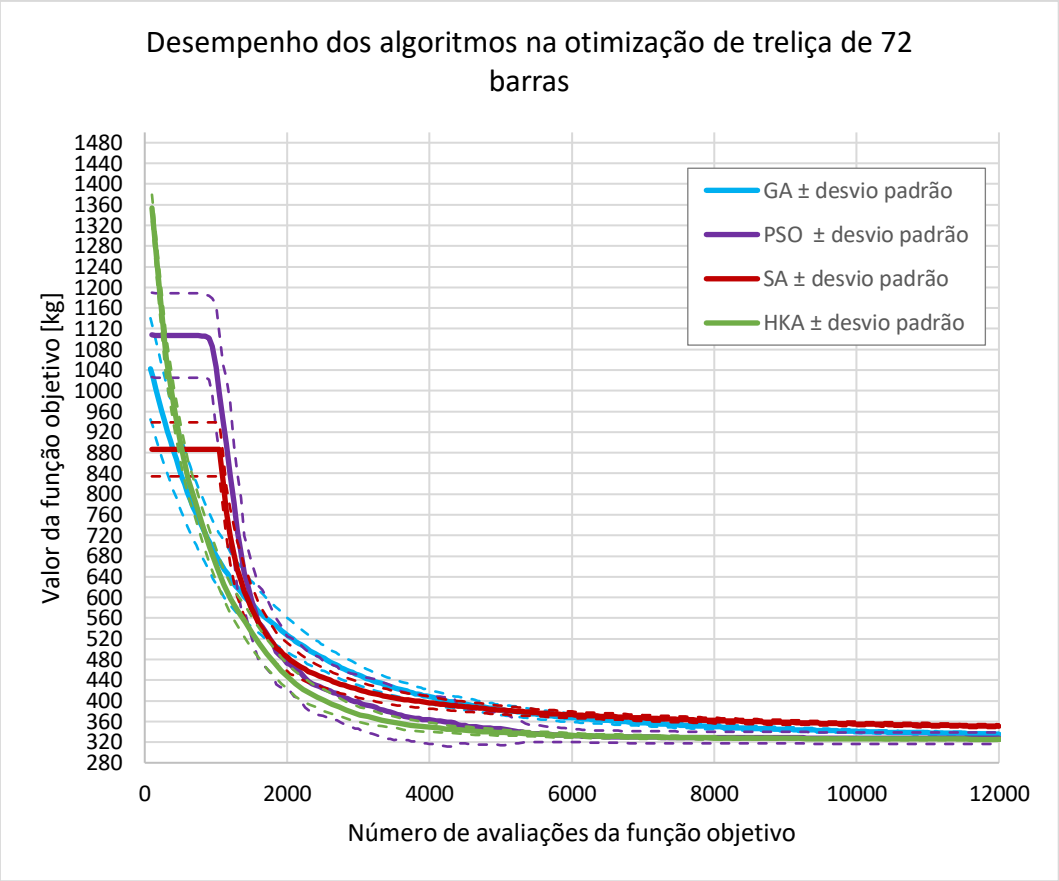

Figura 13: Desempenho do GA, PSO, SA e HKA na otimização da treliça de 72 barras

Nas primeiras iterações, não houve melhora no SA e no PSO. Para o SA, esse comportamento ocorre devido ao fato de este algoritmo ser baseado em um único indivíduo. Assim, se faz necessária uma amostra inicial, de 1000 treliças neste caso, para determinação da temperatura inicial. Já para o PSO, não houve melhora nas primeiras iterações devido à relação entre a diversidade da população inicial e o valor máximo de velocidade tolerado, que neste caso é constante ao longo das iterações.

A Figura 14 apresenta o *box plot* com as estatísticas dos valores finais da função objetivo em cada um dos quatro algoritmos para o problema da treliça de 72 barras. Nela, a linha vermelha representa a mediana, os limites inferior e superior da caixa azul representam o 25º quartil e o 75º percentil, respectivamente, os limites superior e inferior tracejados representam os valores mais extremos desconsiderando os outliers, que são as cruzes vermelhas.

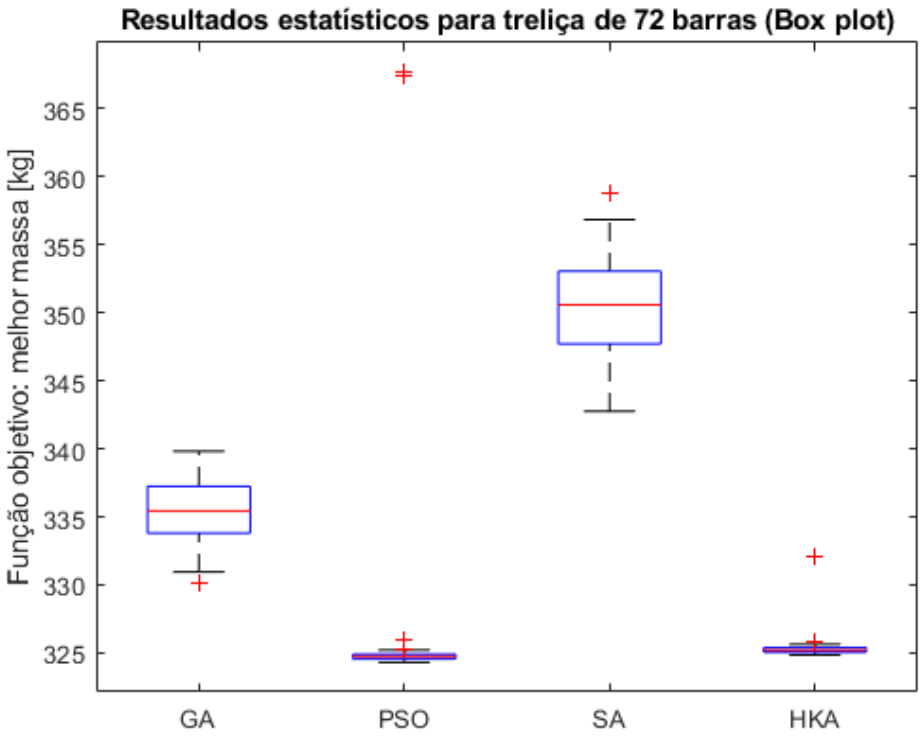

Figura 14: Box plot estatístico do GA, PSO, SA e HKA na otimização da treliça de 72 barras

O SA obteve um pior desempenho para mediana e para a dispersão dos resultados do que os outros algoritmos no caso da treliça espacial de 72 barras. Tanto o PSO quanto o HKA obtiveram resultados muito próximos entre si de mediana e desvio padrão com relação aos outros algoritmos. Porém, o PSO obteve os piores *outliers* dentre todos os algoritmos, o que impactou no seu desvio padrão neste caso.

De acordo com os *box plots* para treliças de 10 e 72 barras, em comparação com os outros algoritmos, o PSO apresentou uma menor mediana na solução da treliça de 72 barras do que na treliça de 10 barras. Já o GA e o SA apresentaram uma maior mediana na solução da treliça de 72 do que na treliça de 10 barras. O HKA apresentou mediana próxima da mínima para os dois casos.

A partir dos resultados obtidos, podemos concluir que para cada problema um algoritmo pode se apresentar como melhor opção. Além disso, um algoritmo pode desempenhar melhor com relação a um parâmetro, porém apresentar um resultado pior em outros parâmetros estatísticos. Para a treliça plana de 10 barras, neste trabalho, o HKA

apresentou melhor massa absoluta, de 524,6064 kg, melhor média, e melhor desvio padrão dentre os 4 algoritmos avaliados, isto é, GA, PSO, SA e HKA.

Por outro lado, para o caso do problema da treliça espacial de 72 barras, o desempenho de cada algoritmo foi diferente. Neste caso, o PSO apresentou a melhor massa absoluta, de 324,3094 kg, porém gerou os piores *outliers* dentre os algoritmos aplicados. Assim como na treliça de 10 barras, o HKA obteve a melhor massa média e o melhor desvio padrão dentre os algoritmos estudados. Com isso, verificamos o *no free lunch* (NFL) *theorem*, detalhada em [2]. Esta teoria afirma que todos os algoritmos desempenham igualmente bem numa média de todos os possíveis problemas.

5
Conclusões

Comparado os problemas da treliça plana de 10 barras e a treliça espacial de 72 barras, verificamos o *no free lunch* (NFL) *theorem*, que afirma que não existe algoritmo melhor que outro quando avaliados numa média de todos os problemas possíveis [2]. Para a treliça de 10 barras, o HKA obteve a menor massa absoluta, enquanto para a treliça de 72 barras, o PSO obteve a menor massa absoluta.

Em termos gerais, considerando todos os resultados da literatura, o AHEFA [18] obteve a melhor massa absoluta tanto para a treliça plana de 10 barras quanto para a treliça espacial de 72 barras. No caso da treliça de 10 barras, o menor valor foi de 524,4516 kg. Já no caso da treliça de 72 barras, o menor valor foi de 324,2376 kg.

As diferenças entre os resultados do melhor algoritmo deste trabalho, HKA, e o melhor algoritmo geral da literatura, AHEFA [18], foram menores que 1%. Para a treliça plana de 10 barras, essa diferença foi de 0,030% para o melhor valor e de 0,540% para o valor médio. Já para a treliça de espacial de 72 barras, a diferença percentual entre o HKA e o AHEFA [18] foi de 0,188% para o melhor valor e de 0,324% para o valor médio.

Mesmo com os algoritmos puros, isto é, sem aprimoramentos, alguns algoritmos deste trabalho apresentaram resultados melhores que algoritmos aprimorados recentes na literatura. No caso da treliça de 10 barras, o HKA obteve valor final absoluto da função objetivo melhor que o ISOS [7] e o CSS-BBBC [19]. Para a treliça de 72 barras, o PSO e o HKA obtiveram valores finais absolutos da função objetivo melhores que o ACCS [6], o ISOS [7], CSS-BBBC [19].

Para o futuro, propõe-se a abrangência dos testes dos algoritmos, GA, PSO, SA e HKA, para outros benchmarks de treliças típicos da literatura. Além disso, propomos a otimização dessas treliças com uma versão aprimorada do HKA, o α-HKA [11], em que o coeficiente de desaceleração é ajustado automaticamente de acordo com a dispersão dos resultados.

6
Referências bibliográficas

[1] MIGUEL, Letícia Fleck Fadel; MIGUEL, Leandro Fleck Fadel. Shape and size optimization of truss structures considering dynamic constraints through modern metaheuristic algorithms. **Expert Systems with Applications**, v. 39, n. 10, p. 9458-9467, 2012.

[2] SIMON, Dan. **Evolutionary optimization algorithms**. John Wiley & Sons, 2013.

[3] TOSCANO, Rosario; LYONNET, Patrick. Heuristic Kalman algorithm for solving optimization problems. **IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)**, v. 39, n. 5, p. 1231-1244, 2009.

[4] TEJANI, Ghanshyam G.; SAVSANI, Vimal J.; PATEL, Vivek K. Adaptive symbiotic organisms search (SOS) algorithm for structural design optimization. **Journal of Computational Design and Engineering**, v. 3, n. 3, p. 226-249, 2016.

[5] FARSHCHIN, M.; CAMP, C. V.; MANIAT, M. Multi-class teaching–learning-based optimization for truss design with frequency constraints. **Engineering Structures**, v. 106, p. 355-369, 2016.

[6] KAVEH, Ali; KOOSHKBAGHI, Mohsen. Enhanced Artificial Coronary Circulation System Algorithm for Truss Optimization with Multiple Natural Frequency Constraints. **Periodica Polytechnica Civil Engineering**, 2019.

[7] TEJANI, Ghanshyam G. et al. Truss optimization with natural frequency bounds using improved symbiotic organisms search. **Knowledge-Based Systems**, v. 143, p. 162-178, 2018.

[8] CARVALHO, José PG et al. Truss optimization with multiple frequency constraints and automatic member grouping. **Structural and Multidisciplinary Optimization**, v. 57, n. 2, p. 547-577, 2018.

[9] KAVEH, A.; MAHJOUBI, S. Hypotrochoid spiral optimization approach for sizing and layout optimization of truss structures with multiple frequency constraints. **Engineering with Computers**, p. 1-20, 2018.

[10] COELLO, Carlos A. Coello. Use of a self-adaptive penalty approach for engineering optimization problems. **Computers in Industry**, v. 41, n. 2, p. 113-127, 2000.

[11] AYALA, Helon V. Hultmann; DOS SANTOS COELHO, Leandro; REYNOSO-MEZA, Gilberto. Heuristic Kalman Algorithm for

Multiobjective Optimization. **IFAC-PapersOnLine**, v. 50, n. 1, p. 4460-4465, 2017.

[12] TOSCANO, Rosario; LYONNET, Patrick. A Kalman optimization approach for solving some industrial electronics problems. **IEEE Transactions on Industrial Electronics**, v. 59, n. 11, p. 4456-4464, 2012.

[13] LINGYUN, Wei et al. Truss optimization on shape and sizing with frequency constraints based on genetic algorithm. **Computational Mechanics**, v. 35, n. 5, p. 361-368, 2005.

[14] GOMES, Herbert Martins. Truss optimization with dynamic constraints using a particle swarm algorithm. **Expert Systems with Applications**, v. 38, n. 1, p. 957-968, 2011.

[15] SEDAGHATI, R.; SULEMAN, A.; TABARROK, B. Structural optimization with frequency constraints using the finite element force method. **AIAA journal**, v. 40, n. 2, p. 382-388, 2002.

[16] GRANDHI, Ramana. Structural optimization with frequency constraints-a review. **AIAA journal**, v. 31, n. 12, p. 2296-2303, 1993.

[17] HO-HUU, V. et al. Optimal design of truss structures with frequency constraints using improved differential evolution algorithm based on an adaptive mutation scheme. **Automation in Construction**, v. 68, p. 81-94, 2016.

[18] LIEU, Q. X. et al. An adaptive hybrid evolutionary firefly algorithm for shape and size optimization of truss structures with frequency constraints. **Computers & Structures**, 195, 99-112, 2018.

[19] KAVEH, A. AND ZOLGHADR, A. Truss optimization with natural frequency constraints using a hybridized CSS-BBBC algorithm with trap recognition capability. **Computers & Structures**, 102, pp.14-27, 2012.

[20] HOSSEINZADEH, Y., TAGHIZADIEH, N., & JALILI, S. Hybridizing electromagnetism-like mechanism algorithm with migration strategy for layout and size optimization of truss structures with frequency constraints. **Neural Computing and Applications**, 27(4), 953-971, 2016.

[21] CHRISTENSEN, P. W., & KLARBRING, A. An introduction to structural optimization (Vol. 153). **Springer Science & Business Media**. 2008

Anexo A – Código Matlab

Para auxiliar na compreensão dos procedimentos apresentados neste trabalho, são mostrados a seguir alguns códigos do Matlab, que também podem colaborar como referências para trabalhos futuros. Para implementação dos algoritmos GA, PSO e SA, foram utilizados como referência o trabalho de Simon [2], e para implementação do HKA foi utilizado como referência o trabalho de Toscano et al. [3].

Run30Times.m

```
clc
clear all
close all

global sol countObjFunCalls ObjFun_ObjFunCalls;

solutionsMatrix = zeros(20,30);
%['G1',...'G16';'mass';'f1';'f3';'NumOjbFunCalls'];
ObjFun_ObjFunCalls_Matrix = zeros(2000,31);

rng('default');

for k=1:30
    k
    fprintf('%s\n',datestr(now,'HH:MM'));

    countObjFunCalls = 0;
    run('SeventyTwoBarTruss_SA.m');
    truss=sol(2:17);

    mass=GetMass72bar(truss,NODE,ELEM,rho)
    [c1,c3] = GetConstraints72bar(truss,NODE,ELEM,E,rho);
    f1 = (1-c1)*4
    f3 = (1-c3)*6

    solutionsMatrix(:,k) = [transpose(10000*truss); mass; f1;f3;
countObjFunCalls];
    ObjFun_ObjFunCalls_Matrix(1:size(ObjFun_ObjFunCalls,1),1) =
ObjFun_ObjFunCalls(:,1);
    ObjFun_ObjFunCalls_Matrix(1:size(ObjFun_ObjFunCalls,1),k+1) =
ObjFun_ObjFunCalls(:,2);
end

xlswrite('TABLEsolutionsSA.xlsx', solutionsMatrix);
xlswrite('TABLEObjFunVsObjFunCallsSA.xlsx',
ObjFun_ObjFunCalls_Matrix);
```

SeventyTwoBarTruss_SA.m

```
% clc
% clear all
% close all

global E rho W1 W2 NODE ELEM sol

a = 1.524; % 60*0.0254;
NODE = zeros(20,3);
ELEM = zeros(72,2);
for Level = 1:5
  z = 4*a -(Level-1)*a;
  nn = (Level-1)*4;
  nnn= (Level  )*4;
  rn = (Level-1)*18;
  rnn= (Level  )*18;

  NODE(nn+1,:) =  [  0,  0,z];
  NODE(nn+2,:) =  [2*a,  0,z];
  NODE(nn+3,:) =  [2*a,2*a,z];
  NODE(nn+4,:) =  [  0,2*a,z];

  if Level < 5
    % vertical rods
    for i = 1:4
      ELEM(rn+i,:) = [nn+i,nnn+i];
    end

    % diagonal rods
    ELEM(rn+5,:) = [nnn+1,nn+2];
    ELEM(rn+6,:) = [nn+1,nnn+2];
    ELEM(rn+7,:) = [nnn+2,nn+3];
    ELEM(rn+8,:) = [nn+2,nnn+3];
    ELEM(rn+9,:) = [nnn+3,nn+4];
    ELEM(rn+10,:) = [nn+3,nnn+4];
    ELEM(rn+11,:) = [nnn+4,nn+1];
    ELEM(rn+12,:) = [nn+4,nnn+1];

    % horizontal rods
    ELEM(rn+13,:) = [nn+1,nn+2];
    ELEM(rn+14,:) = [nn+2,nn+3];
    ELEM(rn+15,:) = [nn+3,nn+4];
    ELEM(rn+16,:) = [nn+4,nn+1];
    ELEM(rn+17,:) = [nn+1,nn+3];
    ELEM(rn+18,:) = [nn+2,nn+4];
  end
end
NElem = size(ELEM,1);

E = 6.98*1e10; %6.89475729*1e10; % N/m^2 - 10^7 psi
(10^7*4.4482216152/0.0254^2 N/m^2
rho = 2770; %2767.99047; % kg/m^3 - 0.1 lb/in^3
(0.1*0.45359237/0.0254^3 kg/m^3)
```

```
NVar = 16;                      % Number of variables

W1 = 300; %Penalizing factor 1 - to be adjusted (count)
W2 = 300; %Penalizing factor 2 - to be adjusted (sum)

%%%%% Parametros SA
N=1000; % Nb of samples for the determination of the init
temperature
minT=0; % minimal temperature for the stopping rule
coefT=0.70; % coefficient for the decrease of the temperature
stdgen=0.014; % std for the generation of a new candidate solution
NbMaxConsRej=9000; % Max nb of consecutive rejection
NbMaxTry=9000;     % Max nb of try
NbMaxSuc=9000;     % Max nb of success within one temperature
NbMaxIter=11000;    % Max number of iteration for the stoping rule
%%%%%

param=[N minT coefT stdgen NbMaxConsRej NbMaxTry NbMaxSuc
NbMaxIter];

lb = 0.645e-4*ones(1,NVar); % Lower bound of variables
ub = 50e-4*ones(1,NVar);    % Upper bound of variables

D = [lb; ub];

sol=SA(param,D,'ObjectiveFunction_SeventyTwoBarTruss');

10000*sol(2:17);

best_truss = sol(2:17);
mass = GetMass72bar(best_truss,NODE,ELEM,rho);
[c1,c3] = GetConstraints72bar(best_truss,NODE,ELEM,E,rho);
f1 = (1-c1)*4;
f3 = (1-c3)*6;
```

BoxPlotMaker.m

```
clc
clear all
close all

best_GA = transpose([337.6621   336.8151    334.6416    339.5644
330.9504    336.0918    332.8751    337.9062    334.8610
333.4384    334.4622    336.0872    336.1126    336.8814
336.9872    339.8135    331.6172    330.0699    339.5387
337.6573    334.1441    333.7314    333.8886    333.3632
335.1022    335.7699    334.4292    337.4878]);
best_PSO = transpose([325.3374  324.8577    324.4299    324.8474
324.6278    324.6059    324.5895    324.3094    324.9290
324.7045    324.7760    325.9544    325.2141    324.8487
324.8272    324.5974    324.4736    324.8710    324.3883
367.6657    367.3721    324.5350    324.9439    324.8838
324.5564    324.7313    324.5491    324.3252]);
best_SA = transpose([356.8272   353.4076    358.8285    354.1223
349.5512    352.6902    342.7601    345.1303    346.6490
356.4778    352.0802    348.8264    351.1420    346.5052
349.7666    354.2423    348.3144    349.9999    351.8226
352.8313    352.7472    353.2180    348.6391    346.2072
347.1169    345.7238    352.8549    349.2270]);
best_HKA = transpose([324.9100  325.4241    325.2592    325.3157
325.3889    325.5237    325.5363    325.1943    324.9335
325.1324    332.0231    325.2353    325.1101    325.8903
324.9775    325.2836    324.8469    325.3966    325.0523
324.9860    325.0736    325.2145    325.0587    325.0376
325.6399    325.2533    325.1440    325.1362]);

boxplot([best_GA, best_PSO, best_SA,
best_HKA],'Labels',{'GA','PSO','SA','HKA'},'Whisker',1);
ylabel('Função objetivo: melhor massa [kg]')
title('Resultados estatísticos para treliça de 72 barras (Box
plot)')
```