Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas / Faculdade de Engenharia
Programa de Pós-Graduação em Modelagem Computacional

Érica da Costa Reis Carvalho

Um algoritmo por enxame de partículas para a solução de problemas de otimização estrutural multiobjetivo considerando frequências naturais de vibração

Juiz de Fora
2018

Érica da Costa Reis Carvalho

Um algoritmo por enxame de partículas para a solução de problemas de otimização estrutural multiobjetivo considerando frequências naturais de vibração

Tese apresentada ao Programa de Pós-Graduação em Modelagem Computacional da Universidade Federal de Juiz de Fora, na área de concentração em Modelagem Computacional, como requisito parcial para obtenção do título de Doutora em Modelagem Computacional.

Orientador: Prof. Afonso Celso de Castro Lemonge

Coorientador: Prof. Heder Soares Bernardino

Coorientadora: Prof.ª Patrícia Habib Hallak

Juiz de Fora

2018

Ficha catalográfica elaborada através do programa de geração
automática da Biblioteca Universitária da UFJF,
com os dados fornecidos pelo(a) autor(a)

Carvalho, Érica da Costa Reis.
Um algoritmo por enxame de partículas para a solução de problemas de otimização estrutural multiobjetivo considerando frequências naturais de vibração / Érica da Costa Reis Carvalho. -- 2018.
166 p. : il.

Orientador: Afonso Celso de Castro Lemonge
Coorientadores: Heder Soares Bernardino, Patrícia Habib Hallak
Tese (doutorado) - Universidade Federal de Juiz de Fora, ICE/Engenharia. Programa de Pós-Graduação em Modelagem Computacional, 2018.

1. Otimização por Enxame de Partículas. 2. Otimização Estrutural. 3. Otimização Multiobjetivo. 4. Frequências Naturais de Vibração. 5. Restrições de Cardinalidade. I. Lemonge, Afonso Celso de Castro, orient. II. Bernardino, Heder Soares, coorient. III. Hallak, Patrícia Habib, coorient. IV. Título.

Érica da Costa Reis Carvalho

Um algoritmo por enxame de partículas para a solução de problemas de otimização estrutural multiobjetivo considerando frequências naturais de vibração

Tese apresentada ao Programa de Pós-Graduação em Modelagem Computacional da Universidade Federal de Juiz de Fora, na área de concentração em Modelagem Computacional, como requisito parcial para obtenção do título de Doutora em Modelagem Computacional.

Aprovada em: 08 de outubro de 2018

BANCA EXAMINADORA

Prof. D.Sc. Afonso Celso de Castro Lemonge - UFJF

Prof. D.Sc. Heder Soares Bernardino - UFJF

Prof.ª D.Sc. Patrícia Habib Hallak - UFJF

Prof.ª D.Sc. Beatriz de Souza Leite Pires de Lima - COPPE/UFRJ

Prof. D.Sc. Dênis Emanuel da Costa Vargas - IF Sudeste MG, Campus Rio Pomba

Prof. D.Sc. Alexandre Abrahão Cury - UFJF

Prof. D.Sc. Marcelo Miranda Barros - UFJF

Aos meus pais, que sempre investiram na minha formação.
Aos meus avós, por todo incentivo recebido.
Ao Bruno, pelo amor e companheirismo.
Aos meus verdadeiros amigos.

AGRADECIMENTOS

Aos meus pais, Ana Paula e Fernando, meu imensurável agradecimento por me apoiarem e me incentivarem ao longo de minha vida acadêmica. Amo vocês!

Ao meu noivo Bruno, sempre ao meu lado durante todo o doutorado, agradeço pela boa vontade em me ajudar, por seu carinho e por seu valioso apoio.

Aos meus avós Nídia e Nilton (in memoriam), pelo exemplo de caráter, dignidade e sabedoria e por terem me ensinado o valor da educação e do conhecimento.

Ao meu irmão Paulo Fernando, pelo carinho e alegria.

A todos os meus familiares que me apoiam e me incentivam em tudo.

Aos meus orientadores, professores Afonso Lemonge, Heder Bernardino e Patrícia Hallak, pela orientação valiosa, pela confiança e dedicação que muito me auxiliaram durante a elaboração deste trabalho.

Ao José Pedro por sua importante ajuda, sempre disponível e com muita paciência.

Aos membros da banca, pela disponibilidade e atenção em avaliar este trabalho.

Aos colegas da Pós Graduação em Modelagem Computacional, agradeço a amizade e o companheirismo.

Aos funcionários da pós, pela atenção e eficiência nos serviços prestados.

Aos colegas de trabalho do CEFET-MG Campus Leopoldina, pelo apoio e compreensão.

A todas as amizades conquistadas nessa etapa, e a todos os amigos e amigas, pelos momentos de alegria e acolhimento.

À UFJF e à FAPEMIG pelo apoio concedido.

À Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001, pela bolsa que me foi concedida.

A todos que, direta ou indiretamente, contribuíram para a realização deste trabalho.

A Deus.

“A tarefa não é tanto ver aquilo que ninguém viu,
mas pensar o que ninguém ainda pensou
sobre aquilo que todo mundo vê.”
(Arthur Schopenhauer)

RESUMO

O interesse em algoritmos de otimização multiobjetivo cresceu nos últimos anos devido à sua aplicabilidade em problemas de diversas áreas, especialmente as da engenharia. Em geral, os objetivos considerados nesses problemas são conflitantes e uma frente de Pareto composta pelas soluções não-dominadas é esperada como a solução para tais problemas. No contexto da computação evolutiva, existem diversos algoritmos aplicados a esse tipo de problema, como os algoritmos genéticos, evolução diferencial e enxame de partículas. Este trabalho tem como objetivo propor e avaliar a capacidade de um algoritmo por enxame de partículas multiobjetivo, denominado *Multiobjective Craziness based Particle Swarm Optimization* (MOCRPSO) em um conjunto de problemas de otimização estrutural multiobjetivo com restrições. O problema consiste em minimizar a massa de estruturas, considerando como segundo objetivo o deslocamento máximo dos nós ou as frequências naturais de vibração da estrutura. Adicionalmente, restrições de cardinalidade são adotadas a fim de obter uma busca automática da ligação das variáveis, procurando o melhor agrupamento dos membros. Um Método de Penalização Adaptativa (APM), que foi aplicado com sucesso na solução de problemas de otimização mono e multiobjetivo, é usado aqui para lidar com as restrições. Para investigar o desempenho do método proposto, seis problemas teste com e sem restrições e seis estruturas treliçadas são analisadas e os resultados encontrados ilustram sua eficiência quando comparados com outros algoritmos encontrados na literatura.

Palavras-chave: Otimização por Enxame de Partículas. Otimização Estrutural. Otimização Multiobjetivo. Frequências Naturais de Vibração. Restrições de Cardinalidade.

ABSTRACT

The interest in multiobjective optimization algorithms has grown in recent years due to its applicability in problems from several areas, especially those from engineering. In general, the objectives considered in these problems are conflicting and a Pareto Front curve composed of the non-dominated solutions is expected as the solution to such problems. In the context of evolutionary computation there are many of algorithms applied to this type of problem, such as genetic algorithms, differential evolution, and particle swarm. This study aims to evaluate the ability of a multiobjective particle swarm algorithm, called Multiobjective Craziness based Particle Swarm Optimization (MOCRPSO) in a set of unconstrained multiobjective structural optimization problems. The problem consists to minimize the mass of structures, considering as second objective the displacement of the nodes or the natural frequencies of vibration of the strucuture. Additionally, cardinality constraints are adopted in order to obtain an automatic variable linking, searching for the best member grouping of the bars. An Adaptive Penalty Method (APM), which has been successfully applied to solving mono and multiobjective optimization problems, is used here to handle the constraints. To investigate the performance of the proposed method, six test-problems with and without constraints and six truss structures are analyzed and the results obtained illustrate its efficiency when compared to other algorithms from the literature.

Key-words: Particle Swarm Optimization. Structural Optimization. Multiobjective Optimization. Frequency Natural Vibration. Cardinality Constraints.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AG	Algoritmo Genético
AI	*Artificial Intelligence*
APM	*Adaptive Penalty Method*
AMOPSO	*Accelerated MO Particle Swarm Optimization*
CMOP	*Constrained Multiobjective Optimization Problem*
CRPSO	*Craziness based Particle Swarm Optimization*
DEMO	*Differential Evolution for Multiobjective Optimizatio*
DE	*Differential Evolution*
EA	*Evolutionary Algorithm*
EAF	*Empirical Attainment Function*
FEM	*Finite Element Method*
GDE3	*The Third Step of Generalized Differential Evolution*
GDE3-APM	Algoritmo GDE3 acoplado à tecnica APM
MO	*Multiobjective Optimization*
MOACS	*Multiobjective Ant Colony System*
MOAS	*Multiobjective Ant System*
MOEA	*Multiobjective Evolutionary Algorithm*
MOP	*Multiobjective Optimization Problem*
MOCRPSO	*Multiobjective Craziness based Particle Swarm Optimization*
MODE	*Multiobjective Differential Evolution*
MOPSO	*Multiobjective Particle Swarm Optimization*
NSGA-II	*Nondominated Sorting Genetic Algorithm II*
PFAE	*Pareto Front Approximation Error*
PSO	*Particle Swarm Optimization*
PPs	*Performance Profiles*

SI	*Swarm Intelligence*
SPEA	*Strength Pareto Evolutionary Algorithm*
VEGA	*Vector Evaluated Genetic Algorithm*
ZDT	*Zitzler-Deb-Thiele*
M	Matriz de massa
C	Matriz de amortecimento
K	Matriz de rigidez
$\{x\}$	Vetor de deslocamento
$\{\dot{x}\}$	Vetor de velocidade
$\{\ddot{x}\}$	Vetor de aceleração
$\{p\}$	Vetor de cargas nodais
ω	Frequência natural de vibração
K_E	Matriz de rigidez do elemento
M_E	Matriz de massa do elemento
E	Módulo de elasticidade
A	Área da seção transversal
L	Comprimento do membro
ρ	Massa específica
K_G	Matriz de rigidez global
M_G	Matriz de massa global
T	Matriz de rotação
K_R	Matriz de rigidez reduzida
W	Peso da estrutura
u	Deslocamento do nó
n_a	Número de barras
n_d	Número de graus de liberdade

σ	Tensão da barra
$\bar{\sigma}$	Tensão máxima admitida
n_f	Número de frequências naturais de vibração
$\bar{u}$	Deslocamento máximo admitido
n	Dimensão do espaço
v	Velocidade da partícula
x	Posição da partícula
$gBest$	Melhor posição global
$pBest$	Melhor posição local
$F(x)$	Função aptidão
$f(x)$	Função objetivo
$penal(x)$	Função de penalização
g	Restrição de desigualdade
h	Restrição de igualdade
n_g	Número de restrições de desigualdade
n_h	Número de restrições de igualdade
n_c	Número total de restrições
$viol(x)$	Violação de cada restrição
k	Parâmetro de penalização
pop	Número de violações de todos os indivíduos da população
m	Cardinalidade utilizada
n_{card}	Número máximo de restrições de cardinalidade
a	Limite inferior
b	Limite superior
n_{obj}	Número de funções objetivo
$v^{craziness}$	Operador *craziness*

ARQ	Arquivo externo
$pMut$	Taxa de mutação

SUMÁRIO

1 INTRODUÇÃO

Na engenharia estrutural, os recentes avanços tecnológicos aliados ao desenvolvimento de recursos e técnicas computacionais têm permitido projetos de estruturas cada vez mais arrojadas. Esta tendência tem levado ao projeto de estruturas leves e flexíveis que demandam um aprimoramento dos modelos para os cálculos estruturais. Neste contexto, a consideração em projeto das ações dinâmicas passa a ser mandatório no intuito de mitigar ou evitar possíveis vibrações indesejadas, que podem levar ao desconforto humano ou, até mesmo, ao comprometimento do desempenho estrutural.

Dentre as ações que induzem as vibrações estruturais, têm-se as ambientais, tais como vento, veículos, pedestres, dentre outras. Esforços e deslocamentos provenientes de carregamentos dinâmicos são de vital importância. Desta forma, o estudo da dinâmica das estruturas não pode ser desprezado nas modelagens mais complexas que atualmente são mais fidedignas e desejadas.

A solução de problemas de engenharia implica, na maioria das vezes, na otimização envolvendo múltiplos objetivos, frequentemente conflitantes [8]. Estes problemas de otimização não possuem uma única solução ótima, mas um conjunto de soluções ditas não-comparáveis. Quando modelos estruturais são submetidos a problemas de otimização, em geral, objetiva-se minimizar o custo dos materiais e satisfazer as condições de conforto e segurança. Na tentativa de minimizar o custo desses materiais, o principal parâmetro envolvido no processo é o peso da estrutura. Todavia, ao se minimizar o peso afeta-se diretamente as propriedades dinâmicas da estrutura.

A consideração de restrições impostas ao problema de otimização limitando as frequências naturais de vibração é de aplicação inquestionável nestes tipos de problemas. Sendo assim, a maximização dessas frequências pode ser utilizada como uma forma de se obter um espectro de frequências mais favorável, tornando a estrutura menos sensível às cargas dinâmica. A ideia é projetar uma estrutura com frequências naturais diferentes das frequências de excitação evitando, portanto, problemas de ressonância, que podem levar a estrutura ao colapso.

Muitos autores [9, 10, 11] relatam que problemas de otimização dimensional envolvendo frequências naturais de vibração são problemas de otimização não-lineares. Esta complexidade se deve ao fato de que a redução do peso conflitua com as limitações de frequências, principalmente quando estas são limitadas a valores muito baixos. Além disso, as frequências naturais de vibração são extremamente sensíveis ao número de barras (no caso de estruturas reticuladas com uma variação de topologia). Ao aumentar o número de variáveis de projeto de uma estrutura, consequentemente, sua busca é dificultada.

Os problemas de otimização multiobjetivo têm despertado grande interesse na área de Otimização [12, 13, 14, 15, 16, 17, 18]. Nestes problemas, a qualidade da solução

é definida com base na relação entre os seus objetivos [19, 20]. Para tratar essa classe de problemas, vários métodos vêm sendo desenvolvidos ao longo dos anos. Dentre eles, destacam-se os Algoritmos Evolutivos (*Evolutionary Algorithms* - EAs) [21], baseados em processos naturais. Estes métodos são comumente caracterizados pela presença de mecanismos artificiais inspirados na biologia, tais como, reprodução, mutação, etc. De forma geral, os EAs vêm ganhando uma crescente popularidade no campo científico e das engenharias [22, 23, 20].

O algoritmo de Otimização por Enxame de Partículas (*Particle Swarm Optimization* - PSO) [24] é um exemplo de EA bastante difundido e fornece modelos computacionais baseados no conceito de inteligência coletiva. Um PSO modificado denominado *Craziness based Particle Swarm Optimization* (CRPSO) e proposto por Kar *et al.* [25], é usado nesta tese e apresenta uma nova expressão para a velocidade e um operador denominado "*craziness velocity*". Uma nova abordagem que amplia o algoritmo CRPSO para lidar com problemas de otimização multiobjetivo foi proposta e desenvolvida pela autora desta tese e ficou denominada de *Multiobjective Craziness based Particle Swarm Optimization* (MOCRPSO). O novo algoritmo incorpora um mecanismo de *crowding distance* juntamente com um operador de mutação baseado no algoritmo MOPSO-CD, desenvolvido por Raquel & Naval [5]. O MOCRPSO é usado neste estudo como o algoritmo de busca nos problemas de otimização multiobjetivo analisados.

Um novo conjunto de problemas de otimização é analisado aqui considerando restrições de cardinalidade para descobrir o melhor agrupamento dos membros, bem como o uso de espaços discretos ou mistos (envolvendo variáveis discretas e contínuas). Para tratar essas restrições, o MOCRPSO é usado com uma codificação especial das partículas candidatas [26], limitando o número máximo de áreas distintas das seções transversais a serem usadas em uma solução otimizada. Este tipo de restrição é importante pois proporciona economia na fabricação, na compra de grandes volumes, no transporte, na soldagem, na conferência das partes executadas e, certamente, alivia o projetista da tarefa de decidir sobre a definição do melhor agrupamento das barras. Neste estudo, os problemas com restrições são transformados em problemas sem restrições através da introdução de um Método de Penalidade Adaptativa (*Adaptive Penalty Method* - APM), proposto por Barbosa & Lemonge em [27].

1.1 JUSTIFICATIVA

Os problemas de otimização em engenharia cada vez mais têm incorporado complexidade em suas modelagens, análises e tomadas de decisões no que dizem respeito à escolha da melhor solução dentro de um conjunto disponível de soluções apresentadas. Isso se traduz no caso em que várias soluções otimizadas são oferecidas ao projetista, e este deve escolher qual a mais adequada, considerando aspectos relevantes para a sua decisão.

Esta situação é muito comum em problemas de otimização onde ocorrem conflitos entre o peso da estrutura e seu deslocamento máximo. Ou, ainda, quando é necessário afastar os valores das frequências naturais de vibração de uma estrutura das frequências de sua fonte de excitação, a fim de evitar o fenômeno da ressonância.

Baseado nas informações apresentadas, justifica-se a escolha deste tema ao fato de que, na literatura não são encontrados trabalhos que usam um algoritmo baseado em enxame de partículas combinado com a técnica de penalização APM para a solução de problemas de otimização estrutural multiobjetivo envolvendo, como um dos objetivos, as frequências naturais de vibração. E ainda, adicionando-se na modelagem do problema o agrupamento ótimo das barras.

1.2 OBJETIVOS

Considerando as informações apresentadas na justificativa, esta tese tem como objetivo propor um algoritmo de otimização por enxame de partículas multiobjetivo denominado *Multiobjective Crazinesse based Particle Swarm Optimization* (MOCRPSO) para a análise de problemas de otimização, envolvendo objetivos e restrições de várias naturezas. Esse objetivo tem caráter inovador na literatura em razão do algoritmo proposto e das aplicações em que o mesmo será submetido.

Inicialmente, um conjunto de seis funções teste com e sem restrições, tradicionalmente encontradas na literatura [28], foram utilizadas para avaliar o desempenho do algoritmo proposto. Os resultados encontrados serão comparados com aqueles encontrados pelo algoritmo NSGA-II (*Nondominated Sorting Genetic Algorithm II*) [28]. Após a avaliação preliminar do algoritmo proposto, o objetivo principal deste estudo envolverá a solução de dois problemas de otimização estrutural que possui, entre seus objetivos:

(i) a minimização do peso da estrutura e a minimização do máximo deslocamento, envolvendo restrições de tensões normais máximas e o agrupamento ótimo das barras;

(ii) a minimização do peso da estrutura e a maximização da frequência natural de vibração, envolvendo restrições de deslocamentos máximos, tensões normais máximas e o agrupamento ótimo das barras.

A maximização da frequência natural de vibração, apresentada no item (ii), é dividida em três formulações de problemas de otimização diferentes:

- maximizar a menor frequência;
- maximizar a soma das frequências e

- maximizar a maior frequência.

Os resultados encontrados em (i) serão comparados com os resultados de dois pares de algoritmos multiobjetivo: GDE3 (*The Third Step of Generalized Differential Evolution*) e GDE3-APM, apresentados por Vargas *et al.* [6] e MOAS (*Multiobjective Ant System*) e MOACS (*Multiobjective Ant Colony System*), apresentados por Angelo *et al.* [14].

1.3 ESTRUTURA DO TRABALHO

A presente tese está dividida em 7 capítulos, incluindo esta introdução que apresenta uma visão geral sobre o tema que será tratado durante os próximos capítulos. No Capítulo 2, são apresentados conceitos referentes ao problema de otimização estrutural, a descrição do problema de dinâmica estrutural, bem como detalhes para a obtenção das frequências naturais de vibração. Ainda nesse capítulo, o problema de otimização estrutural é formulado e as estruturas utilizadas no conjunto de experimentos são enunciadas. O algoritmo PSO padrão é apresentado e detalhado no Capítulo 3, juntamente com o método para tratamento de restrições e as restrições de cardinalidade. Ao final do capítulo, uma breve seleção de trabalhos relacionados a este estudo é apresentada. Conceitos básicos e uma revisão bibliográfica da otimização multiobjetivo é apresentada no Capítulo 4, juntamente com a descrição de alguns dos principais Algoritmos Evolutivos Multiobjetivos. No Capítulo 5, o algoritmo multiobjetivo proposto é descrito e detalhes de seus parâmetros e de sua codificação são apresentados. No Capítulo 6, os experimentos numéricos são descritos, analisados e comparados. Uma análise estatística envolvendo resultados encontrados na literatura é apresentada. Finalmente, no último capítulo são apresentadas as conclusões obtidas com o desenvolvimento desta tese, bem como propostas para trabalhos futuros.

2 OTIMIZAÇÃO ESTRUTURAL

A palavra otimização já se tornou corriqueira nos tempos atuais e pode ser definida como "fazer tão perfeito, efetivo ou funcional quanto possível" [29]. A otimização tem como meta dimensionar um produto da melhor forma possível em relação a um objetivo, de modo que este produto atenda às condições impostas pela sua concepção. Por exemplo: na indústria aeronáutica, o peso é um fator preponderante para se obter um bom projeto de uma peça e é necessário reduzir ao máximo esta grandeza. No entanto, esta peça deve resistir com segurança a todos os esforços nela aplicados. Além do peso, outra variável de suma importância para a indústria é o custo final do produto. Desta forma, minimizando-se o peso, tem-se como consequência o uso de menos matéria-prima para se fabricar uma peça, resultando em redução do custo. Em uma visão empresarial, cada grama de material economizado pode representar um acréscimo considerável nos lucros para uma empresa, fazendo com que esta se diferencie em relação às concorrentes. Logo, projetar sem utilizar a otimização pode acarretar em superdimensionamentos desnecessários ao projeto.

O conceito de otimização estrutural é mais antigo do que se pensa. De acordo com Gandomi *et al.* [30], o primeiro cientista a aplicar esse conceito foi Maxwell em 1869. Ele formulou uma teoria que estabeleceu a base para o desenvolvimento de projetos de minimização de peso de estruturas. Baseada na teoria de Maxwell, uma nova formulação para a minimização de peso de estruturas reticuladas foi proposta por Michell em 1904. As primeiras formulações envolviam treliças e pórticos planos, grelhas, placas e cascas, geralmente considerando o aço como material. As dimensões do problema quanto ao número de variáveis, complexidade da função objetivo e restrições dependia dos recursos computacionais disponíveis na época. As funções objetivo, em quase sua totalidade, diziam respeito a um único objetivo que era a minimização do peso da estrutura e as restrições relacionadas às tensões e deslocamentos para casos estáticos de carregamento [31].

Espera-se da otimização estrutural uma ferramenta que automatiza todo o processo de criação de uma concepção estrutural de forma mais independente possível da experiência dos projetistas. Evidentemente, a experiência do projetista não deve ser desprezada, pois é de extrema importância e pode ser enriquecida com as soluções apontadas pelos processos de otimização. Comumente, nos problemas de otimização estrutural, define-se uma ou mais funções que representam o custo ou o desempenho que, em sua maioria, são submetidas a restrições. Estas funções são definidas como funções objetivo. Quando os projetos demandam a análise de características dinâmicas, estes aspectos são incluídos na formulação do problema. Em muitos casos, as restrições limitam deslocamentos, tensões e, no caso desses aspectos dinâmicos, por exemplo, frequências naturais de vibração.

Um problema de otimização estrutural pode ser formulado de diversas maneiras, podendo variar seus objetivos e suas restrições. É possível categorizá-lo em três tipos

(i) otimização dimensional: utiliza como variável de projeto um parâmetro de um elemento estrutural. No caso de treliças, a variável de projeto é a área da seção transversal. (ii) otimização geométrica (ou de forma): para esse tipo de otimização, a forma da estrutura pode ser alterada, porém o número de barras e suas conectividades devem ser mantidos e (iii) otimização topológica: permite-se alterar a conectividade das barras. Diversos trabalhos envolvendo problemas de otimização estrutural têm sido desenvolvidos ao longo dos anos [32, 33, 34, 15].

2.1 O PROBLEMA DE DINÂMICA ESTRUTURAL

No passado, as estruturas eram examinadas utilizando apenas análises estáticas considerando ações constantes ao longo do tempo (exemplo, peso próprio e cargas permanentes). Isso ocorria principalmente devido à escassez de recursos computacionais, o que impossibilitava resolver as elevadas quantidades de cálculos envolvidos nos algoritmos de análise dinâmica. Com o decorrer do tempo, o avanço computacional tornou possível a consideração de análises dinâmicas, dando importância a inúmeros fenômenos que variam de grandeza, direção ou sentido no tempo. Desta forma, além das ações permanentes, ações como a movimentação de pessoas e dos motores de máquinas sobre lajes, a incidência do vento em edifícios altos, o tráfego rodoviário sobre pontes e a ocorrência de sismos passam a ser fundamentais na modelagem de problemas envolvendo estruturas [35].

A análise dinâmica pode ser vista como uma abordagem mais ampla da análise estrutural, uma vez que considera a variação do carregamento e das respostas da estrutura com o tempo. As estruturas são solicitadas por diversos tipos de ações, que podem introduzir comportamentos caracterizados por respostas estáticas e/ou dinâmicas. O termo dinâmica pode ser entendido simplesmente como uma variação no tempo de qualquer grandeza. Desta forma, uma carga dinâmica é aquela em que sua magnitude, direção ou posição variam com o tempo. Similarmente, as respostas estruturais a esta carga dinâmica também são variáveis no tempo.

A construção de estruturas cada vez maiores e de grande esbelteza impulsionou a necessidade de conhecer melhor o seu comportamento dinâmico, sendo importante avaliar a segurança da estrutura à ressonância [36]. A ressonância é o fenômeno que acontece quando um sistema físico recebe energia por meio de excitações de frequência igual ou próxima a uma de suas frequências naturais de vibração. Assim, esse sistema passa a vibrar com amplitudes cada vez maiores, podendo colapsar [37]. Um exemplo desse fenômeno ocorre quando cantores líricos conseguem “quebrar” taças de cristais usando apenas o som emitido pela sua voz. Uma onda sonora é emitida através da voz que é capaz de produzir vibrações em objetos situados em sua proximidade. Se a frequência emitida for equivalente à frequência natural do objeto, ocorre o fenômeno de ressonância que pode ocasionar na ruptura do mesmo.

A Ponte de Tacoma, ilustrada pela Fig. 1, é um exemplo clássico de engenharia neste contexto. A ponte era localizada no condado de Pierce, Washington - USA e entrou em colapso 4 meses após sua construção devido à ocorrência de fenômenos de ressonância, provocados pela incidência do vento na estrutura. Esse fenômeno contribuiu decisivamente para a conscientização dos engenheiros de estruturas em relação à necessidade de compreender adequadamente o comportamento dinâmico destas construções.

(a) (b)

Figura 1 – Ponte de Tacoma: (a) dia da abertura em 1º de julho de 1940 e (b) dia do colapso em 7 de novembro de 1940. Extraído de [1].

2.1.1 **Frequências naturais e modos de vibração**

Muitos sistemas dinâmicos podem ser representados por uma equação diferencial de segunda ordem linear e com coeficientes constantes. Na forma discreta, a equação de movimento de um sistema dinâmico é obtida através da seguinte equação [38]:

$$[M]\{\ddot{x}\} + [C]\{\dot{x}\} + [K]\{x\} = \{p\}, \tag{2.1}$$

onde $[M]$ é a matriz de massa, $[C]$ é a matriz de amortecimento e $[K]$ é a matriz de rigidez do modelo, $\{\ddot{x}\}$ é o vetor de aceleração, $\{\dot{x}\}$ é o vetor de velocidade, $\{x\}$ é o vetor de deslocamento de todos os graus de liberdade da estrutura e $\{p\}$ é o vetor de cargas nodais.

Nos sistemas com vibrações livres não amortecidas não há dissipação de energia, e, por isso, são definidos como sistemas conservativos. Quando não existem forças perturbadoras aplicadas ao sistema, mas a matriz de massa $[M]$ é sujeita a um deslocamento $\{x\}$ e velocidade $\{\dot{x}\}$ iniciais não nulos, ocorrem essas vibrações livres.

Para a determinação das frequências naturais de vibração, as parcelas relacionadas ao amortecimento $[C]$ e a excitação $\{p\}$ da Equação 2.1 são desconsideradas. Deste modo, a equação dinâmica que rege o comportamento de uma estrutura não amortecida em vibração livre é

$$[M]\{\ddot{x}\} + [K]\{x\} = 0. \tag{2.2}$$

A solução não trivial da Equação 2.2 constitui em resolver a Equação 2.3, que consiste na associação das matrizes de massa e rigidez dos elementos que a compõem, formando duas matrizes globais de massa e rigidez.

$$[K] - \omega^2[M] = 0. \tag{2.3}$$

As raízes da Equação 2.3 são os valores característicos ou autovalores, e correspondem ao quadrado das frequências naturais de um sistema estrutural. Para cada uma dessas raízes é associado um autovetor, que representa o modo de vibração do referido sistema.

2.1.2 **Matrizes dos elementos**

Um elemento de treliça plana, ilustrado pela Fig. 2, e espacial, ilustrado pela Fig. 3, é modelado utilizando dois nós com dois/três graus de liberdade em cada extremidade.

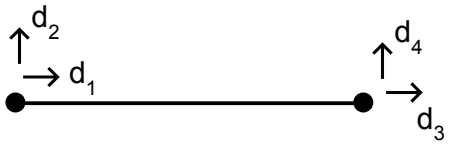


Figura 2 – Elemento de treliça plana.

Figura 3 – Elemento de treliça espacial.

A matriz de rigidez $[K_E]$ e de massa $[M_E]$ no referencial local correspondentes do sistema de coordenadas para um elemento de treliça plana são escritas como [39]

$$[K_E] = \frac{EA}{L}\begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.4}$$

e

$$[M_E] = \frac{\rho AL}{6}\begin{bmatrix} 2 & 0 & 1 & 0 \\ 0 & 2 & 0 & 1 \\ 1 & 0 & 2 & 0 \\ 0 & 1 & 0 & 2 \end{bmatrix}. \tag{2.5}$$

E para um elemento de treliça espacial

$$[K_E] = \frac{EA}{L}\begin{bmatrix} 1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.6}$$

e

$$[M_E] = \frac{\rho AL}{6}\begin{bmatrix} 2 & 0 & 0 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 & 1 & 0 \\ 0 & 0 & 2 & 0 & 0 & 1 \\ 1 & 0 & 0 & 2 & 0 & 0 \\ 0 & 1 & 0 & 0 & 2 & 0 \\ 0 & 0 & 1 & 0 & 0 & 2 \end{bmatrix}, \tag{2.7}$$

onde E, A e L e ρ são, respectivamente, o módulo de elasticidade, a área da seção transversal, o comprimento do membro e a massa específica.

Definidas as matrizes de rigidez e massa nos referencias locais, faz-se necessário a utilização da matriz de rotação para que as mesmas sejam levadas ao referencial global usando as relações

$$[K_G] = [T]^t[K_E][T] \tag{2.8}$$

e

$$[M_G] = [T]^t[M_E][T], \tag{2.9}$$

onde $[T]$ é a matriz de transformação. Para estruturas planas, a matriz pode ser expressa como

$$[T] = \begin{bmatrix} c & s & 0 & 0 \\ -s & c & 0 & 0 \\ 0 & 0 & c & s \\ 0 & 0 & -s & c \end{bmatrix}, \tag{2.10}$$

onde $c = \cos(\alpha)$, $s = \sin(\alpha)$ e α é o ângulo entre o elemento e o eixo global X. Da mesma forma, para uma treliça espacial a matriz de transformação $[T]$ pode ser escrita como

$$[T] = \begin{bmatrix} \xi_1 & \xi_2 & \xi_3 & 0 & 0 & 0 \\ \eta_1 & \eta_2 & \eta_3 & 0 & 0 & 0 \\ \zeta_1 & \zeta_2 & \zeta_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & \xi_1 & \xi_2 & \xi_3 \\ 0 & 0 & 0 & \eta_1 & \eta_2 & \eta_3 \\ 0 & 0 & 0 & \zeta_1 & \zeta_2 & \zeta_3 \end{bmatrix}, \tag{2.11}$$

onde $\{\xi_1, \eta_1, \zeta_1\}$ são os cossenos diretores na direção do eixo global X em relação ao sistema de coordenadas local xyz. Do mesmo modo, $\{\xi_2, \eta_2, \zeta_2\}$ e $\{\xi_3, \eta_3, \zeta_3\}$ são os cossenos diretores na direção dos eixos globais Y e Z em relação ao sistema de coordenadas local xyz, respectivamente.

2.1.3 **Exemplo**

A modelagem de uma treliça de 10 barras no seu referencial global é ilustrada na Fig. 4. A treliça é composta por 10 barras e 6 nós numerados de 1 a 6. Cada nó possui um deslocamento linear que pode ser decomposto, a fim de facilitar os cálculos, em duas componentes que irão acompanhar o referencial adotado (representado por d_n). A Equação 2.12 representa a matriz de rigidez dessa estrutura cheia no referencial global.

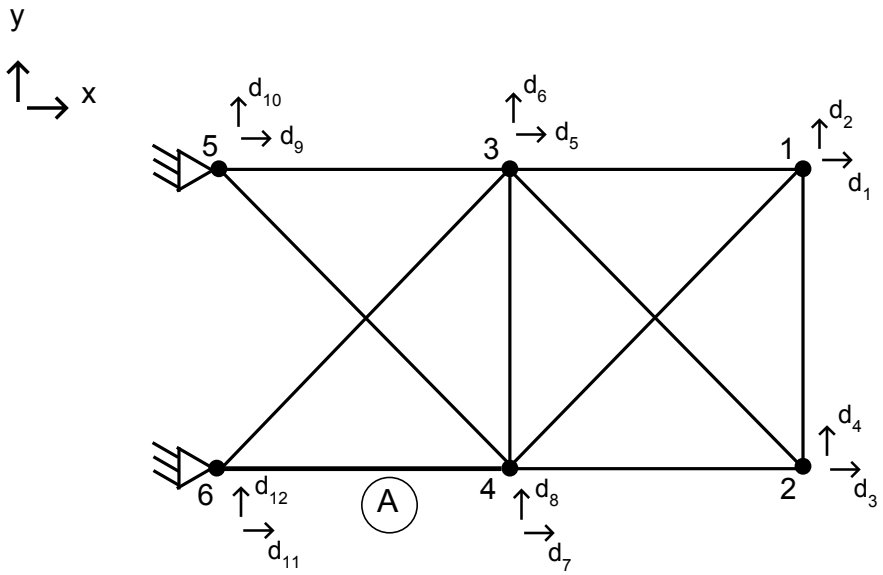


Figura 4 – Treliça de 10 barras no referencial global.

$$[K_G] = \begin{bmatrix} k_{1,1} & k_{1,2} & k_{1,3} & . & . & . & k_{1,12} \\ k_{2,1} & k_{2,2} & k_{2,3} & . & . & . & k_{2,12} \\ k_{3,1} & k_{3,2} & k_{3,3} & . & . & . & k_{3,12} \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ k_{12,1} & k_{12,2} & k_{12,3} & . & . & . & k_{12,12} \end{bmatrix}. \tag{2.12}$$

A Fig. 5 representa a barra A da treliça de 10 barras da Fig. 4 no seu referencial local. Nota-se que os índices dos deslocamentos d_n correspondem somente à numeração de cada barra em seu referencial local. Essa numeração auxilia na montagem das matrizes de massa e rigidez reduzida da estrutura.

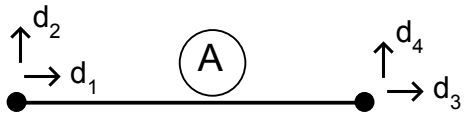


Figura 5 – Barra A da Fig. 4 no seu referencial local.

Para a barra A, seu vetor de deslocabilidades (conhecido como vetor LM) é dado por [11 12 7 8], ou seja, as deslocabilidades [1 2 3 4] da barra no referencial local (Fig. 5) correspondem às deslocabilidades [11 12 7 8] da estrutura no referencial global (Fig. 4). Isso significa que sua matriz de rigidez local, dada pela Equação 2.4, serão agrupadas nas linhas e colunas [11 12 7 8] das matrizes de rigidez cheia no referencial global, dada pela Equação 2.12.

Efetuados os agrupamentos para todas as 10 barras, tem-se a matriz cheia já apresentada na Equação 2.12. As condições de contorno são aplicadas nos nós 5 e 6: é sabido que os deslocamentos nestes nós são iguais a 0. Dessa maneira, as linhas [9 10 11 12] são anuladas, reduzindo a matriz cheia para uma matriz reduzida, apresentada na Equação 2.13. Por fim, é necessário resolver o sistema de equações (Equação 2.1) para deslocamentos e tensões ou realizar o mesmo procedimento para a matriz de massa e, assim, obter os seus respectivos autovalores (no caso de obtenção das frequências).

$$[K_R] = [K] = \begin{bmatrix} k_{1,1} & k_{1,2} & . & . & . & k_{1,8} \\ k_{2,1} & k_{2,2} & . & . & . & k_{2,8} \\ k_{3,1} & k_{3,2} & . & . & . & k_{3,8} \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ k_{8,1} & k_{8,2} & . & . & . & k_{8,8} \end{bmatrix}. \tag{2.13}$$

2.2 FORMULAÇÃO MATEMÁTICA

O problema de otimização estrutural analisado neste trabalho refere-se à otimização dimensional de treliças planas e espaciais. Objetiva-se minimizar o peso da estrutura W, tendo como variáveis de projeto as áreas das seções transversais das barras. Como se trata de um problema de otimização multiobjetivo, o mesmo foi formulado apresentando dois casos diferentes: estático e dinâmico. A formulação de cada caso é apresentada a seguir.

2.2.1 **Formulação para o caso estático**

Para o caso estático, além da minimização do peso da estrutura (primeiro objetivo), o segundo objetivo refere-se à minimização do máximo deslocamento dos nós livres. A

formulação matemática é apresentada a seguir.

$$\min \begin{cases} W = \sum_{i=1}^{n_a} \rho A_i L_i \quad \text{e} \\ \text{máximo}(|u_j|), \quad j = 1, \ldots, n_d, \end{cases} \tag{2.14}$$

onde ρ é o peso específico do material, A_i é a área da seção transversal, L_i é o comprimento da i-ésima barra da estrutura, u_j é o deslocamento do j-ésimo nó e n_a e n_d representam, respectivamente, a quantidade de barras e os graus de liberdade da estrutura. O problema está sujeito a restrições normalizadas de tensões σ em cada membro

$$\frac{\sigma_i}{\bar{\sigma}} - 1 \leq 0, \quad i = 1, \ldots, n_a, \tag{2.15}$$

onde $\bar{\sigma}$ é a tensão máxima admitida.

2.2.2 **Formulação para o caso dinâmico**

Em problemas de otimização estrutural, estabelecer limites para as frequências naturais de vibração é extremamente desejável pelo projetista. No entanto, pode não ser trivial, em uma análise preliminar, estabelecer os limites desejáveis para estas frequências. Neste sentido, uma possibilidade de conduzir esta análise de maneira mais ampla seria a consideração das frequências naturais como objetivos em um problema de otimização estrutural multiobjetivo onde as mesmas não fariam parte das restrições do problema. O resultado esperado pelo tomador de decisão será uma frente de Pareto onde será dada ao mesmo a opção de definição dos limites das frequências naturais de vibração. Esta análise possibilitará a reformulação do problema mono-objetivo onde as frequências serão consideradas novamente como restrições e, agora, como limites bem definidos.

Deste modo, o caso dinâmico possui como primeiro objetivo a minimização do peso da estrutura e como segundo objetivo, a maximização da frequência natural de vibração. O segundo objetivo foi estendido e possui três formulações diferentes que são: maximizar a menor frequência, maximizar a soma das frequências e maximizar a maior frequência. Tais formulações são apresentadas a seguir

$$\max \text{ mínimo } (\omega_j), \quad j = 1, \ldots, n_f, \tag{2.16}$$

$$\max \sum_{j=1}^{n_f} \omega_j \quad \text{e} \tag{2.17}$$

$$\max \text{ máximo } (\omega_j), \quad j = 1, \ldots, n_f, \tag{2.18}$$

onde ω_j é a j-ésima frequência natural de vibração e n_f é o número de frequências naturais da estrutura. O problema está sujeito a restrições normalizadas de tensões σ em cada membro e deslocamentos u de cada nó, descritos a seguir

$$\frac{\sigma_i}{\bar{\sigma}} - 1 \leq 0, \quad i = 1, \ldots, n_a \tag{2.19}$$

e

$$\frac{u_l}{\bar{u}} - 1 \leq 0, \quad l = 1, \ldots, n_d, \tag{2.20}$$

onde $\bar{u}$ é o deslocamento máximo admitido e n_d é o número de graus de liberdade da estrutura.

Dessa maneira, os experimentos e análises dos problemas de otimização apresentados no Capítulo 6 serão conduzidos para cada formulação de forma separada.

2.3 PROBLEMAS CLÁSSICOS DE OTIMIZAÇÃO ESTRUTURAL

Esta seção apresenta a descrição das treliças utilizadas nos experimentos conduzidos no Capítulo 6. São utilizadas 6 estruturas divididas da seguinte maneira: (i) as treliças de 10, 25, 60, 72 e 942 barras são utilizadas no segundo conjunto de experimentos (caso estático) e (ii) as treliças de 10, 25, 72 e 200 barras são utilizadas no terceiro conjunto de experimentos (caso dinâmico). Detalhes de cada treliça são apresentados nas subseções 2.3.1 a 2.3.6. Para cada uma são apresentadas também informações adicionais referentes à sua formulação em cada caso (estático e dinâmico), quando aplicável. É importante lembrar que todos os problemas analisados possuem dois objetivos. Os valores utilizados para a massa não-estrutural são baseados nos valores encontrados em [40] e [41].

2.3.1 **Treliça plana de 10 barras**

A treliça de 10 barras é um problema clássico ilustrado pela Fig. 6 e amplamente estudada [42, 43, 14, 15]. O objetivo é minimizar o peso da estrutura. As variáveis de projeto são as áreas da seção transversal dos dez elementos. Para o caso contínuo, as áreas podem ser escolhidas entre 0.1 in^2 a 40.0 in^2. Para o caso discreto, as áreas são escolhidas dentro de um conjunto que contém 32 possibilidades (in^2): 1.62, 1.80, 1.99, 2.13, 2.38, 2.62, 2.93, 3.13, 3.38, 3.47, 3.55, 3.63, 3.88, 4.22, 4.49, 4.59, 4.80, 4.97, 5.12, 5.74, 7.97, 11.50, 13.50, 14.20, 15.50, 16.90, 18.80, 19.90, 22.00, 26.50, 30.00, 33.50.

A treliça está sujeita a restrição de tensão normal máxima $\bar{\sigma}$ limitada em ± 25 ksi. A densidade do material $\rho = 0.1$ lb/in^3, o módulo de elasticidade $E = 10^4$ ksi e as cargas verticais descendentes de 100 $kips$ são aplicadas nos nós 2 e 4. Para o caso dinâmico, uma massa não-estrutural de 1000 lb é adicionada em cada nó livre da estrutura e as restrições de deslocamento são limitadas em 2 in nas direções x e y.

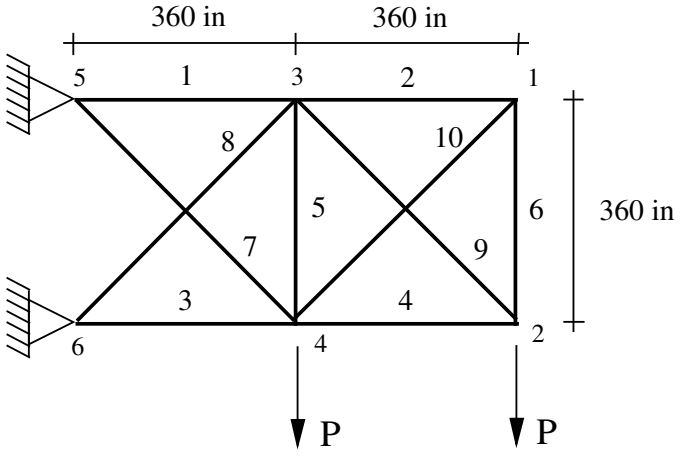


Figura 6 – Treliça plana de 10 barras.

2.3.2 **Treliça espacial de 25 barras**

A treliça de 25 barras, ilustrada conforme a Fig. 7, é um problema largamente estudado na literatura [42, 44, 14, 15]. As variáveis de projeto são as áreas das seções transversais agrupadas em 8 grupos conforme Tabela 1, a fim de manter a simetria da estrutura.

Tabela 1 – Agrupamento para a treliça de 25 barras.

Grupo	Conectividade
A_1	1-2
A_2	1-4, 2-3, 1-5, 2-6
A_3	2-5, 2-4, 1-3, 1-6
A_4	3-6, 4-5
A_5	3-4, 5-6
A_6	3-10, 6-7, 4-9, 5-8
A_7	3-8, 4-7, 6-9, 5-10
A_8	3-7, 4-8, 5-9, 6-10

Para o caso contínuo as variáveis de projeto variam entre 0.1 a 3.4 in^2. Para o caso discreto, as áreas das seções transversais devem ser escolhidas dentro de um conjunto com 30 opções (in^2): 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0, 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.8, 3.0, 3.2, 3.4.

A restrição de tensão em cada barra tem seu valor limitado em ± 40 ksi. A densidade do material $\rho = 0.1$ lb/in^3 e o módulo de elasticidade $E = 10^4$ ksi. O carregamento aplicado sobre a estrutura é apresentado na Tabela 2. Para o caso dinâmico, uma massa não-estrutural de 100 lb é adicionada nos nós 1 e 2 da estrutura e restrições de deslocamento

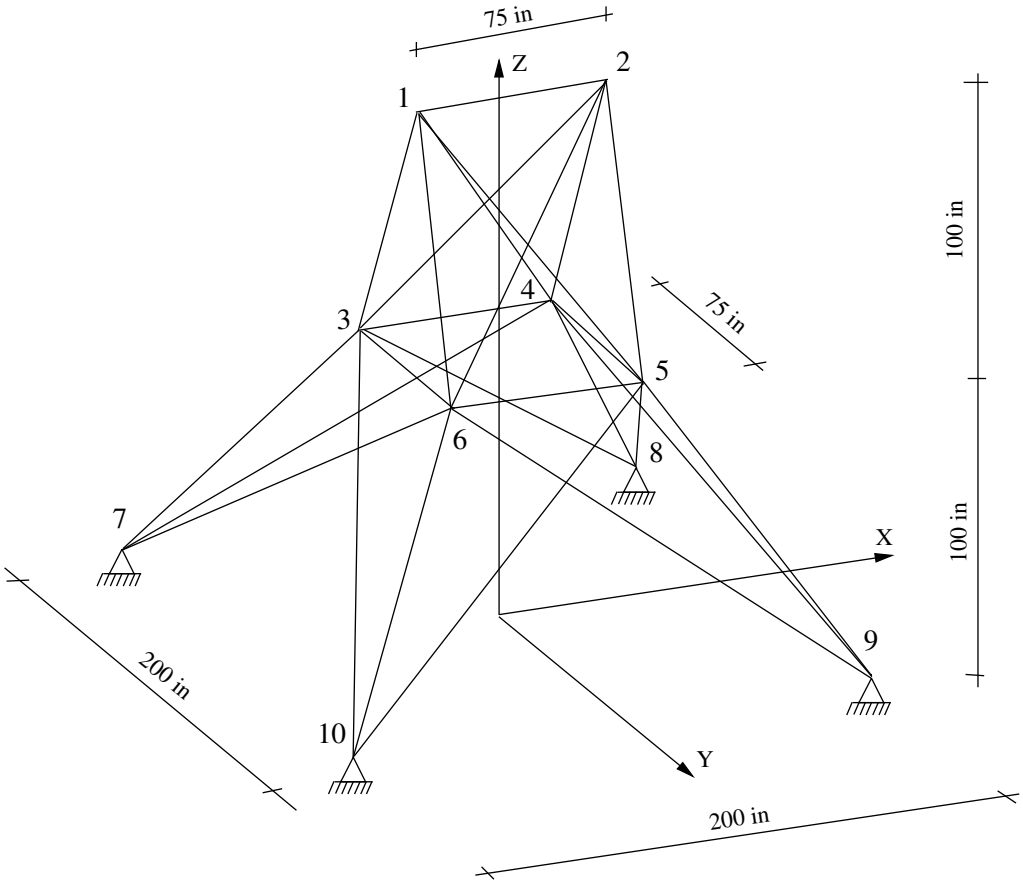


Figura 7 – Treliça espacial de 25 barras.

também nos nós 1 e 2 limitadas a 0.35 in nas direções x e y.

Tabela 2 – Carregamento para a treliça de 25 barras (em $kips$).

Nó	F_x	F_y	F_z
1	1	-10.0	-10.0
2	0	-10.0	-10.0
3	0.5	0	0
6	0.6	0	0

2.3.3 **Treliça plana de 60 barras**

A treliça de 60 barras [14, 15], ilustrada pela Fig. 8, possui raio externo do anel que vale 100 in e raio interno de 90 in. As 60 barras da estrutura são agrupadas em 25 grupos conforme Tabela 3 e variam em um espaço de busca contínuo de 0.5 in^2 a 5 in^2. Para o caso discreto, o conjunto de áreas são (in^2): 0.5, 0.6, 0.7, 0.8, ..., 4.7, 4.8 e 4.9.

A treliça é submetida a três casos de carregamentos, observados na Tabela 4. A tensão máxima é de 10 ksi, o módulo de elasticidade $E = 10^4$ ksi e a densidade do material

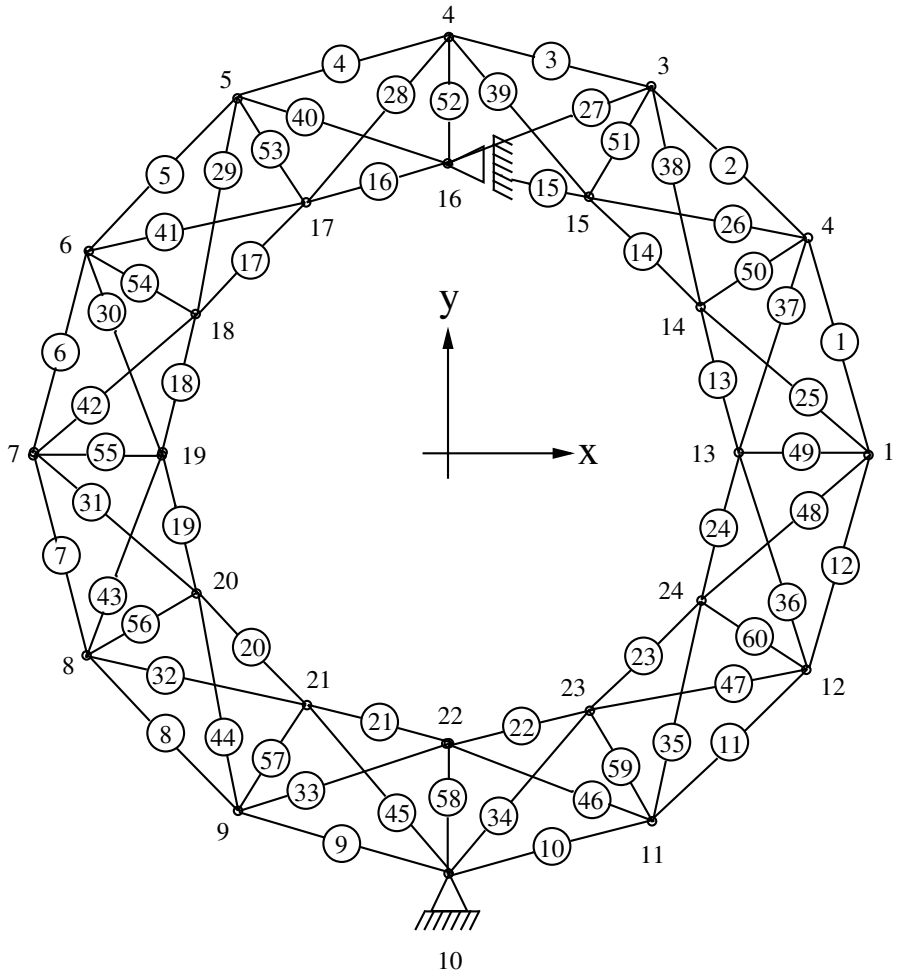


Figura 8 – Treliça plana de 60 barras.

$\rho = 0.1\ lb/in^3$. O carregamento aplicado sobre a estrutura é apresentado na Tabela 4.

2.3.4 **Treliça espacial de 72 barras**

A treliça de 72 barras ilustrada pela Fig. 9 e amplamente estudada [45, 14, 15] possui valores das variáveis de projeto para o caso contínuo variando de 0.1 à 2.5 in^2, organizados em 16 grupos conforme Tabela 5. Para o caso discreto, as variáveis devem ser escolhidas entre o conjunto de valores (in^2): 0.1, 0.2, 0.3, 0.4, 0.5, 0.6,0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0, 2.1, 2.2, 2.3, 2.4, 2.5.

As restrições do problema envolvem a tensão máxima permitida de cada barra limitada em $\pm 25\ ksi$. A densidade do material $\rho = 0.1\ lb/in^3$ e o módulo de elasticidade $E = 10^4\ ksi$. Para esse problema são definidos dois casos de carregamentos que podem ser observados na Tabela 6. Para o caso dinâmico, uma massa não-estrutural de 5000 lb é adicionada nos nós 1-4 da estrutura e as restrições de deslocamento são limitadas em 0.25 in nas direções x e y.

Tabela 3 – Agrupamento para a treliça de 60 barras.

Grupo	Barras	Grupo	Barras
A_1	49 ao 60	A_{14}	25 e 37
A_2	1 e 13	A_{15}	26 e 38
A_3	2 e 14	A_{16}	27 e 39
A_4	3 e 15	A_{17}	28 e 40
A_5	4 e16	A_{18}	29 e 41
A_6	5 e 17	A_{19}	30 e 42
A_7	6 e 18	A_{20}	31 e 43
A_8	7 e 19	A_{21}	32 e 44
A_9	8 e 20	A_{22}	33 e 45
A_{10}	9 e 21	A_{23}	34 e 46
A_{11}	10 e 22	A_{24}	35 e 47
A_{12}	11 e 23	A_{25}	36 e 48
A_{13}	12 e 24		

Tabela 4 – Carregamento para a treliça de 60 barras (em *kips*).

Carregamento	Nó	F_x	F_y
1	1	-10.0	0
	7	9.0	0
2	15	-8.0	3.0
	18	-8.0	3.0
3	22	-20.0	10.0

2.3.5 **Treliça plana de 200 barras**

A treliça de 200 barras tem uma estrutura de 77 nós conforme Fig. 10 e suas 200 barras são agrupadas em 29 grupos conforme Tabela 7. O módulo de elasticidade E $= 30 \times 10^4$ ksi e a densidade do material $\rho = 0.283$ lb/in^3. As tensões máxima de cada elemento é limitada em 10 ksi para membros com tração e compressão. Os limites inferior e superior das variáveis de projeto para o caso contínuo são de 0.1 in^2 a 20 in^2, respectivamente. Para o caso discreto, o conjunto de variáveis de projeto variam de (in^2): $0.1, 0.15, 0.2, 0.25, 0.3, \ldots, 20$.

Três casos de carregamento são considerados: (i) 1.0 $kips$ atuando na direção x nos nós 1, 6, 15, 20, 29, 34, 43, 48, 57, 62 e 71; (ii) -10 $kips$ na direção y nos nós 1, 2, 3, 4, 5, 6, 8, 10, 12, 14, 15, 16, 17, 18, 19, 20, 22, 24, 26, 28, 29, 30, 31, 32, 33, 34, 36, 38, 40, 42, 43, 44, 45, 46, 47, 48, 50, 52, 54, 56, 57, 58, 59, 60, 61, 62, 64, 66, 68, 70, 71, 72, 73, 74 e 75 e (iii) os casos de carregamentos (i) e (ii) agindo simultaneamente. Para o caso dinâmico, uma massa não-estrutural de 220.458 lb é adicionada nos nós 1-5 da estrutura.

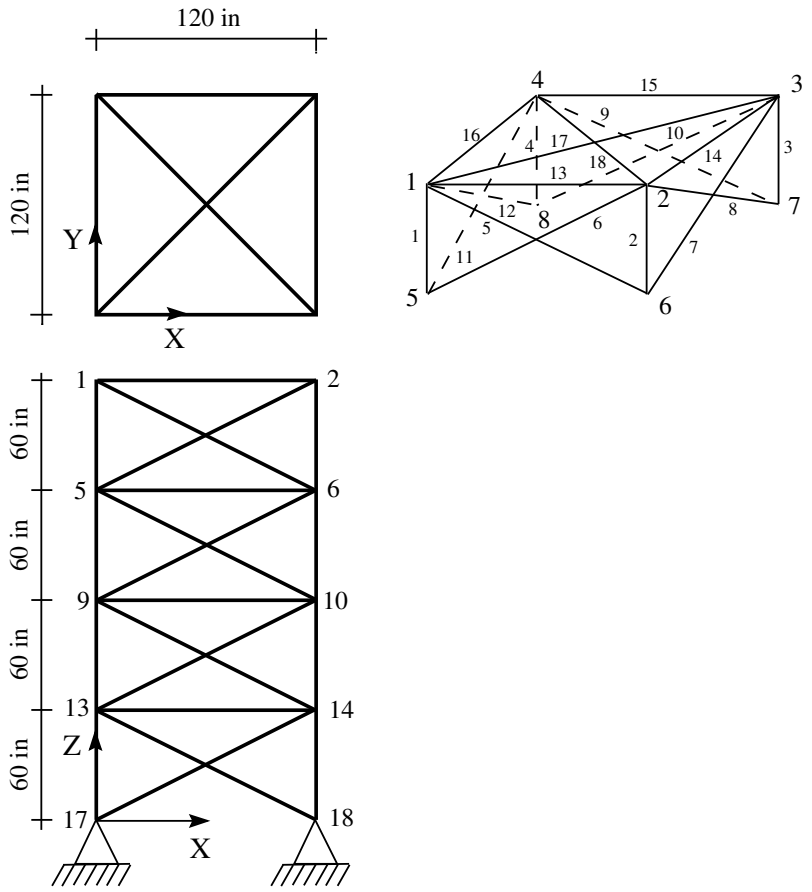


Figura 9 – Treliça espacial de 72 barras.

Tabela 5 – Agrupamento para a treliça de 72 barras.

Grupo	Barras
A_1	1, 2, 3 e 4
A_2	5, 6, 7, 8, 9, 10, 11 e 12
A_3	13, 14, 15 e 16
A_4	17 e 18
A_5	19, 20, 21 e 22
A_6	23, 24, 25, 26, 27, 28, 29 e 30
A_7	31, 32, 33 e 34
A_8	35 e 36
A_9	37, 38, 39 e 40
A_{10}	41, 42, 43, 44, 45, 46, 47 e 48
A_{11}	49, 50, 51 e 52
A_{12}	53 e 54
A_{13}	55, 56, 57 e 58
A_{14}	59, 60, 61, 62, 63, 64, 65 e 66
A_{15}	67, 68, 69 e 70
A_{16}	71 e 72

Tabela 6 – Carregamento para a treliça de 72 barras (em *kips*).

Carregamento	Nó	F_x	F_y	F_z
1	1	5	5	-5
2	1	0	0	-5
	2	0	0	-5
	3	0	0	-5
	4	0	0	-5

Tabela 7 – Agrupamento para a treliça de 200 barras.

Grupo	Barras
A_1	1, 2, 3 e 4
A_2	5, 8, 11, 14 e 17
A_3	19, 20, 21, 22, 23 e 24
A_4	18, 25, 56, 63, 94, 101, 132, 139, 170 e 177
A_5	26, 29, 32, 35 e 38
A_6	6, 7, 9, 10, 12, 13, 15, 16, 27, 28, 30, 31, 33, 34, 36 e 37
A_7	39, 40, 41 e 42
A_8	43, 46, 49, 52 e 55
A_9	57, 58, 59, 60, 61 e 62
A_{10}	64, 67, 70, 73 e 76
A_{11}	44, 45, 47, 48, 50, 51, 53, 54, 65, 66, 68, 69, 71, 72, 74 e 75
A_{12}	77, 78, 79 e 80
A_{13}	81, 84, 87, 90 e 93
A_{14}	95, 96, 97, 98, 99 e 100
A_{15}	102, 105, 108, 111 e 114
A_{16}	82, 83, 85, 86, 88, 89, 91, 92, 103, 104, 106, 107, 109, 110, 112 e 113
A_{17}	115, 116, 117 e 118
A_{18}	119, 122, 125, 128 e 131
A_{19}	133, 134, 135, 136, 137 e 138
A_{20}	140, 143, 146, 149 e 152
A_{21}	120, 121, 123, 124, 126, 127, 129, 130, 141, 142, 144, 145, 147, 148, 150 e 151
A_{22}	153, 154, 155 e 156
A_{23}	157, 160, 163, 166 e 169
A_{24}	171, 172, 173, 174, 175 e 176
A_{25}	178, 181, 184, 187 e 190
A_{26}	158, 159, 161, 162, 164, 165, 167, 168, 179, 180, 182, 183, 185, 186, 188 e 189
A_{27}	191, 192, 193 e 194
A_{28}	195, 197, 198 e 200
A_{29}	196 e 199

2.3.6 **Treliça espacial de 942 barras**

O último experimento refere-se à treliça de 942 barras, ilustrada pela Fig. 11 [14, 15]. As 942 barras são agrupadas em 59 variáveis de projeto e um caso de carregamento é considerado, com cargas verticais na direção z de -3 *kips*, -6 *kips* e -9 *kips* em cada

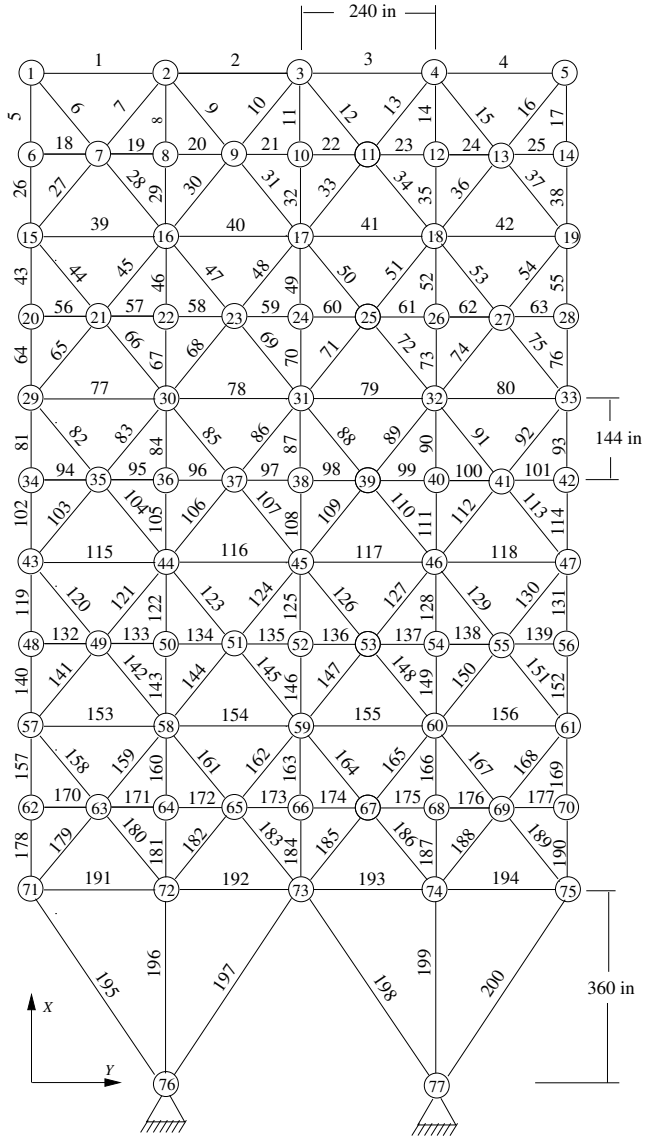


Figura 10 – Treliça plana de 200 barras.

nó no primeiro, segundo e terceiro setores, respectivamente. Cargas laterais na direção y de 1 *kips* em todos os nós da torre e cargas laterais na direção x de 1.5 *kips* e 1 *kips* em cada nó à esquerda e à direita da torre, respectivamente.

Para o caso contínuo, os limites das variáveis de projeto variam de 1 à 200 in^2. E para o caso discreto, as variáveis de projeto são escolhidas à partir de valores inteiros no mesmo intervalo. As restrições incluem tensão normal máxima de ± 25 *ksi* para todas as barras. O módulo de elasticidade $E = 10^4$ *ksi* e a densidade do material $\rho = 0.1$ lb/in^3.

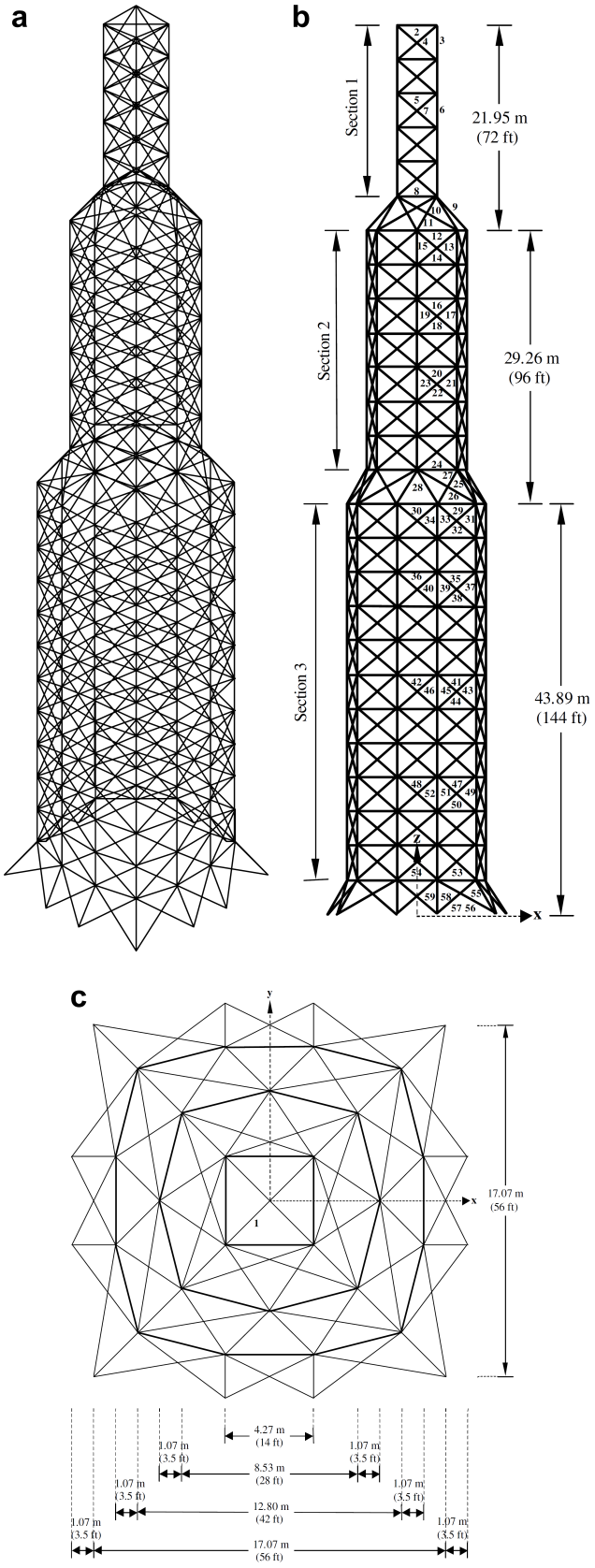


Figura 11 – Treliça espacial de 942 barras.

3 INTELIGÊNCIA DE ENXAMES

A vida em sociedade oferece mais chances de sobrevivência pois facilita a caça e a coleta de alimentos, aumenta a probabilidade de acasalamento, reduz a possibilidade de ataque por predadores, entre outros. Esse comportamento é encontrado em pássaros, peixes, formigas, abelhas e em outros tipos de insetos, e pode ser chamado de comportamento de enxames [46]. Tais comportamentos inspiraram pesquisadores no final da década de 80 no desenvolvimento de ferramentas computacionais para a solução de problemas de coordenação e controle de robôs [47, 48].

Com base nisso, surgiu o termo Inteligência de Enxames (*Swarm Intelligence* - SI), que é um campo emergente da Inteligência Artificial (*Artificial Intelligence* - AI) de inspiração biológica baseada nos modelos de comportamento sociais de formigas, abelhas e pássaros. Os sistemas SI consistem em uma população de partículas que interagem localmente umas com a outras e com seu ambiente. As partículas seguem regras muito simples e as interações entre essas partículas levaram ao surgimento de um comportamento global "inteligente". As técnicas mais conhecidos na SI são a otimização por colônia de formigas [49], algoritmos de colônia de abelhas [50], algoritmo por saltos de rãs [51], algoritmos de coleta de alimentos por bactérias [52] e otimização por enxame de partículas, que por sua vez, será o foco deste estudo.

3.1 OTIMIZAÇÃO POR ENXAME DE PARTÍCULAS

O algoritmo de Otimização por Enxame de Partículas (*Particle Swarm Optimization* - PSO) surgiu a partir de estudos de Russell Elberhart e James Kennedy [24] em 1995, com o objetivo de tratar problemas oriundos de funções contínuas não lineares [53]. O PSO é um algoritmo populacional inspirado no comportamento social de bando de pássaros na procura por alimento e tem mostrado ser eficiente para uma variedade de problemas de otimização [54, 55, 56, 57, 58]. Em engenharia estrutural, o PSO tem sido aplicado com sucesso em diferentes tipos de problemas [59, 60, 61, 62, 63, 64, 65, 66, 67, 68].

O algoritmo PSO possui boa capacidade de encontrar soluções sub-ótimas em uma quantidade razoável de tempo, é de fácil implementação e possui uma quantidade pequena de parâmetros. Entretanto, mesmo tendo os benefícios mencionados, o PSO padrão pode apresentar uma convergência prematura [69, 70], assim como outros algoritmos bio-inspirados. Dessa maneira, melhorar a capacidade de exploração do PSO tem sido um tópico de pesquisa ativo nos últimos anos [71].

O PSO é um algoritmo composto por uma população (chamada de enxame) formada por indivíduos (chamados de partículas) que se movimentam em um espaço R^n (onde n é a dimensão do espaço). Dentro deste espaço, cada partícula i corresponde a uma possível solução, representada por sua posição x_i e um valor de velocidade v_i, para um

dado problema. Cada partícula tem dois tipos de conhecimento: o social e o cognitivo. O conhecimento social indica que a partícula tem a capacidade de obter "sabedoria" observando outras partículas que pertencem ao seu grupo. E o conhecimento cognitivo, diz respeito à capacidade em obter sua própria experiência ao longo do tempo e saber qualificá-la como sendo boa ou ruim.

Sendo assim, duas variáveis denominadas $gBest$ e $pBest$ são adotadas. O $gBest$ representa a melhor posição global entre todas as partículas do enxame. E o $pBest$ é a melhor posição da i-ésima partícula. Os vetores de velocidade e posição da partícula i são armazenados durante o processamento do algoritmo em um tempo t e utilizados para a atualização do enxame no tempo $t+1$ usando as seguintes equações [24]

$$v_j^{(i)}(t+1) = v_j^{(i)}(t) + c_1 \cdot r_1(x_{pBest}^{(i)} - x_j^{(i)}) + c_2 \cdot r_2(x_{gBest} - x_j^{(i)}) \tag{3.1}$$

$$x_j^{(i)}(t+1) = x_j^{(i)}(t) + v_j^{(i)}(t+1) \tag{3.2}$$

onde $v_j^{(i)}$ e $x_j^{(i)}$ representam a velocidade e a posição atual da j-ésima variável de projeto da i-ésima partícula, respectivamente. c_1 e c_2 são coeficientes que controlam a influência da informação cognitiva e social, respectivamente, e r_1 e r_2 são dois valores aleatórios gerados com distribuição uniforme entre 0 e 1. A Fig. 12 apresenta um esquema das partículas agindo em um único mínimo global.

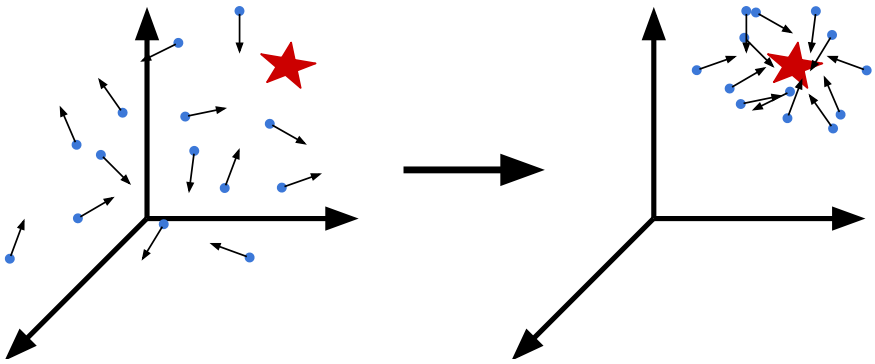


Figura 12 – Esquema das partículas agindo em um único mínimo global.

3.1.1 **Parâmetros**

Existem alguns parâmetros que precisam ser ajustados em um algoritmo PSO. Os fatores de aprendizagem c_1 e c_2 da Equação 5.1 (também conhecidos como parâmetros de confiança), podem mudar radicalmente o comportamento do algoritmo, provocando uma possível instabilidade. Eles normalmente são iguais variando entre [0, 4].

A velocidade v das partículas é limitada através de uma velocidade máxima denominada de $vmax$. Ela determina a variação máxima que uma partícula pode tomar durante uma iteração. Se $vmax$ é grande, facilita a exploração global. Por outro lado, se

vmax for muito pequena facilitará a exploração local. Experimentos utilizando *vmax* fixa foram conduzidos por Shi e Eberhart [72] e observou-se que o ajuste de velocidade máxima pode ser eliminado pois apresenta problemas: a velocidade máxima ideal é particular do problema e nenhuma regra específica é conhecida.

3.1.2 **Topologia das partículas**

Outro componente importante que influencia diretamente no desempenho do algoritmo é a topologia de comunicação entre as partículas. A topologia de comunicação determina como as partículas do enxame trocam informações. As principais topologias utilizadas são: global e local. Na topologia global (Fig. 13(a)), o enxame está organizado de modo que todas as partículas estejam conectadas entre si. A topologia global proporciona uma convergência mais rápida, visto que a informação da melhor posição é disseminada rapidamente entre todas as partículas do enxame. Em contrapartida, ela pode não garantir a qualidade da solução obtida, pois as partículas podem prender-se em mínimos locais.

Na topologia local (Fig. 13(b)), o enxame está organizado em formato de anel e cada partícula possui geralmente dois vizinhos. Embora a troca de informação entre as partículas seja mais lenta do que a global, este tipo de topologia provê uma melhor qualidade de soluções em comparação com a topologia global. Outros tipos de topologia podem ser utilizadas, tais como: árvore [73], roda [74], grafos [2], *Von Neumann* [75], entre outros.

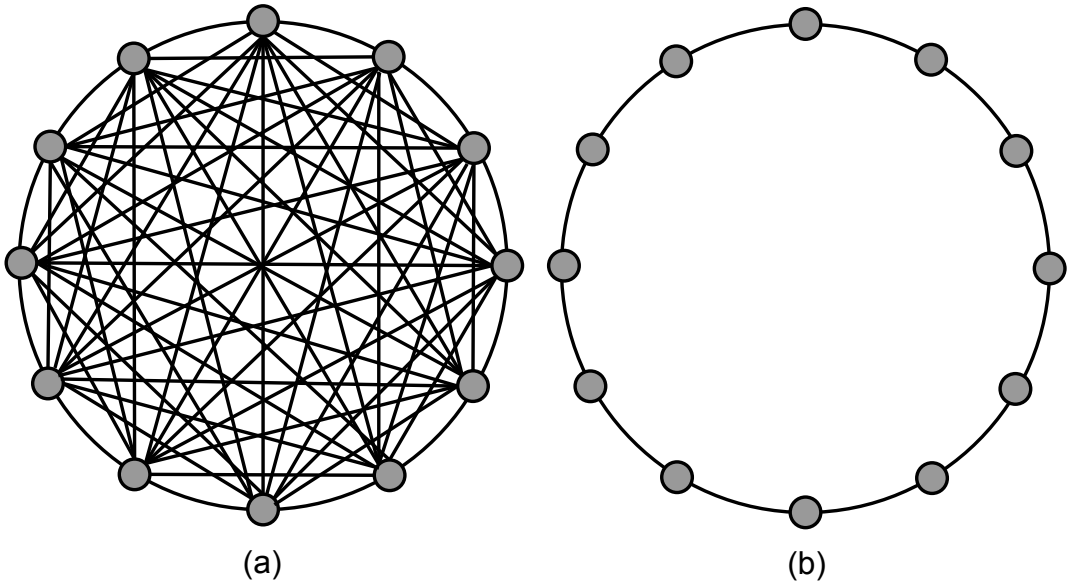


Figura 13 – Topologia das partículas: (a) Topologia global (b) Topologia local, adaptada de [2].

3.2 TRATAMENTO DE RESTRIÇÕES

A maioria dos problemas de engenharia contém restrições. Métodos adequados para o tratamento dessas restrições são de grande importância. Os EAs podem ser vistos como algoritmos de busca que não possuem técnicas explícitas para o tratamento de restrições, já que, em sua forma original, não incorporam qualquer mecanismo nessa concepção. Devido a isso, vários autores têm proposto uma variedade de técnicas de tratamento de restrições explicitamente projetadas para EAs [76, 77, 78].

A abordagem mais comum na comunidade de EAs para lidar com restrições é o uso de funções de penalização [79]. As funções de penalização foram originalmente propostas por Courant [80] na década de 40 e posteriormente expandida por Carroll e Fiacco [81, 82]. A ideia principal é transformar problemas de otimização com restrições em problemas sem restrições adicionando (ou subtraindo) um certo valor na função objetivo baseado na quantidade de violação das restrições presente em uma determinada solução. Dessa forma, os métodos de penalização permitem a resolução de problemas com restrições por métodos tipicamente utilizados em problemas sem restrições. Tais funções, embora sejam de aplicação bastante geral, exigem um considerável conhecimento do domínio e experimentação em cada aplicação específica, a fim de ser eficaz.

Vários pesquisadores têm estudado heurísticas sobre a concepção dessas funções de penalização. Provavelmente, o mais conhecido desses estudos é o realizado por Richardson *et al.* em [83]. A forma geral de uma função de penalização, segundo Mezura-Montes e Coello [84], pode ser definida como

$$F(\mathbf{x}) = f(\mathbf{x}) + penal(\mathbf{x}) \tag{3.3}$$

onde $F(\mathbf{x})$ é a função aptidão, $f(\mathbf{x})$ é a função objetivo e $penal(\mathbf{x})$ é a função de penalização que pode ser calculada como

$$penal(\mathbf{x}) = \sum_{j=1}^{n_g} r_j \cdot max(0, g_j(\mathbf{x}))^2 + \sum_{k=1}^{n_h} c_k \cdot |h_k(\mathbf{x})| \tag{3.4}$$

onde r_j e c_k são constantes positivas chamadas de fatores de penalização, $g_j(x)$ são as restrições de desigualdade, $h_k(x)$ são as restrições de igualdade, n_g é o número de restrições de desigualdade e n_h é o número de restrições de igualdade. A função de penalização $penal(\mathbf{x})$ é dependente de fatores que exigem um cuidadoso ajuste para determinar a severidade das sanções a serem aplicadas. Os valores dos fatores de penalização são altamente dependentes do problema [84].

As técnicas de penalização podem ser classificadas como multiplicativas e aditivas. No caso multiplicativo, um fator positivo de penalidade é introduzido de modo a amplificar o valor da função objetivo de um indivíduo infactível em um problema de minimização. No caso aditivo, uma função de penalidade é adicionada à função objetivo a fim de definir

o valor da aptidão de um indivíduo infactível. As técnicas de penalização podem ainda ser divididas em técnicas de interior e exterior [85]. A ideia, em ambos os casos, é amplificar o valor da função aptidão de um indivíduo infactível.

Essas técnicas apresentam muitas variações na maneira de se definir os fatores de penalização. Elas podem ser classificadas como **estática**, **dinâmica** ou **adaptativa**. As penalidades do tipo estática permanecem constantes durante todo o processo evolutivo. A função objetivo penalizada seria, então, a função objetivo da solução candidata mais uma "multa" (para problemas de minimização). As penalizações dinâmicas aparecem como alternativa de suprir a dificuldade do usuário em determinar os coeficientes de penalização, uma desvantagem das penalizações estáticas. Nas penalizações dinâmicas, o período ou instante do processo evolutivo está diretamente envolvido com os coeficientes de penalização.

Nas funções com penalização adaptativa, os parâmetros de penalização são atualizados ao longo do processo de evolução de acordo com informações coletadas da população. Assim, eles podem ser alterados baseando-se nas gerações, no grau de violação das restrições do problema, na função objetivo, entre outros. Uma técnica adaptativa, conhecida como APM, será utilizada neste trabalho para o tratamento das restrições e é descrita a seguir.

3.2.1 Método de Penalização Adaptativa (APM)

Um Método de Penalização Adaptativa (*Adaptive Penalty Method* - APM) foi originalmente introduzido por Barbosa & Lemonge [86]. O método usa informações da população, como a média da função objetivo e o nível de violação de cada restrição durante a evolução. Ao usar o APM, a função aptidão pode ser escrita como

$$F(x) = \begin{cases} f(x), & \text{se } x \text{ é factível} \\ \bar{f}(x) + \sum_{j=1}^{n_c} k_j \; viol_j(x), & \text{caso contrário,} \end{cases} \tag{3.5}$$

onde n_c é o número de restrições, k_j é o parâmetro de penalização, $viol_j(x)$ é a violação de cada restrição e $\bar{f}(x)$ é definido como

$$\bar{f}(x) = \begin{cases} f(x), & \text{se} \quad f(x) > \langle f(x) \rangle \\ \langle f(x) \rangle, & \text{se} \quad f(x) \leq \langle f(x) \rangle, \end{cases} \tag{3.6}$$

onde $\langle f(x) \rangle$ é a média dos valores da função objetivo da população atual.

O parâmetro de penalização k_j é definido em cada iteração como

$$k_j = | \langle f(x) \rangle | \frac{\langle \delta_j(x) \rangle}{\sum_{l=1}^{pop} [\langle \delta_l(x) \rangle]^2}, \tag{3.7}$$

onde $\langle \delta_j(x) \rangle$ é a violação da j-ésima restrição, calculados sobre a população atual e *pop* é a quantidade de violações de todos os indivíduos da população.

3.2.2 **Restrições de cardinalidade**

É muito comum em configurações de estruturas reticuladas o desejo de agrupar as barras para manter a simetria, facilitar a fabricação, transporte, armazenagem, execução e conferência. Entretanto, descobrir qual o melhor agrupamento, e este em uma solução otimizada, não é uma tarefa trivial. A fim de explorar as vantagens desejáveis durante as configurações de projeto, pode ser interessante explorar os efeitos arquitetônicos que ligam os membros de uma estrutura em grupos. Por exemplo, um projetista pode definir que no máximo m áreas da seção transversal diferentes sejam adotadas em toda a estrutura, o que leva à necessidade de organizar as barras em apenas m grupos.

A cardinalidade pode ser entendida como uma restrição adicional com o intuito de definir o agrupamento de barras da estrutura que apresentam o mesmo valor de área da seção transversal. Os valores de cardinalidade devem ser escolhidos com base em critérios definidos pelo projetista, como facilidade, rapidez e economia na produção das barras (visto que mais barras terão uma mesma área), armazenagem, execução e posterior verificação e liberação da estrutura.

Como característica dessas escolhas, é fato que quanto menor o valor da cardinalidade adotada (número menor de seções transversais distintas), maior será o peso final da estrutura, visto que o algoritmo terá menos opções de busca. Sendo assim, é proposto que se analise uma mesma estrutura com diferentes valores de cardinalidade e apresente os resultados ao projetista em um gráfico com valores de número de cardinalidade x peso. De posse deste gráfico e com os critérios anteriormente mencionados, o projetista poderá proceder à tomada de decisão com relação à escolha do número de áreas que lhe satisfaça.

Barbosa & Lemonge [87, 88, 89] introduziram uma codificação especial para um cromossomo (codificação binária) através do conceito de restrições de cardinalidade para descobrir o agrupamento ótimo das barras definindo um número m máximo de diferentes áreas da seção transversal a serem usadas na estrutura otimizada.

Assim, o problema de otimização deve considerar restrições adicionais que exijam não mais do que m ($m \leq n_{card}$) diferentes áreas da seção transversal devem ser utilizadas, isto é

$$A_i \in C_m = \{S_1, S_2, \ldots, S_m\}, \quad i = 1, \ldots, n_{card}, \tag{3.8}$$

onde as áreas da seção transversal $S_j, j = 1, \ldots, m$ são desconhecidas, mas pertencem ao conjunto $S = \{A_1, \ldots, A_{n_{card}}\}$.

Estas restrições adicionais são definidas como restrições de cardinalidade e o problema de otimização padrão (sem restrição de cardinalidade) é "recuperado" quando $m = n_{card}$.

3.3 TRABALHOS RELACIONADOS

Diversos trabalhos podem ser encontrados na literatura que utilizam meta-heurísticas em otimização estrutural e/ou restrições de cardinalidade em sua formulação. Alguns desses trabalhos serão abordados em uma breve revisão da literatura, apresentada à seguir.

Barbosa & Lemonge [26] propuseram uma codificação especial de um cromossomo em um algoritmo genético com o objetivo de estabelecer uma estratégia para descobrir o agrupamento ideal de membros. O conjunto de valores distintos das variáveis de projeto deve ser um subconjunto - de cardinalidade não superior a um determinado valor - de um conjunto maior de itens disponíveis. Bons resultados foram encontrados nos experimentos numéricos usando codificação binária e operadores genéticos.

Barbosa *et al.* [87] propuseram um algoritmo genético para a solução de problemas de otimização estrutural com restrições de cardinalidade e encontraram resultados satisfatórios utilizando variáveis discretas e/ou contínuas e um esquema de penalização adaptativa. Um outro estudo utilizando cardinalidade foi apresentado por Lemonge *et al.* [34] em problemas de otimização dimensional de estruturas reticuladas e bons resultados foram encontrados nos experimentos numéricos realizados.

Wei *et al.* [90] propuseram a otimização dimensional e de forma com restrições de frequências para melhorar o desempenho dinâmico das estruturas. Um Algoritmo Genético híbrido denominado *Niche Hybrid Parallel Genetic Algorithm* (NHPGA) foi proposto para reduzir significativamente o custo computacional e melhorar a qualidade da solução. O NHPGA é uma mistura das vantagens da computação paralela, busca *simplex* e um AG com uma técnica de nicho. Vários exemplos de otimização de treliças demonstraram que o NHPGA pode reduzir significativamente o tempo de computação e alcançar soluções de maior qualidade.

Um estudo de otimização topológica de estruturas reticuladas com restrições de frequência foi apresentado por Liu *et al.* [91]. Devido à natureza discreta de tal problema, o projeto ótimo não pode ser obtido por busca contínua. Para lidar com tal dificuldade, um algoritmo genético com agrupamento automático e com melhorias no operador de *crossover* e na função de penalização é aplicado para otimizar a estrutura. Dois exemplos demonstraram o bom desempenho da abordagem proposta.

Kripka *et al.* [92] apresentaram uma aplicação de estratégias de otimização para o custo de vigas em edifícios de concreto armado. Como o número de vigas com tamanhos diferentes deve ser limitado devido a razões práticas, restrições de cardinalidade também são consideradas com o objetivo de identificar o agrupamento ideal dos elementos. Alguns resultados obtidos para diferentes números de grupos de membros são apresentados a fim de ilustrar a estratégia proposta.

Kaveh & Zolghadr [93] aplicaram um novo algoritmo denominado *Democratic*

Particle Swarm Optimization para a solução de problemas de otimização estrutural com restrições de frequências de vibração. Quatro problemas de minimização foram utilizados nos experimentos e algoritmo proposto apresentou desempenho superior quando comparado ao PSO padrão.

A otimização de forma e dimensionamento de estruturas treliçadas foi realizada por Kaveh & Javadi [94] utilizando um método híbrido denominado HRPSO: Busca Harmônica (HS), Otimização *Ray* (RO) e Otimização por Enxame de Partículas (PSO). Várias restrições de frequência são consideradas tornando a otimização um problema altamente não-linear. Problemas clássicos são utilizados na análise e os resultados demonstraram a eficiência e a robustez do método híbrido proposto, quando comparado com outros algoritmos da literatura.

Dois algoritmos de otimização inspirados em colônias de formigas (*Ant Colony Optimization* - ACO) foram propostos por Angelo *et al.* [14] para a solução de problemas de otimização estrutural multi-objetivo com restrições de cardinalidade. Tal restrição é executada diretamente quando uma formiga "cria" uma solução candidata, enquanto as outras restrições são tratadas por um método de penalização adaptativa. Os resultados encontrados são comparados com soluções disponíveis na literatura e forneceram melhores resultados do que os obtidos quando nenhuma restrição de cardinalidade é considerada.

Um algoritmo de otimização baseado em ensino-aprendizagem (TLBO) modificado denominado de MS-TLBO é apresentado em Tejani *et al.* [95] para a otimização dimensional e de forma de treliças com restrições de frequências naturais. Os resultados revelaram que o MS-TLBO é mais eficiente em comparação com o original e com outros algoritmos TLBO encontrados na literatura.

Farshchin *et al.* [96] apresentaram uma técnica de otimização baseada em ensino-aprendizagem multi-classe (MC-TLBO) para a otimização estrutural com restrições de frequência. O MC-TLBO é aplicado em vários problemas de otimização de treliças com restrições de frequência e os resultados são comparados com os resultados de um algoritmo TLBO modificado e com outros métodos de otimização encontrados na literatura.

Tejani *et al.* [97] testaram três versões modificadas do algoritmo SOS (*Symbiotic Organisms Search*) para a aplicação em estruturas de engenharia submetidas à excitação dinâmica, que podem levar a vibrações indesejáveis. Seis treliças planas e espaciais foram submetidas à análise estrutural. Os resultados obtidos utilizando os métodos propostos são comparados com os obtidos usando outros métodos de otimização bem estabelecidos na literatura.

Uma análise de desempenho de um algoritmo GDE3 combinado com o APM em cinco problemas de otimização estrutural multiobjetivo com codificação real para as variáveis de projeto é apresentada em Vargas *et al.* [15]. Os resultados obtidos são

comparados aos encontrados na literatura, indicando o potencial do algoritmo proposto.

Kalayci *et al.* [98] desenvolveram um trabalho que apresenta uma abordagem baseado em um algoritmo artificial de colônia de abelhas com procedimentos de execução de viabilidade e tolerância à inviabilidade para resolver problemas de otimização de portfólio com restrições de cardinalidade. Experimentos computacionais confirmaram a eficácia da metodologia proposta.

A otimização dimensional e de forma de estruturas treliçadas sujeitas à restrições de frequência é abordada em Kaveh & Zolghadr [99]. No trabalho é utilizado um algoritmo multi-agente denominado *Tug of War Optimization* - TWO. Experimentos computacionais são realizados e os resultados são comparados com outros algoritmos de otimização e indicaram que o algoritmo TWO apresentou um bom desempenho em todos os problemas analisados.

Um novo algoritmo de Evolução Diferencial (DE) foi proposto em Ho-Huu *et al.* [18] para resolver problemas de otimização de dimensionamento e de forma de estruturas treliçadas com restrições de frequência. O método proposto, chamado ReDE, é uma nova versão do algoritmo DE com duas melhorias. Primeiro, a seleção da roda da roleta é utilizada para escolher membros para a fase de mutação, ao invés de seleção aleatória como no DE convencional. Segundo, uma técnica de seleção elitista é aplicada à fase de seleção, ao invés da seleção básica para melhorar a velocidade de convergência. A eficiência e a confiabilidade do algoritmo foram demonstradas através de cinco exemplos. Os resultados revelaram que o algoritmo proposto superou muitos métodos de otimização encontrados na literatura.

4 OTIMIZAÇÃO MULTIOBJETIVO

Dentro da área da Otimização, existem problemas que podem apresentar vários objetivos a serem alcançados ao invés de apenas um. Geralmente, esses objetivos são conflitantes entre si e, raramente, existe uma solução que seja ótima para todos eles simultaneamente. Essas são características comuns de problemas de Otimização Multiobjetivo (*Multiobjective Optimization Problems* - MOPs).

Ao contrário do que acontece comumente em problemas mono-objetivo, a resolução de um MOP fornece um conjunto de soluções que representam os melhores compromissos entre os objetivos. Neste caso, a tomada de decisão será de responsabilidade do analista, que deverá escolher uma entre as diversas soluções do conjunto de soluções disponíveis [100]. É importante mencionar novamente que os objetivos podem ser conflitantes [101, 102] e que o problema pode estar sujeito a restrições explícitas ou implícitas.

Na engenharia, a maioria dos problemas de decisão são problemas de otimização multiobjetivo [8]. Para tais problemas, os melhores resultados em todos os objetivos podem nem sempre ser alcançados, de modo que os tomadores de decisão precisem equilibrar e avaliar a importância de cada objetivo [103].

4.1 CONCEITOS BÁSICOS

Antes de apresentar o problema multiobjetivo, é conveniente definir alguns conceitos comuns a qualquer método de otimização. São eles:

- variáveis de decisão (de projeto): são as incógnitas a serem determinadas pela solução do modelo,
- função objetivo: é a função matemática que define a qualidade da solução em função das variáveis de decisão e
- restrições: são as limitações físicas do sistema.

Um Problema de Otimização Multiobjetivo com Restrições (*Constrained Multiobjective Optimization Problem* - CMOP) é composto por um conjunto de funções objetivos a serem otimizadas e um conjunto de restrições que devem ser satisfeitas para que a solução seja factível. O problema pode ser formulado como segue [104]

min

$$Z(\mathbf{x}) = (z_1(\mathbf{x}), \ldots, z_{n_{obj}}(\mathbf{x})), \tag{4.1}$$

sujeito a

$$
\begin{aligned}
&g_j(\mathbf{x}) \leq 0, \quad j = 1, \ldots, n_g, \\
&h_l(\mathbf{x}) = 0, \quad l = n_g + 1, \ldots, n_h \text{ e} \\
&a_i \leq x_i \leq b_i, \quad i = 1, \ldots, n_a,
\end{aligned}
$$

onde $\mathbf{x} = x_1, \ldots, x_{n_a}$ é um vetor n_a-dimensional de variáveis de projeto, $Z(\mathbf{x})$ é uma função que retorna um vetor de valores $Z : \mathbb{R}^{n_v} \rightarrow \mathbb{R}^{n_{obj}}$, $g(\mathbf{x})$ e $h(\mathbf{x})$ são as restrições de desigualdade e igualdade, respectivamente. As variáveis a e b são os limites inferior e superior de x e n_{obj}, n_g e n_h representam, respectivamente, o número de funções objetivo, restrições de desigualdade e restrições de igualdade.

O conceito de otimalidade na otimização multiobjetivo baseia-se na noção introduzida por Francis Ysidro Edgeworth em 1881 e posteriormente generalizada por Vilfredo Pareto em 1896. Na otimização multiobjetivo, emprega-se o conceito de dominância de Pareto para comparar duas soluções factíveis do problema. A imagem do conjunto de soluções no espaço dos objetivos é chamada de Frente de Pareto [105]. A principal meta na resolução de um MOP é encontrar a Frente de Pareto, ou pelo menos uma boa aproximação discreta dela, da qual o tomador de decisão irá escolher a solução que melhor atenda às suas preferências.

A otimização multiobjetivo trabalha com dois espaços: o espaço de variáveis e o dos objetivos. O espaço de variáveis é onde se faz a busca pelas soluções do problema, ou seja, é o domínio das variáveis do problema. O espaço dos objetivos é o espaço formado pela imagem do conjunto de soluções do problema. A Fig. 14 apresenta um exemplo de mapeamento feito pelas funções objetivo de um CMOP entre o espaço de variáveis tridimensional e um espaço dos objetivos bidimensional.

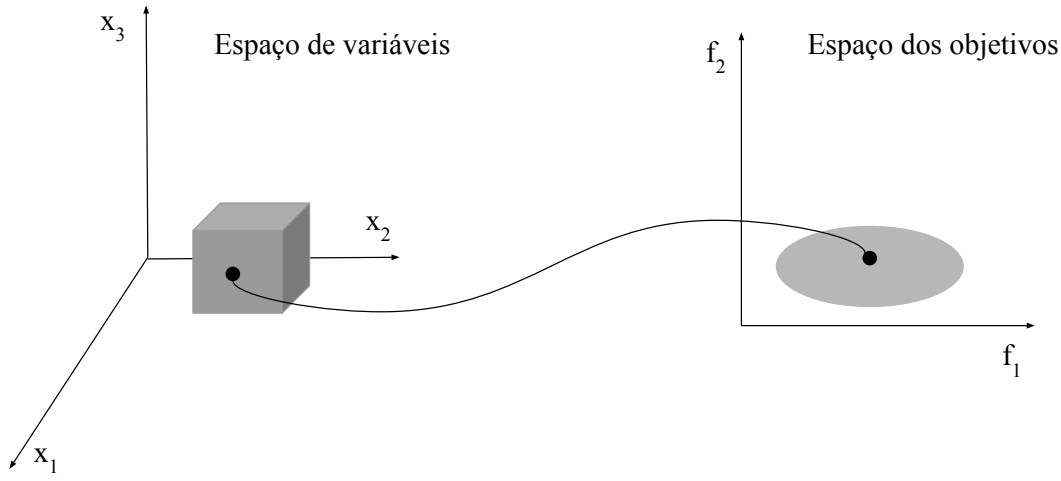


Figura 14 – Exemplo de um mapeamento de uma solução no espaço de variáveis tridimensional para sua imagem no espaço dos objetivos bidimensional, adaptada de [3].

Na existência de duas soluções s_1 e s_2, diz-se que s_1 domina s_2 ($s_1 \preceq s_2$) se as condições a seguir forem satisfeitas (para problemas de minimização)

- A solução s_1 é melhor ou igual a s_2 em todas as funções objetivo.
- A solução s_1 é estritamente melhor que s_2 em pelo menos uma função objetivo.

O conceito de dominância entre dois pontos é ilustrado na Fig. 15. Nota-se que A e B são soluções associadas a valores distintos de x, onde $B \preceq A$. O conjunto de soluções não-dominadas é chamado de conjunto Pareto-ótimo e representa a solução do problema. A fronteira de Pareto é o conjunto de valores das funções objetivo das soluções do conjunto Pareto-ótimo.

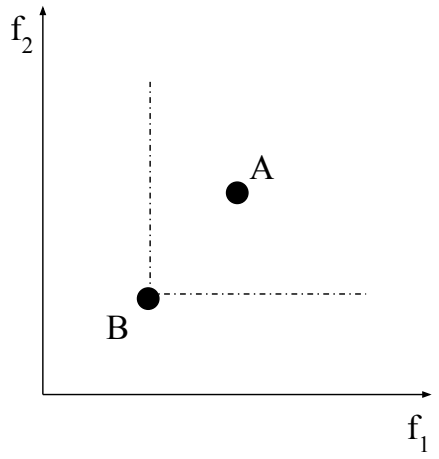


Figura 15 – Ilustração do conceito de dominância entre dois pontos para um problema de minimização, adaptada de [3].

Como a otimização multiobjetivo trabalha com esses dois espaços, é desejável que as soluções estejam adequadamente distribuídas em ambos. Dado que encontrar um conjunto de soluções adequadamente distribuídas é uma tarefa que pode consumir consideráveis recursos computacionais, faz-se necessário que tais soluções sejam obtidas de forma eficiente.

Na otimização multiobjetivo existem três metas importantes [3]: (i) obter um conjunto de soluções que esteja o mais próximo possível da Fronteira de Pareto; (ii) obter um conjunto de soluções com a maior diversidade possível, incluindo critérios como uniformidade no espalhamento ao longo da Fronteira de Pareto e (iii) alcançar as duas metas anteriores com a maior eficiência computacional possível. A Figura 16 ilustra as duas primeiras metas. Nota-se que a convergência e a diversidade podem ser conflitantes entre si e, portanto, ao avaliar o desempenho de um algoritmo é necessário usar métricas que considerem ambas as metas.

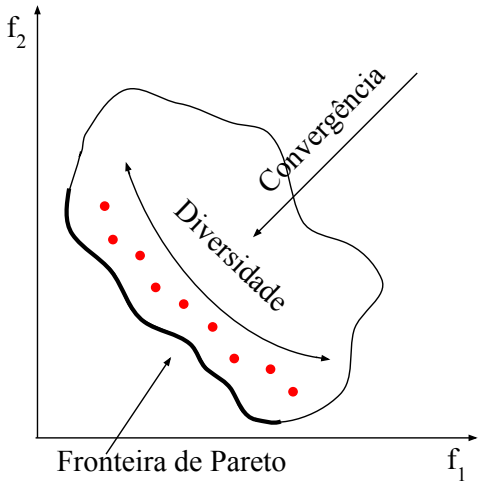


Figura 16 – Metas da otimização multiobjetivo, adaptada de [3].

4.2 ALGORITMOS EVOLUTIVOS MULTIOBJETIVO

Os Algoritmos Evolutivos Multiobjetivo (*Multiobjective Evolutionary Algorithm* - MOEAs) têm atraído um grande esforço de investigação durante as últimas décadas e ainda formam uma das áreas mais promissoras de investigação no campo da Computação Evolutiva [106]. Muitos pesquisadores dedicados à concepção e implementação de EAs para resolver problemas de otimização multiobjetivo têm mostrado excelentes resultados através da resolução de problemas *benchmark* multiobjetivos e problemas de engenharia [107, 108, 109, 110].

O primeiro MOEA desenvolvido foi proposto por Schaffer [111] em 1985 e é denominado VEGA (*Vector Evaluated Genetic Algorithm*). Este algoritmo é um AG modificado que avalia cada objetivo separadamente. Um dos problemas do VEGA é que as soluções na fronteira, em geral, possuem baixa diversidade. Alguns MOEAs tradicionalmente encontrados na literatura e com um número considerável de trabalhos desenvolvidos nas mais variadas aplicações, são apresentados a seguir.

4.2.1 **NSGA-II**

Um MOEA largamente estudado na literatura foi proposto por Deb *et al.* [28] e é conhecido como NSGA-II (*Nondominated Sorting Genetic Algorithm II*). A principal característica do algoritmo é a ordenação pela não dominância. As soluções são separadas em dominadas e não-dominadas. Aquelas não-dominadas são classificadas como sendo preferidas em relação às demais. O processo é repetido utilizando somente os indivíduos dominados até que toda a população seja classificada da mesma forma.

Além disso, no NSGA-II a população é dividida em d subconjuntos, denominado de *rank*. Cada *rank* corresponde aos indivíduos não-dominados da população que não

são dominados por nenhum outro *rank*. Assim, os indivíduos de *rank* 1 tem preferência de seleção sobre os demais. O mesmo acontece com os indivíduos de *rank* 2 que tem preferência de seleção sobre os demais, exceto em relação aos indivíduos de *rank* 1. Desse modo, indivíduos com *rank* menores tem mais chance de permanecerem na população e de serem pais de novos indivíduos. A Fig. 17 ilustra esse procedimento de seleção que atualmente é a base da maioria dos MOEAs.

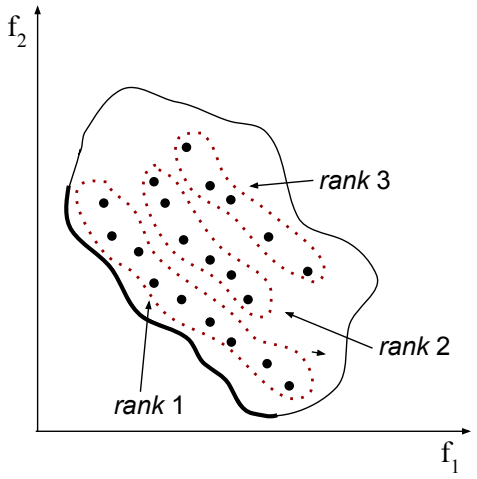


Figura 17 – Ordenação de Pareto proposta por [4].

Uma vez que a organização em *ranks* está completa, é calculada a distância de aglomeração (*crowding distance* - CD). A CD fornece uma estimativa da densidade de soluções em torno de determinado ponto. As soluções que têm os valores da função objetivo mais baixos e mais altos recebem valores de CD infinito para que elas sempre sejam selecionadas. A Fig. 18 ilustra o cálculo da *crowding distance* de um ponto i, que é uma estimativa do tamanho do maior cuboide que inclui i sem incluir qualquer outro ponto [5].

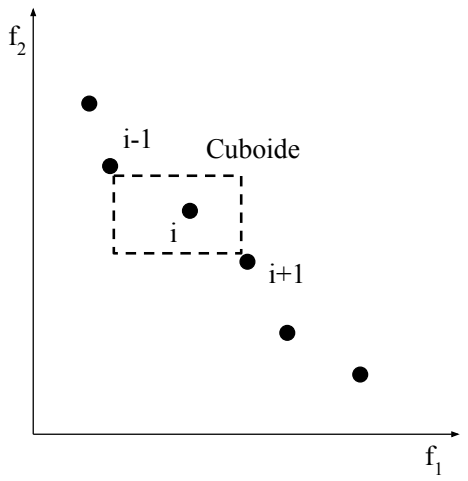


Figura 18 – *Crowding distance*, adaptada de [5].

O processo descrito acima é efetuado para cada função objetivo e irá ordenar cada indivíduo de acordo com sua CD em relação aos pontos vizinhos do mesmo *rank*. Esse operador permite que se tenha um melhor espalhamento dos resultados, evitando aglomerações de soluções em uma mesma sub-região do espaço de soluções viáveis. Serão selecionados os indivíduos do primeiro ou dos primeiros *ranks*, em ordem crescente. Contudo, caso eles estejam no mesmo nível, serão escolhidos os indivíduos com maior valor de CD.

O resultado desta classificação é um conjunto de alternativas não-dominadas entre si (pertencentes ao primeiro *rank*) e melhores que as demais opções (nos demais *ranks*) de acordo com os objetivos estabelecidos. As soluções do primeiro *rank* compõem o conjunto Pareto-ótimo, onde a "melhor" solução pode ser escolhida segundo critérios do tomador de decisão.

Na literatura são encontrados inúmeros trabalhos onde o NSGA-II é utilizado em diversas aplicações [112, 113, 114, 115].

4.2.2 MODE

Xue [116] propôs o algoritmo MODE (*Multiobjective Differential Evolution*), desenvolvido para o domínio contínuo (C-MODE) e discreto (D-MODE). No algoritmo, cada vetor alvo gera um vetor experimental que é inserido na população, independente da relação de dominância existente entre eles. Após isso, o mesmo esquema de seleção do NSGA-II é aplicado. O MODE usa informações de indivíduos não-dominados na construção de novos indivíduos, sempre que estes estão sendo produzidos a partir de indivíduos dominados. Isso provoca uma melhora significativa na qualidade das soluções.

Um conjunto de métricas de desempenho chamado de *Pareto Front Approximation Error* (PFAE) é introduzido para avaliar as soluções não-dominadas. Uma comparação entre o MODE e o *Strength Pareto Evolutionary Algorithm* (SPEA) [117] é apresentada pelos autores, onde o MODE apresentou desempenho superior nos problemas analisados.

4.2.3 DEMO

UM MOEA baseado no algoritmo DE foi proposto por Robic & Filipic [118], e ficou conhecido como DEMO (*Differential Evolution for Multiobjective Optimization*). O DEMO utiliza a relação de dominância na tomada de decisão da substituição do vetor alvo pelo experimental gerado da seguinte forma: o vetor que domina o outro, o substitui na população. Se os dois vetores são não-dominados entre si, então ambos permanecem na população que será submetida ao procedimento de seleção do NSGA-II.

Experimentos envolvendo o algoritmo DEMO e outros MOEAs foram realizados em [118], onde o DEMO apresentou resultados competitivos em relação ao NSGA-II e ao

SPEA [117] e superior em relação aos demais MOEAs analisados.

4.2.4 **GDE3**

Kukkonen & Lampinen [119] propuseram o algoritmo GDE3 (*The Third Step of Generalized Differential Evolution*), que é uma extensão do GDE [120] e do GDE2 [121]. O GDE3 introduz o uso da relação de dominância na tomada de decisão da substituição do vetor alvo pelo experimental gerado por ele. O algoritmo adota uma ordenação de Pareto no espaço das restrições para o tratamento das mesmas.

Uma comparação entre o GDE3 e o NSGA-II pode ser observada em [119], onde os resultados apresentados mostraram que o GDE3 é competitivo. Um outro estudo que apresenta uma comparação entre o GDE3, o NSGA-II, o MODE e o DEMO é apresentado por Vargas *et al.* [122].

4.2.5 **SPEA2**

O algoritmo *Strength Pareto Evolutionary Algorithm 2* - SPEA2, proposto por Zitzler *et al.* [123], é baseado em um AG e é um melhoramento do algoritmo SPEA [117]. A principal característica do SPEA2 é o uso de um arquivo externo para armazenar as soluções não-dominadas encontradas durante o processo evolutivo.

Diversos trabalhos que utilizam o SPEA2 com o algoritmo de busca são encontrados na literatura [124, 125, 126].

4.2.6 **AMOPSO**

O algoritmo AMOPSO (*Accelerated MO Particle Swarm Optimization*) foi proposto por Ellaia *et al.* [127] e é baseado no PSO para a solução de problemas de otimização multiobjetivo e problemas de otimização estrutural dimensional e topológica. O AMOPSO modifica o PSO padrão utilizando a ideia de dominância de Pareto na seleção da melhor partícula global e na seleção da melhor posição da partícula atual. Além disso, um método de *Opposition-Based Reinforcement Learning* [128] é usado para gerar a população inicial e as restrições são tratadas por uma estratégia de penalização exterior com um fator de penalização fixo.

Experimentos numéricos são realizados em [127] e os resultados obtidos são comparados com os encontrados pelo NSGA-II e pelo (MISA) [129]. O AMOSPO apresentou resultados satisfatórios para os problemas analisados, mostrando sua competitividade em relação aos dois algoritmos.

4.2.7 **MOPSO-CD**

Raquel & Naval Jr. [5] desenvolveram um algoritmo multiobjetivo denominado MOPSO-CD que estende o PSO padrão e incorpora ao algoritmo um arquivo externo que armazena as soluções não-dominadas. Além disso, um operador de mutação é incluído a fim de manter a diversidade de soluções não-dominadas no arquivo externo.

O MOPSO-CD usa o mecanismo de *crowding distance* na seleção do melhor global e no método de exclusão de partículas do arquivo externo. O melhor global das partículas é selecionado entre as soluções não-dominadas com os maiores valores de *crowding distance*. Sempre que o arquivo externo estiver cheio, o valor do *crowding distance* é novamente utilizado para selecionar qual solução será substituída no arquivo. Isso promove a diversidade entre as soluções que estão armazenadas no arquivo, uma vez que as soluções que estão nas áreas mais populosas provavelmente serão substituídas por uma nova solução [5].

O desempenho do MOPSO-CD é avaliado em três funções teste extraídas da literatura e faz uso de duas métricas de desempenho na análise. Os resultados mostraram que o MOPSO-CD é altamente competitivo ao convergir para a frente de Pareto e gerou um conjunto bem distribuído de soluções não-dominadas.

4.3 MEDIDAS DE DESEMPENHO

Para avaliar a qualidade de um MOEA, diversas métricas podem ser encontradas na literatura que consideram a diversidade e a convergência do conjunto de soluções [6]. Nesta tese, foram adotadas duas medidas de desempenho: o hipervolume e a função de resultado empírico. Elas foram escolhidas devido às suas popularidades e suas capacidades de indicar se um algoritmo obteve melhores resultados do que outro. A descrição de cada uma será apresentada a seguir.

4.3.1 **Hipervolume**

O hipervolume foi proposto por Zitzler & Thiele [117] e é uma medida de desempenho bastante popular na literatura devido ao fato de não exigir o conhecimento da Fronteira de Pareto real do problema. É uma medida unária que avalia tanto a convergência quanto a diversidade das soluções. A técnica fornece um valor calculado como sendo o hipervolume dos hipercubos definidos pelas soluções Pareto ótimas e um ponto de referência. A Fig. 19 apresenta um exemplo para um conjunto de soluções não-dominadas de um problema com dois objetivos. O hipervolume é a área da região hachurada e o conjunto de soluções Pareto-ótimas são os pontos em vermelho. Quanto maior o valor do hipervolume, melhor é a relação convergência/diversidade das soluções em relação à Frente de Pareto real do

problema. O código utilizado para a obtenção do hipervolume foi desenvolvido por Fonseca *et al.* [130][1].

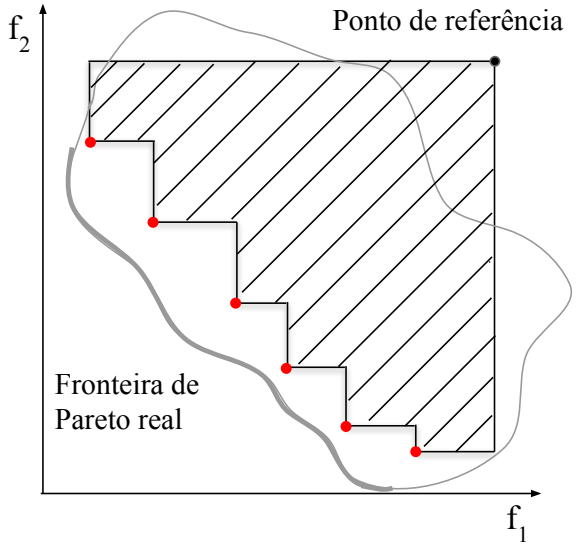


Figura 19 – Exemplo de um hipervolume com duas funções objetivo, adaptada de [6].

4.3.2 Função de resultado empírico

A função de resultado empírico (*Empirical Attainment Function* - EAF), proposta por Fonseca & Peter [131], atua como uma medida de distribuição de probabilidade dos resultados obtidos por um algoritmo no espaço dos objetivos. A EAF é um resumo dos resultados de várias execuções de um algoritmo e, ao mesmo tempo, permite detectar se e onde um algoritmo é melhor que outro. Uma discussão sobre a obtenção da curva EAF pode ser encontrada em [131] e [132].

Uma ferramenta baseada no software R foi desenvolvida por Manuel López-Ibáñez para calcular e gerar o gráfico EAF[2]. Um exemplo pode ser observado através da Fig. 20, na qual exibe um gráfico da EAF (*best*, *median* e *worst*) obtida por 10 execuções independentes de um determinado MOEA sobre um problema com 2 funções objetivo. Analisando a figura, é possível observar que qualquer solução obtida por qualquer uma dessas 10 execuções do algoritmo tem 0% de chance de dominar algum dos pontos da curva *best*, 50% de chance de ser dominada por algum dos pontos da curva *median* e 0% de chance de ser dominada por algum dos pontos da curva *worst*.

[1]Disponível em: http://iridia.ulb.ac.be/ manuel/hypervolume.
[2]Disponível em: http://lopez-ibanez.eu/eaftools.

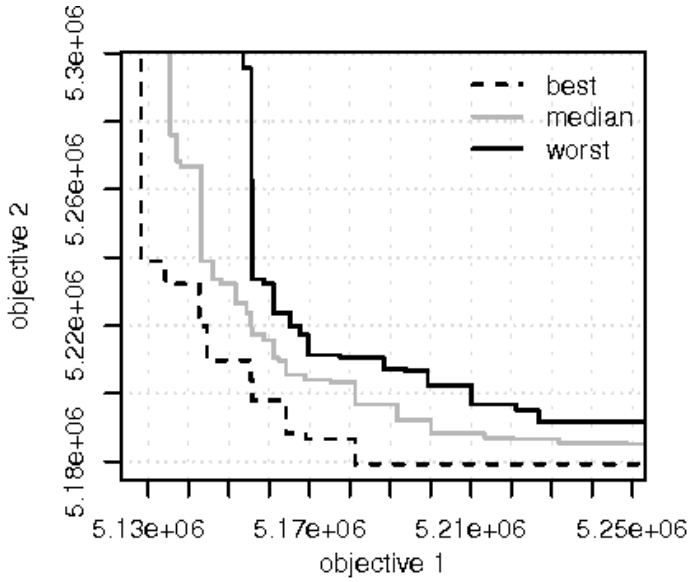


Figura 20 – Exemplo de um gráfico EAF (*best*, *median* e *worst*), extraída de [7].

4.4 PERFIS DE DESEMPENHO

Uma ferramenta gráfica conhecida como perfis de desempenho (do inglês, *Perfomance Profiles*) foi proposta por Dolan e Moré [133] para facilitar a interpretação e a visualização dos resultados obtidos em experimentos com grande quantidade de dados.

A avaliação experimental de algoritmos na prática não é nada trivial e apresenta algumas dificuldades, tais como Silva [134]: a definição de conjuntos de problemas é heterogênea, a decisão de como representar e interpretar os resultados obtidos nos experimentos e a determinação de medidas de desempenho para avaliar o algoritmo.

Considere um conjunto P de problemas teste p_j, com $j = 1, 2, ..., n_p$, um conjunto de algoritmos a_i com $i = 1, 2, ..., n_a$ e $t_{p,a} > 0$ uma métrica de desempenho (como, por exemplo, tempo computacional, média, etc.). A razão de desempenho é definida como:

$$r_{p,a} = \frac{t_{p,a}}{min\{t_{p,a} : a \in A\}}. \tag{4.2}$$

O perfil de desempenho do algoritmo é definido como:

$$\rho_a(\tau) = \frac{1}{n_p}|\{p \in P : r_{p,a} \leq \tau\}| \tag{4.3}$$

onde $\rho_a(\tau)$ é a fração de problemas resolvidos pelo algoritmo com desempenho dentro de um fator τ do melhor desempenho obtido, considerando todos os algoritmos.

Algumas propriedades em relação ao melhor desempenho do algoritmo podem ser observadas em Barbosa *et al.* [135]:

- Quando $\tau = 1$, $\rho_a(\tau)$ é a fração de problemas em que o algoritmo apresenta melhor desempenho quando comparado com os demais algoritmos;

- Quando $\tau = \infty$, $\rho_a(\tau)$ representa a fração de problemas que o algoritmo consegue resolver;
- $\rho_a(1)$ é a porcentagem de problemas em que o algoritmo a tem melhor desempenho. Considerando dois algoritmos E e F, se $\rho_E(1) > \rho_F(1)$ então, o algoritmo E resolve uma quantidade maior de problemas que o algoritmo F.

Uma extensão dessa ferramenta para trabalhar com algoritmos estocásticos foi proposta por Barreto *et al.* [136] e ficou conhecida como perfil de desempenho probabilístico. A ideia era de utilizar uma ferramenta que a princípio foi desenvolvida para ambientes determinísticos em algoritmos estocásticos, já que estes trazem muita incerteza devido aos diferentes desempenhos nas suas diversas execuções. Outras aplicações envolvendo os perfis de desempenho em problemas com restrições, podem ser observadas nas referências [135, 137].

A seguir é apresentado um exemplo do uso dos perfis de desempenho onde são analisados três algoritmos A, B e C para a solução de 5 problemas P1, P2, P3, P4 e P5. Os valores obtidos para cada problema em cada algoritmo é detalhado na Tabela 8.

Tabela 8 – Desempenho dos cinco problemas obtidos através dos algoritmos A, B e C.

	P1	P2	P3	P4	P5
A	10.5	2.9	15.9	0.7	7.7
B	9.3	4.5	16.0	1.1	7.5
C	12.9	2.2	13.6	2.0	6.5

A Figura 21 apresenta o gráfico dos perfis de desempenho onde os algoritmos A e C apresentaram o maior valor de $\rho(1)$, o que significa que os métodos obtiveram o melhor desempenho em um número maior de problemas. O algoritmo C apresentou o menor valor de τ, tal que $\rho(\tau) = 1$, e, portanto, é considerado o mais robusto. A Tabela 9 apresenta as áreas sob as curvas dos perfis de desempenho, onde o algoritmo C obtive o maior valor de área e é considerado o método com o melhor desempenho global.

Tabela 9 – Área normalizada sob as curvas dos perfis de desempenho dos algoritmos A, B e C.

Algoritmo	A	B	C
Área	0.83449	0.82442	1.0

Os perfis de desempenho serão utilizados nas análises dos experimentos que serão conduzidos no Capítulo 6. A ferramenta irá auxiliar as comparações entre os algoritmos multiobjetivo no segundo conjunto de experimentos.

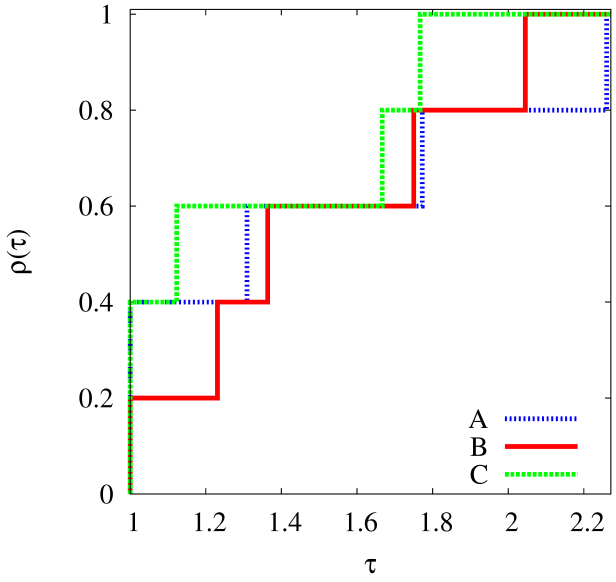


Figura 21 – Um exemplo do uso da ferramenta perfis de desempenho.

4.5 TRABALHOS RELACIONADOS

Uma variedade de trabalhos envolvendo MOEAs de diversas naturezas, com dois ou mais objetivos, têm sido desenvolvidos ao longo dos anos. Uma revisão da literatura composta por alguns desses MOEAs e, em particular, aqueles envolvendo enxame de partículas, é apresentada a seguir.

4.5.1 **MOEAs de diversas naturezas**

Yang [138] estendeu o algoritmo de vaga-lume (*Firefly Algorithm* - FA) para a solução de problemas de otimização multiobjetivo. A abordagem proposta, denominada MOFA, foi validada usando um subconjunto selecionado de funções teste e, em seguida, o algoritmo foi aplicado em problemas de otimização em engenharia. Os resultados dos experimentos foram comparados com resultados de outros algoritmos encontrados na literatura, tais como o NSGA-II, DEMO e Bees [139]. O algoritmo de vaga-lume proposto se mostrou bastante eficiente e competitivo.

Um novo algoritmo *Cuckoo Search* foi desenvolvido por Yang & Deb [140] para a solução de problemas de otimização multiobjetivo. Cinco funções teste foram utilizadas para avaliar o desempenho do algoritmo, são elas: ZDT1, ZDT2, ZDT3, SCH E LZ. Em comparação com outros algoritmos, como o NSGA-II, MODE, DEMO, o *Cuckoo Search* apresentou um bom desempenho na maioria dos problemas analisados.

Coello Coello *et al.* [141] apresentaram um estudo que analisou os últimos desenvolvimentos no campo das metaheurísticas multiobjetivas para resolver problemas com foco na otimização da topologia, de forma e de dimensionamento de estruturas em engenharia

civil. Os autores reveram os algoritmos, as aplicações e as características mais relevantes dos "solucionadores". O trabalho aborda ainda uma série de questões relevantes e em aberto relacionado a problemas de otimização estrutural e técnicas multiobjetivas.

Yepes *et al.* [142] propuseram uma abordagem cognitiva para analisar e reduzir o conjunto Pareto-ótimo na otimização multiobjetivo de problemas de otimização estrutural. A abordagem identifica os padrões de comportamento e pontos críticos do processo de resolução que refletem o conhecimento relevante derivado da perspectiva cognitiva. Os resultados indicaram que as soluções obtidas aumentaram a vida útil da estrutura, produzindo estruturas mais ecológicas e resistentes.

Dois algoritmos de otimização de colônia de formigas (*Ant Colony Optimization* - ACO) foram propostos por Angelo *et al.* [14] para resolver problemas de otimização estrutural multiobjetivo. Cinco problemas de engenharia estrutural multiobjetivo com variáveis de projeto discretas foram utilizados, tendo como objetivos a minimização do peso da estrutura e a minimização do seu deslocamento máximo nodal. O conjunto Pareto gerado nos experimentos são avaliados por meio de métricas de desempenho e os modelos obtidos são comparados com soluções disponíveis na literatura a partir de estudos mono-objetivos.

Jiang & Yang [143] desenvolveram um algoritmo multiobjetivo baseado na decomposição denominado MOEA/D para a solução de problemas de otimização multiobjetivo. No algoritmo proposto, uma estratégia é empregada para dividir todo o processo de otimização em duas fases. Com base na aglomeração de soluções encontradas na primeira fase, o algoritmo decide se há ou não recursos computacionais para lidar com subproblemas não resolvidos na segunda fase. O desempenho do algoritmo proposto é investigado em alguns problemas *benchmarks* existentes a literatura. Os resultados experimentais mostraram que o MOEA/D proposto produziu desempenho promissor nos problemas analisados.

As empresas de energia são incapazes de suportar a demanda do consumidor devido ao crescimento da população, indústrias e edifícios. O uso de aparelhos elétricos automatizados aumentou exponencialmente nas atividades do dia a dia. Para manter um equilíbrio entre a oferta e a demanda, as empresas estão introduzindo a abordagem do gerenciamento da demanda. Muralitharan *et al.* [144] desenvolveram um estudo no qual utiliza um algoritmo de otimização multiobjetivo para a redução de custos para uso de energia e redução do tempo de espera para a execução de dispositivos elétricos. O resultados revelaram que se o consumidor exceder o limite, os dispositivos em funcionamento param temporariamente para manter o uso de energia abaixo do nível de limite, e retomam os aparelhos parados posteriormente. Além disso, a técnica proposta minimiza a conta geral de eletricidade e o tempo de espera para a execução dos aparelhos elétricos.

Yang *et al.* [145] apresentaram um estudo que investigou a viabilidade da aplicação de um modelo multiobjetivo em um projeto de edificações sustentáveis que envolvia a

integração de um modelo de desempenho energético de um edifício. O NSGA-II foi utilizado e o problema apresenta três objetivos conflitantes: minimizar o custo de construção do edifício sustentável, minimizar seu desempenho de energia e maximizar a taxa de abertura das janelas. Os resultados encontrados mostraram que um alto custo de construção foi obtido satisfazendo simultaneamente um baixo desempenho da energia e uma alta taxa de abertura das janelas.

4.5.2 **MOEAs baseados em enxame de partículas**

O algoritmo de Otimização por Enxame de Partículas Multiobjetivo (*Multiobjective Particle Swarm Optimization* - MOPSO) é amplamente estudado nos mais diversos tipo de aplicações [106, 146, 147, 148, 149]. A seguir, é apresentado uma breve revisão da literatura com trabalhos relacionados.

Hosseini *et al.* [150] abordaram técnicas de otimização estrutural multiobjetivo para encontrar os melhores valores de áreas das seção transversal e a configuração dos nós de treliças planas. Um algoritmo MOPSO foi utilizado na otimização e para verificar seu desempenho, três exemplos encontrados na literatura são abordados. A comparação dos resultados mostrou uma variedade desejável de soluções na frente de Pareto.

Um novo método denominado *Parallel Cell Coordinate System* (PCCS) é proposto por Hu & Yen [151] para a solução de problemas de otimização multiobjetivo. A estratégia proposta é baseada no PCCS e em um MOPSO auto-adaptativo, denominado pccsAMOPSO. Doze problemas teste bi e tri-objetivos são utilizados nos experimentos. A validação do algoritmo proposto é feita utilizando oito MOEAs encontrados na literatura e os resultados mostraram que o pccsAMOPSO proposto supera os demais algoritmos analisados.

Sethanan & Neungmatcha [152] apresentaram uma algoritmo PSO multiobjetivo com as variantes *gbest*, *lbest* e *nbest* combinadas (MO-GLNPSO), para resolver problemas de planejamento de rotas na colheita mecânica da cana-de-açúcar (*Mechanical Harvester Route Planning* - MHRP). As soluções encontradas pelos autores levaram a potenciais economias de custos em relação a planejamentos feito manualmente pelo responsável da usina.

Um novo algoritmo denominado *fuzzy Multi-Objective Particle Swarm Optimization* (f-MOPSO) foi proposto por Rezaei *et al.* [153]. O f-MOPSO emprega um sistema de inferência difusa para considerar todos os desempenhos parciais para cada ponto ao otimizar os valores da função objetivo. O algoritmo foi comparado com outros dois MOPSO que foram usados em um estudo de caso de uso conjuntivo de águas superficiais e subterrâneas na planície de Najafabad, no Irã. De forma geral, o f-MOPSO superou os outros dois algoritmos no que diz respeito ao desempenho e análise da fronteira de Pareto.

Um algoritmo adaptativo denominado AMOPSO (*Adaptive Multiobjective Particle Swarm Optimization*), baseado em uma estrutura híbrida de informações de entropia de distribuição de solução e espaçamento de população, é desenvolvido por Han *et al.* [154] para melhorar seu desempenho em termos de velocidade e precisão. Algumas alterações foram feitas a fim de obter uma melhor distribuição de partículas para apresentar diversidade e convergência adequadas, que podem equilibrar a exploração global e as capacidades de exploração local dessas partículas. O desempenho do AMOPSO é validado e comparado com outros cinco algoritmos de última geração em vários problemas *benchmarks* e sistema de distribuição de água. Os resultados experimentais validam a eficácia do algoritmo proposto, bem como demonstram que o AMOPSO supera os outros algoritmos MOPSO na resolução de MOPs.

A seleção de recursos é uma importante técnica de pré-processamento de dados em problemas de classificação. Geralmente, existem algumas situações em que um usuário está interessado não apenas em maximizar o desempenho da classificação, mas também em minimizar o custo, que pode estar diretamente associado aos recursos. Esse tipo de problema é chamado de seleção de recursos baseada em custo. No entanto, a maioria das abordagens de seleção de recursos tratam essa tarefa como um problema de otimização de um único objetivo. Zhang *et al.* [155] apresentaram um primeiro estudo de otimização de enxame de partículas multiobjetivo para este tipo de problema. A fim de melhorar a capacidade de pesquisa do algoritmo, uma tecnologia de codificação baseada em probabilidade e um operador híbrido eficaz, juntamente com as ideias de *crowding distance*, arquivo externo e a relação de dominação de Pareto, são aplicados ao novo algoritmo. O algoritmo proposto é comparado com vários algoritmos de seleção de recursos multiobjetivos em cinco conjuntos de dados *benchmarks*. Resultados experimentais mostraram que o algoritmo proposto é um método altamente competitivo para resolver problemas de seleção de recursos baseados em custo.

Um novo algoritmo denominado DEMPSO (*Diversity Enhanced Multiobjective Particle Swarm Pptimization*) é proposto por Pan *et al.* [156] visando equilibrar a convergência e a diversidade durante o procedimento de busca em um algoritmo PSO multiobjetivo. O novo algoritmo concentra-se nas seguintes inovações. Primeiro, é introduzida uma formulação simplificada orientada para o líder na atualização do PSO. Segundo, é proposto um novo método de análise de variável de decisão para aprimoramento da diversidade. E, em terceiro lugar, uma estratégia de seleção de líderes de duas vertentes é apresentada. Os resultados experimentais em testes *benchmarks* ilustram que o DEMPSO supera outros algoritmos PSO e melhorou muito a capacidade de manutenção da diversidade em espaços de alta dimensão.

5 ALGORITMO PROPOSTO

O algoritmo PSO, inicialmente introduzido na Seção 3.1, tem sofrido muitas mudanças desde a sua criação em 1995. Novas estratégias com o objetivo de melhorar seu desempenho e sua competitividade têm sido propostas na literatura [157, 158]. Uma técnica aperfeiçoada da otimização por enxame de partículas chamada de *Craziness based Particle Swarm Optimization* (CRPSO), proposta por Kar *et al.* [25], é usada nesta tese com o intuito de eliminar as limitações apresentadas pelo PSO padrão. Uma nova expressão da velocidade v_i e um operador chamado "*craziness velocity*" foi introduzida ao algoritmo padrão. A nova velocidade pode ser expressa como [25]

$$\begin{aligned} v_j^{(i)} = \; & r_2 \cdot sign(r_3) \cdot v_j^{(i)}(t) + (1 - r_2) c_1 \cdot r_1 (x_{pbest}^{(i)} - x_j^{(i)}) + (1 - r_2) \cdot \\ & \cdot \; c_2 \; \cdot \; (1 - r_1)(x_{gbest}^{(i)} - x_j^{(i)}) + P(r_4) \cdot sign2(r_4) \cdot v_j^{craziness}, \end{aligned} \tag{5.1}$$

onde r_1, r_2, r_3 e r_4 são valores aleatórios uniformemente distribuidos no intervalo [0,1), $sign(r_3)$ é uma função definida como

$$sign(r_3) = \begin{cases} -1, & r_3 \leq 0.5 \\ 1, & r_3 > 0.5, \end{cases} \tag{5.2}$$

$v_j^{craziness}$, o operador *craziness*, é um parâmetro definido pelo usuário no intervalo [v^{min}, v^{max}], $P(r_4)$ e $sign2(r_4)$ são definidos, respectivamente, como

$$P(r_4) = \begin{cases} 1, & r_4 \leq Pcr \\ 0, & r_4 > Pcr \end{cases} \tag{5.3}$$

e

$$sign2(r_4) = \begin{cases} -1, & r_4 \geq 0.5 \\ 1, & r_4 < 0.5 \end{cases} \tag{5.4}$$

e Pcr é uma probabilidade pré-definida de *craziness*. Pode-se notar que, enquanto Pcr é um valor fixo, $P(r_4)$ varia cada vez que a velocidade é calculada.

Um estudo comparativo entre o algoritmo PSO padrão e o CRPSO foi desenvolvido pela autora desta tese em Carvalho *et al.* [159]. O efeito da introdução do operador *craziness* e dos demais parâmetros na equação da velocidade do CRPSO foram analisados no artigo. Os experimentos foram conduzidos utilizando quatro estruturas treliçadas e comprovaram a superioridade do CRPSO. Além disso, os resultados apresentados pelos dois algoritmos foram comparados com aqueles encontrados por outros algoritmos da literatura.

5.1 VERSÃO MULTIOBJETIVO

A versão multiobjetivo do algoritmo CRPSO foi baseada no algoritmo MOPSO-CD[1], apresentado na Subseção 4.2.7. O MOPSO-CD usa o conceito de arquivo externo *ARQ* para armazenar as soluções não-dominadas. Dessa maneira, o algoritmo CRPSO foi modificado e sua versão multiobjetivo possui a equação da velocidade expressa como

$$\begin{aligned} v_j^{(i)}(t+1) = \; & r_2 \cdot sign(r_3) \cdot v_j^{(i)}(t) + (1-r_2)c_1 \cdot r_1(x_{pbest}^{(i)} - x_j^{(i)}) + (1-r_2) \cdot c_2 \cdot \\ & (1-r_1)(ARQ[gbest] - x_j^{(i)}) + P(r_4) \cdot sign2(r_4) \cdot v_j^{craziness}, \end{aligned} \tag{5.5}$$

onde $ARQ[gBest]$ é a melhor solução não-dominada do arquivo externo.

Um operador de mutação é atribuído ao algoritmo com uma taxa de definida como $pMut = 0.5$. Vale ressaltar, que o mecanismo de *crowding distance* junto com o operador de mutação mantém a diversidade de soluções não-dominadas no arquivo externo.

Desse modo, a versão multiobjetivo do CRPSO abordado nesta tese é definida como *Multiobjective Craziness based Particle Swarm Optmization* - MOCRPSO e os passos do algoritmo são apresentados no Algoritmo 1. Inicialmente, são atribuídos valores aleatórios para os vetores de posição x_i e velocidade v_i. Cada partícula i é avaliada, e, em seguida, seu $pBest_i$ é calculado. A melhor partícula do enxame é então atribuída ao $gBest$. Posteriormente, as soluções não-dominadas encontradas no enxame inicial são armazenadas no arquivo externo ARQ.

Os valores de *crowding distance* de cada solução não-dominada são computados no arquivo externo. Em seguida, essas soluções são ordenadas de forma decrescente de acordo com seus valores de *crowding distance*. Para cada partícula do enxame, são atualizados seus valores da velocidade v_i e da posição x_i. Uma mutação é aplicada, dada uma taxa de mutação. Os novos valores para as partículas são avaliados e o $pBest_i$, é calculado. Em seguida, o $gBest$ é atualizado. As novas soluções não-dominadas são inseridas no arquivo externo, caso não sejam dominadas por nenhuma das soluções armazenadas. Além disso, todas as soluções dominadas pelas novas soluções são removidas do mesmo. Caso o arquivo esteja cheio no momento da inserção das novas soluções, é calculado os valores de *crowding distance* de todas as soluções desse arquivo, ordenados de forma decrescente e, por fim, uma partícula é selecionada aleatoriamente de uma parte inferior do arquivo, constituída pela região mais populosa. Logo, a partícula selecionada é substituída pela nova solução. Todo esse procedimento descrito é repetido até que o número máximo de iterações seja alcançado.

Ao final da execução, o arquivo externo ARQ possui um conjunto com a imagem das soluções não-dominadas, que representa a Fronteira de Pareto de um dado problema.

A definição da melhor e da pior partícula é feita através da escolha aleatória de uma partícula em uma porção equivalente a 10% do total de partículas do enxame previamente

[1] https://sites.google.com/site/prosnaval/codes

ordenados em ordem decrescente de acordo com seu valor de *crowding distance*. Para a melhor partícula essa porção corresponde às partículas que possuem os maiores valores de *crowding distance* e para a pior partícula, os menores valores de *crowding distance*.

Algoritmo 1: Pseudocódigo do algoritmo MOCRPSO.

```
1  enquanto i ← 0 até tamanhoEnxame faça
2      inicializa x_i e v_i aleatoriamente (x é o enxame de partículas);
3      avalia x_i;
4      inicializa pBest_i;
5  fim
6  inicializa gBest;
7  armazena os vetores não-dominados encontrados em x no arquivo ARQ (ARQ é o
   arquivo externo que armazena soluções não-dominadas);
8  repita
9      calcula os valores de crowding distance de cada solução não-dominada no arquivo
       ARQ;
10     classifica as soluções não-dominadas em ARQ em ordem decrescente de acordo
       com os valores de crowding distance;
11     enquanto i ← 0 até tamanhoEnxame faça
12         atualiza v_i de acordo com a Eq. 5.5;
13         atualiza x_i de acordo com a Eq. 3.2;
14         se i < MAX_GEN * pMut então
15             aplica mutação em x_i;
16         fim
17         avalia x_i;
18         pBest_i ← melhor entre x_i e pBest_i ;
19     fim
20     gBest ← é atualizado;
21     insere toda nova solução não-dominada em x no arquivo ARQ se não for
       dominada por nenhuma das soluções armazenadas (todas as soluções dominadas
       pela nova solução são removidas do arquivo);
22     se ARQ está cheio então
23         calcula os valores de crowding distance de cada solução não-dominada no
           arquivo ARQ;
24         classifica as soluções não-dominadas em ARQ em ordem decrescente de
           acordo com os valores de crowding distance;
25         seleciona aleatoriamente a partícula de uma parte inferior especificada
           constituída pela região mais populosa do arquivo e, em seguida, a substitui
           pela nova solução;
26     fim
27 até número máximo de iterações ser alcançado;
```

5.1.1 **Codificação**

Na Subseção 3.2.2 foi apresentado o conceito de restrições de cardinalidade, um tipo de codificação especial que gera um agrupamento das barras com o intuito de modificar o processo de otimização, melhorando a qualidade das soluções.

Um exemplo de codificação de uma partícula do MOCRPSO inspirado no cromossomo proposto por Barbosa & Lemonge em [160] é apresentado na Fig. 22. A figura mostra uma partícula com 12 variáveis de projeto, onde as restrições de cardinalidade (m=2) são representadas pelas duas primeiras variáveis na parte esquerda da partícula. Na parte direita existem 10 índices que apontam para as variáveis de projeto da parte esquerda. Estes índices apontam para uma tabela com um determinado número de valores discretos (ou áreas da seção transversal das barras) a serem escolhidos.

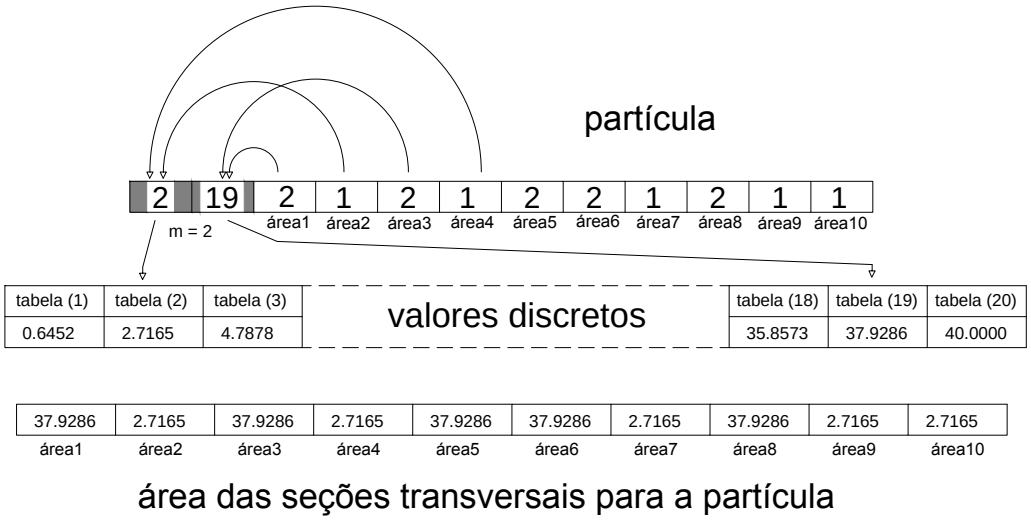


Figura 22 – Exemplo de codificação de uma partícula com 12 áreas da seção transversal para m=2.

Neste exemplo, a variável de projeto 1 aponta para o índice 2 da tabela retornando o valor 2.7165 cm^2 e a variável de projeto 2, aponta para o índice 19, retornando 37.9286 cm^2. Na parte direita da codificação, a quarta variável de projeto (área 4), por exemplo, aponta para a primeira variável de projeto e esta barra tem a área da seção transversal igual a 2.7165 cm^2. Essa codificação será utilizada nos experimentos computacionais analisados no Capítulo 6.

5.2 METODOLOGIA

Em síntese das informações apresentadas, a metodologia proposta nesta tese trata-se de um PSO multiobjetivo baseado no algoritmo MOPSO-CD [5] com adaptações para melhoria de sua convergência baseado no algoritmo CRPSO [25]. Além disso, as barras

das estruturas serão agrupadas de acordo com um número m máximo de diferentes áreas da seção transversal, conceito este definido como restrições de cardinalidade. Por fim, as restrições existentes nos experimentos serão tratadas utilizando o método APM [27]. Essa combinação de elementos será tratada aqui com o nome de MOCRPSO (*Multiobjective Craziness based Particle Swarm Optmization*). O Capítulo 6, a seguir, apresenta o conjunto de experimentos que serão conduzidos para analisar o desempenho do algoritmo proposto.

6 EXPERIMENTOS COMPUTACIONAIS

Neste capítulo, o desempenho do MOCRPSO é investigado em problemas de otimização multiobjetivo com e sem restrições. Três conjuntos de experimentos são utilizados nesta análise. Inicialmente, seis funções teste com e sem restrições, tradicionalmente encontradas na literatura [28], são utilizadas para a validação do algoritmo proposto e os resultados são comparados com aqueles encontrados pelo algoritmo NSGA-II. A escolha do NSGA-II para a comparação se deve ao fato de que o algoritmo é largamente encontrado na literatura em trabalhos envolvendo a solução de MOPs nas mais variadas aplicações. O segundo e terceiro conjunto de experimentos trata-se de problemas de otimização de estruturas treliçadas com restrições, descritas previamente na Seção 2.3.

Em todos os experimentos, o número máximo de avaliações da função objetivo é 50000 (50 partículas e 1000 gerações), o número de execuções independentes é 100 e as soluções apresentadas são rigorosamente factíveis. Cada experimento usa uma população inicial gerada aleatoriamente. Os parâmetros $c_1 = c_2 = 2.05$, $v^{craziness} = 0.001$, $Pcr = 0.5$, topologia de vizinhança global e tolerância para restrições de igualdade $\epsilon = 0.0001$. O limite do arquivo externo $ARQ = 500$. O código foi desenvolvido utilizando linguagem C e as estruturas são analisadas pelo Método dos Elementos Finitos (*Finite Element Method* - FEM) [161] durante o processo evolutivo.

6.1 PRIMEIRO CONJUNTO DE EXPERIMENTOS

O primeiro conjunto de experimentos é composto por 6 funções teste, sendo 3 delas parte da família de funções ZDT (*Zitzler-Deb-Thiele*) [162]. A família de funções ZDT é composta por seis problemas de minimização com 2 objetivos e nenhuma restrição. Neste estudo, as funções ZDT1, ZDT2 e ZDT3 foram utilizadas e são descritas a seguir

ZDT1

min

$$\begin{aligned} f_1(x) &= x_1 \text{ e} \\ f_2(x) &= a(x)b(f_1(x), a(x)) \\ a(x) &= 1 + \frac{9}{29}\sum_{i=2}^{30} x_1 \\ b(f_1(x), a(x)) &= 1 - \sqrt{\frac{f_1(x)}{a(x)}} \end{aligned} \tag{6.1}$$

ZDT2

min

$$\begin{aligned} f_1(x) &= x_1 \text{ e} \\ f_2(x) &= a(x)b(f_1(x), a(x)) \\ a(x) &= 1 + \frac{9}{29}\sum_{i=2}^{30} x_1 \\ b(f_1(x), a(x)) &= 1 - \left(\frac{f_1(x)}{a(x)}\right)^2 \end{aligned} \tag{6.2}$$

ZDT3

min

$$\begin{aligned} f_1(x) &= x_1 \text{ e} \\ f_2(x) &= g(x)b(f_1(x), a(x)) \\ a(x) &= 1 + \frac{9}{29}\sum_{i=2}^{30} x_1 \\ b(f_1(x), a(x)) &= 1 - \sqrt{\frac{f_1(x)}{a(x)}} - \left(\frac{f_1(x)}{a(x)}\right)^2 \sin(10\pi f_1(x)) \end{aligned} \tag{6.3}$$

A quarta função teste é conhecida como CTP1. Esta função tem dois objetivos e três restrições e é descrita como segue [19]

min

$$\begin{aligned} f_1(x) &= x \text{ e} \\ f_2(x) &= (1 + x_2)\exp\left(\frac{x_1}{1 + x_2}\right), \end{aligned}$$

sujeito a

$$\begin{aligned} g_1(x) &= \frac{f_2(x)}{0.858\exp(-0.541 f_1(x))} \geq 1 \text{ e} \\ g_2(x) &= \frac{f_2(x)}{0.728\exp(-0.295 f_1(x))} \geq 1. \end{aligned} \tag{6.4}$$

A quinta função contém dois objetivos e seis restrições. A função denominada *Osyczka Kundu* é definida como

min

$$\begin{aligned} f_1(x) &= -25(x_1 - 2)^2 - (x_2 - 2)^2 - (x_3 - 1)^2 - (x_4 - 4)^2 - (x_5 - 1)^2) \text{ e} \\ f_2(x) &= \sum_{i=1}^{6} x_i^2, \end{aligned}$$

sujeito a

$$\begin{aligned} g_1(x) &= x_1 + x_2 - 2 \geq 0, \\ g_2(x) &= 6 - x_1 - x_2 \geq 0, \\ g_3(x) &= 2 - x_2 + x_1 \geq 0, \\ g_4(x) &= 2 - x_1 + 3x_2 \geq 0, \\ g_5(x) &= 4 - (x_3 - 3)^2 - x_4 \geq 0 \text{ e} \\ g_6(x) &= (x_5 - 3)^2 + x_6 - 4 \geq 0. \end{aligned} \tag{6.5}$$

A sexta e última função teste, denominada *Viennet*, é composta por três objetivos e não possui restrição. A função é descrita à seguir

min

$$\begin{aligned} f_1(x) &= 0.5(x_1^2 + x_2^2) + \sin(x_1^2 + x_2^2), \\ f_2(x) &= \frac{(3x_1 - 2x_2 + 4)^2}{8} + \frac{(x_1 - x_2 + 1)^2}{27} + 15 \text{ e} \\ f_3(x) &= \frac{1}{x_1^2 + x_2^2 + 1} - 1.1\exp(-(x_1^2 + x_2^2)) \end{aligned} \tag{6.6}$$

6.1.1 **Análise dos resultados**

As curvas EAF (*best*, *median* e *worst*) dos algoritmos MOCRPSO e NSGA-II nos problemas analisados são apresentadas nas Figs. 23 a 27. Como a ferramenta utilizada para gerar as curvas não suporta funções tridimensionais, tais curvas não serão exibidas para a função *Viennet*. A comparação entre os algoritmos é apresentada também pela diferença entre as melhores curvas EAF, que representam a imagem das soluções não-dominadas obtidas em todas as execuções independentes, e podem ser observadas nas Figs. 28 a 33. O código tridimensional para obtenção das curvas EAF pode ser encontrado em [163][1]. Por fim, os valores de hipervolume normalizados são apresentados nessas figuras para todos os problemas teste.

Pode-se observar que as curvas EAF das soluções encontradas pelo MOCRPSO são semelhantes àquelas obtidas pelo NSGA-II. O valor de hipervolume obtido pelo MOCRPSO nas quatro primeiras funções teste (ZDT1, ZDT2, ZDT3 e CTP1) para as curvas EAF *best* são maiores que os obtidos pelo NSGA-II. Nas funções Osyezka Kundu e Viennet, o MOCRPSO obteve valores de hipervolume menores, porém competitivos.

Após os resultados e análises apresentados para o primeiro conjunto de experimentos, é possível concluir que o algoritmo MOCRPSO proposto aqui se mostrou competitivo e robusto nos problemas testes analisados. Sendo assim, o algoritmo será utilizado como mecanismo de otimização no segundo e terceiro conjunto de experimentos propostos nesta tese.

[1] https://eden.dei.uc.pt/∼cmfonsec/software.html

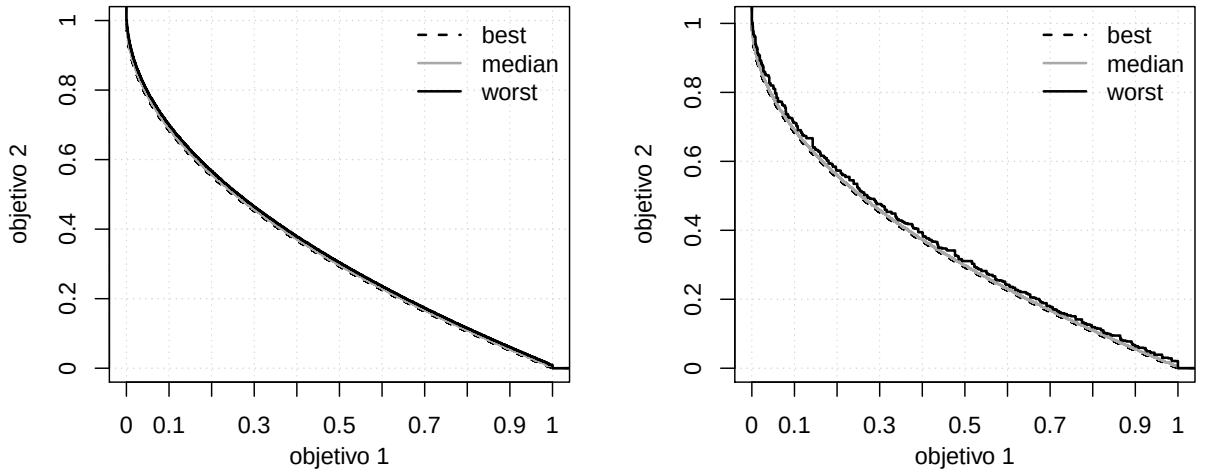


Figura 23 – Curvas EAF (*best*, *median* e *worst*) do MOCRPSO (à esquerda) e NSGA-II (à direita) para a função ZDT1.

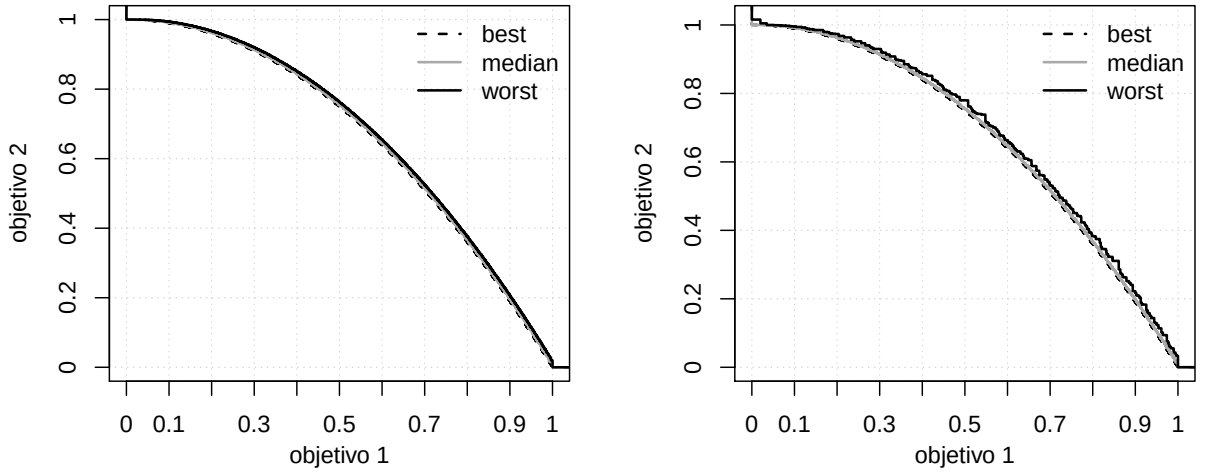


Figura 24 – Curvas EAF (*best*, *median* e *worst*) do MOCRPSO (à esquerda) e NSGA-II (à direita) para a função ZDT2.

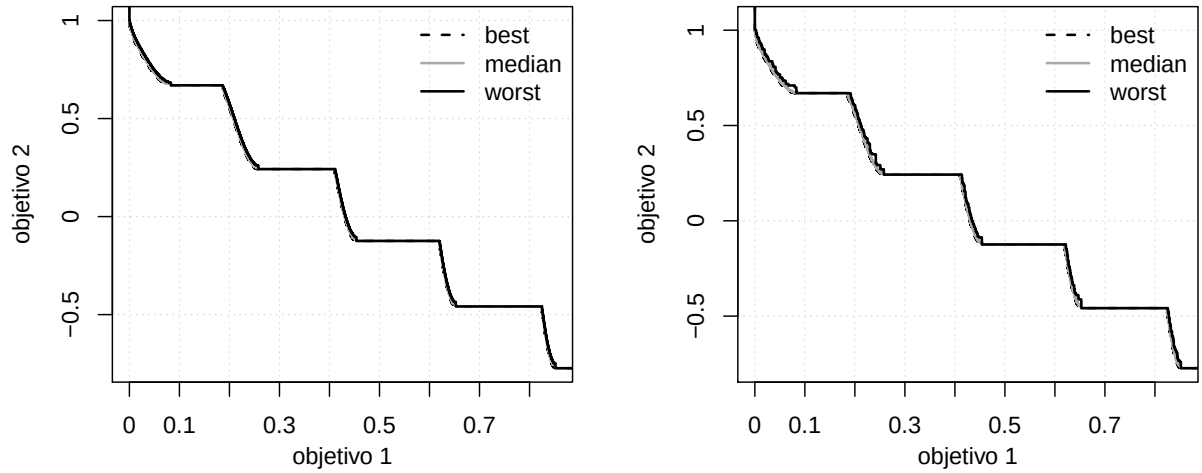


Figura 25 – Curvas EAF (*best*, *median* e *worst*) do MOCRPSO (à esquerda) e NSGA-II (à direita) para a função ZDT3.

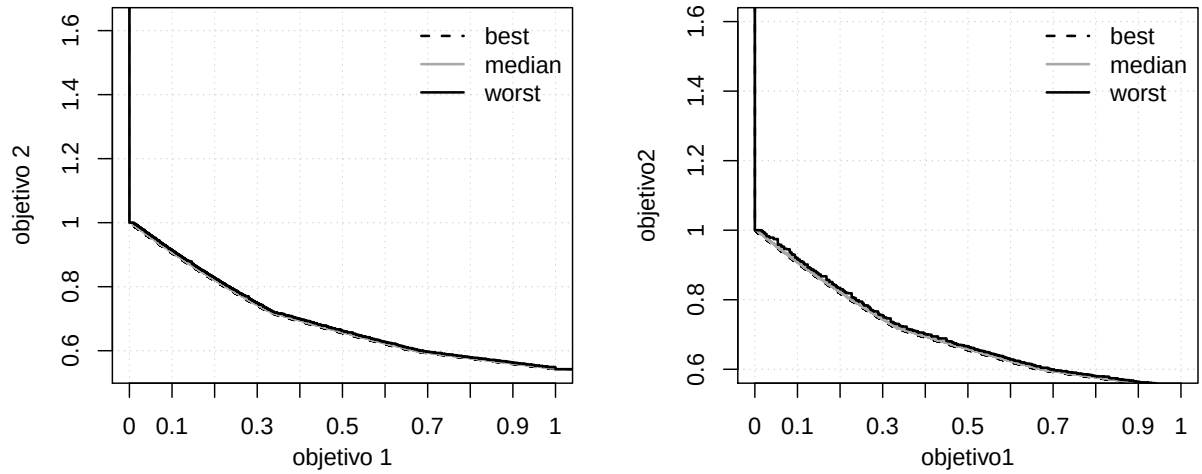


Figura 26 – Curvas EAF (*best*, *median* e *worst*) do MOCRPSO (à esquerda) e NSGA-II (à direita) para a função CTP1.

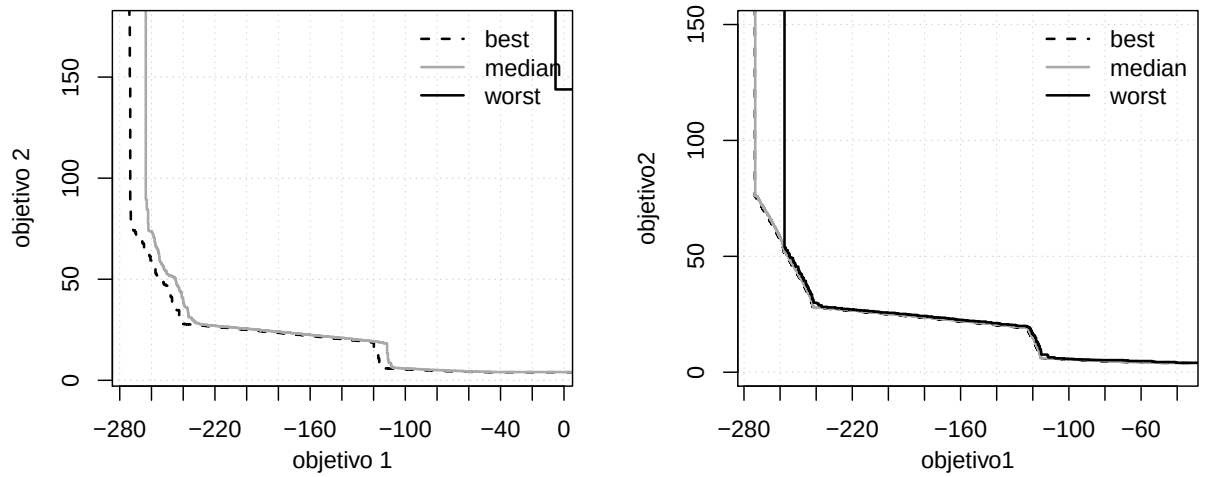


Figura 27 – Curvas EAF (*best*, *median* e *worst*) do MOCRPSO (à esquerda) e NSGA-II (à direita) para a função *Osyczka Kundu*.

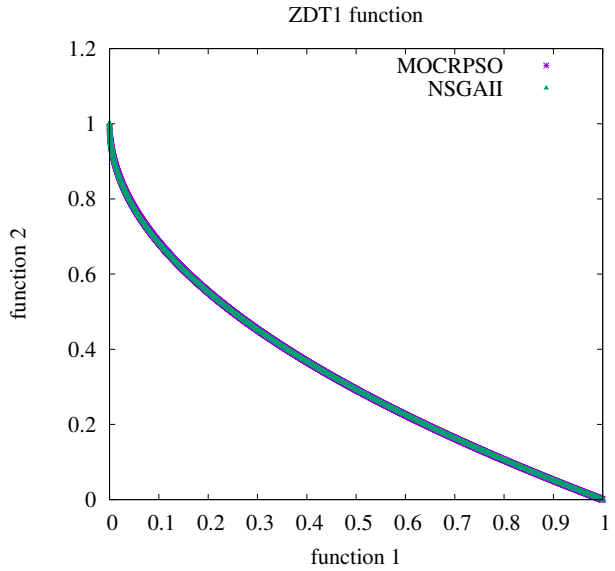


Figura 28 – Diferença entre a melhor curva EAF dos algoritmos MOCRPSO e NSGA-II para a função ZDT1. Hipervolume normalizado MOCRPSO: 0.666661 e NSGA-II: 0.666503.

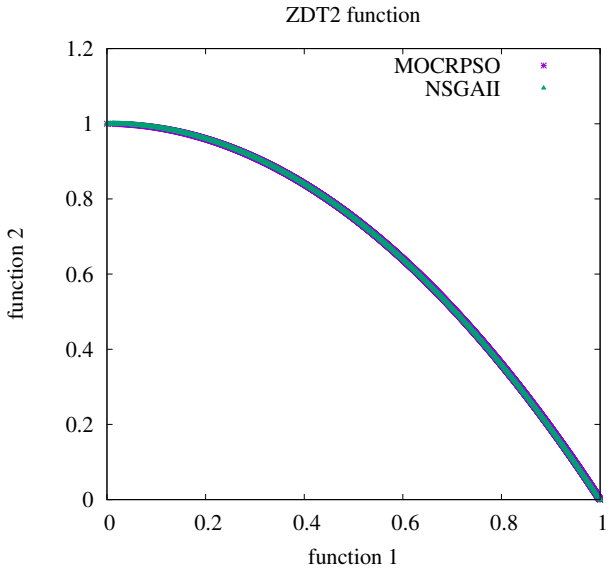


Figura 29 – Diferença entre a melhor curva EAF dos algoritmos MOCRPSO e NSGA-II para a função ZDT2. Hipervolume normalizado MOCRPSO: 0.333322 e NSGA-II: 0.333190.

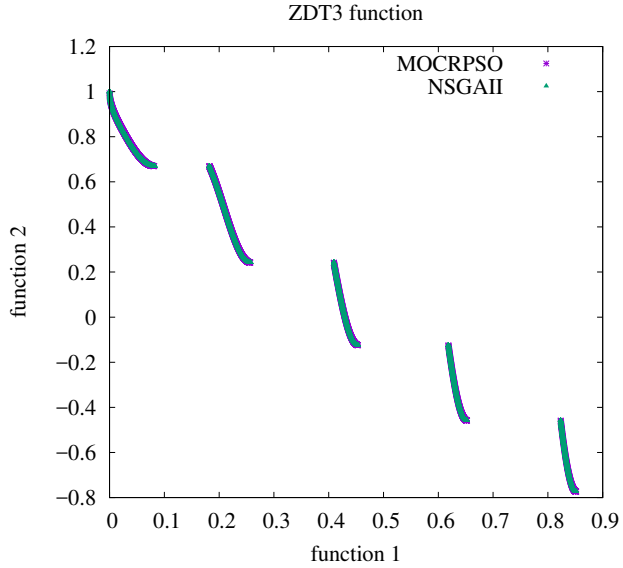


Figura 30 – Diferença entre a melhor curva EAF dos algoritmos MOCRPSO e NSGA-II para a função ZDT3. Hipervolume normalizado MOCRPSO: 0.517434 e NSGA-II: 0.517388.

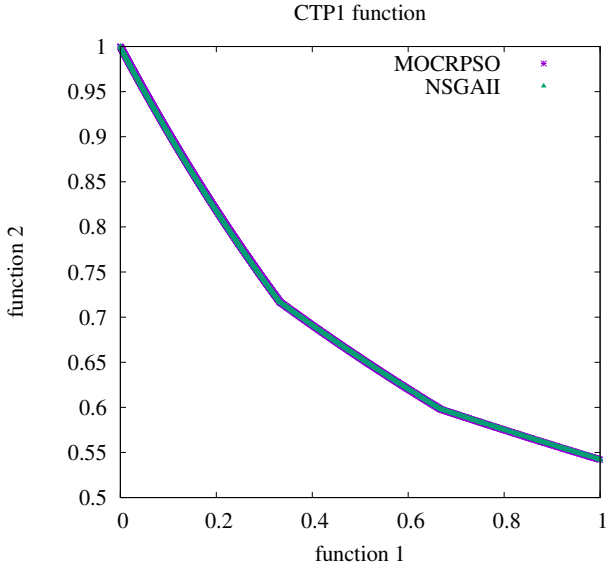


Figura 31 – Diferença entre a melhor curva EAF dos algoritmos MOCRPSO e NSGA-II para a função CTP1. Hipervolume normalizado MOCRPSO: 0.672642 e NSGA-II: 0.672413.

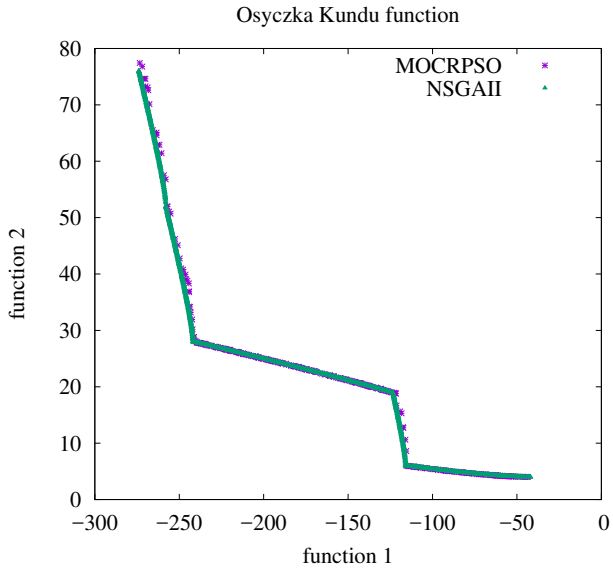


Figura 32 – Diferença entre a melhor curva EAF dos algoritmos MOCRPSO e NSGA-II para a função *Osyczka Kundu*. Hipervolume normalizado MOCRPSO: 0.755558 e NSGA-II: 0.763134.

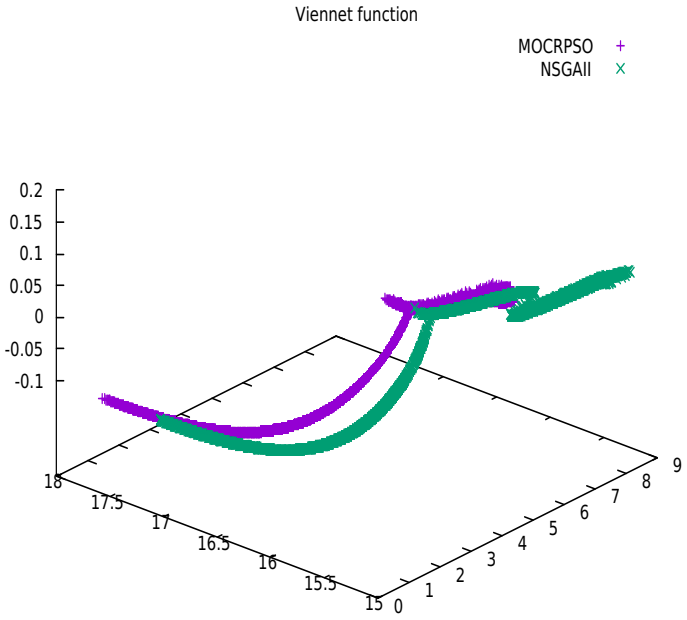


Figura 33 – Fronteira de Pareto (EAF *best*) dos algoritmos MOCRPSO e NSGA-II para a função *Viennet*. Hipervolume normalizado MOCRPSO: 0.756815 e NSGA-II: 0.874023.

6.2 SEGUNDO CONJUNTO DE EXPERIMENTOS

O segundo conjunto de experimentos consiste em encontrar um conjunto de variáveis de projeto que correspondem às áreas da seções transversais das barras $(A_1, \ldots, A_{n_a})$ de uma estrutura, que minimizam o seu peso e o máximo deslocamento dos nós, sujeito a restrições de tensões normais máximas. O problema pode ser formulado como

$$\begin{aligned} &\min \quad W = \sum_{i=1}^{n_a} \rho A_i L_i \quad \text{e} \\ &\text{máximo}(|u_j|), \; j = 1, \ldots, n_d, \end{aligned} \tag{6.7}$$

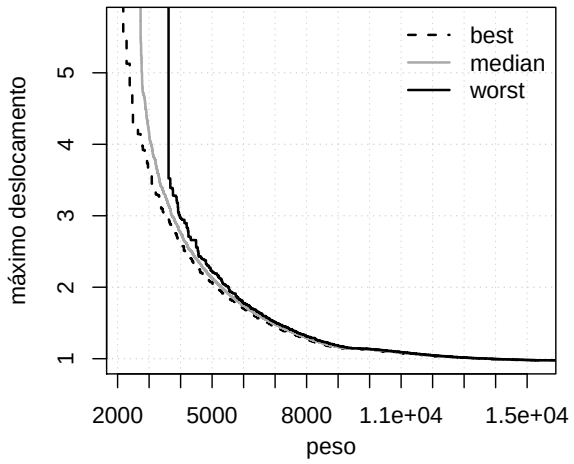
sujeito a

$$\begin{aligned} &\frac{\sigma_{il}}{\bar{\sigma}} - 1 \leq 0, \\ &i = 1, \ldots, n_a, \quad l = 1, \ldots, n_l, \end{aligned} \tag{6.8}$$

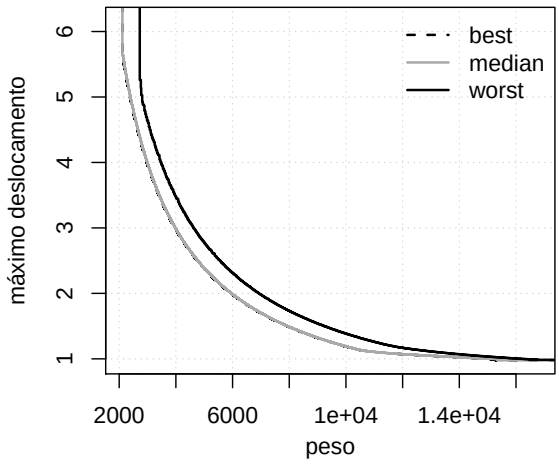
onde ρ é a massa específica, L é o comprimento da barra, u é o deslocamento de cada nó e σ é a restrição de tensão. As variáveis n_a, n_d e n_l representam, respectivamente, o número de variáveis de projeto, graus de liberdade e casos de carregamento.

6.2.1 **Análise dos resultados**

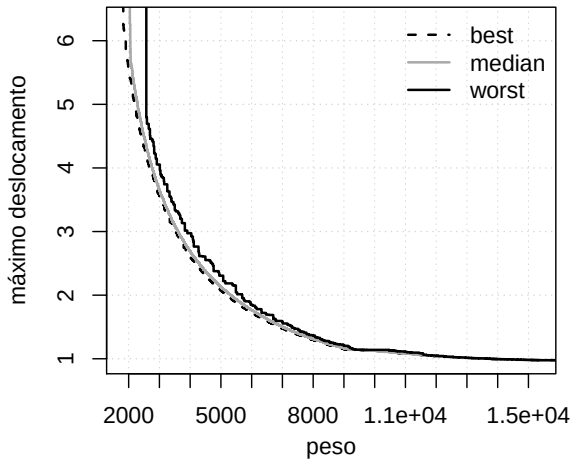
As curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para as treliças de 10, 25, 60, 72 e 942 barras com restrições de cardinalidade para os casos contínuo e discreto são apresentadas nas Figs. 34 a 42. É possível observar que a curvas apresentadas em cada gráfico são muito semelhantes entre si. Entretanto, nas treliças com maior número de barras (treliças de 60 e 942 barras, por exemplo), a diferença entre tais curvas foi mais aparente.



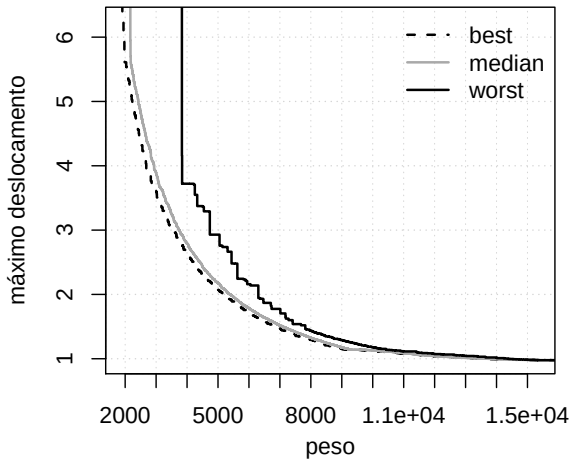
(a) sem r.c.



(b) $m = 2$

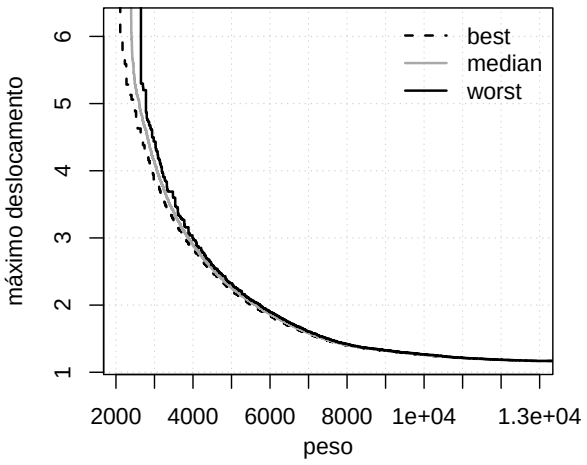


(c) $m = 4$

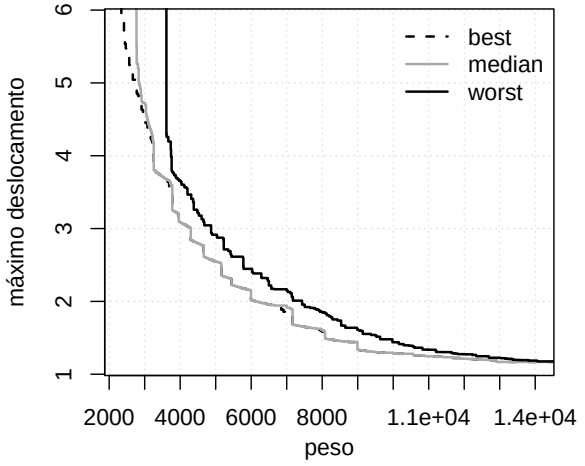


(d) $m = 8$

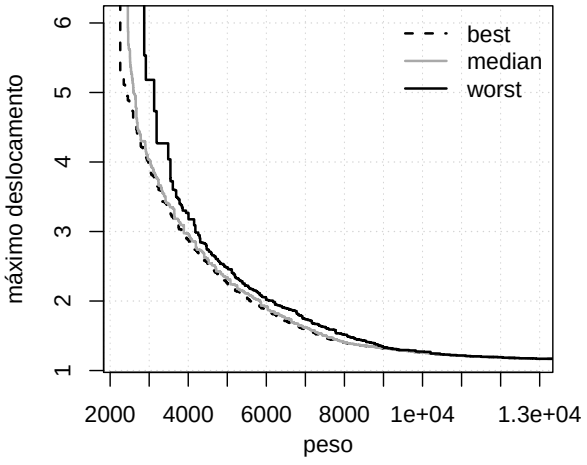
Figura 34 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 10 barras caso contínuo.



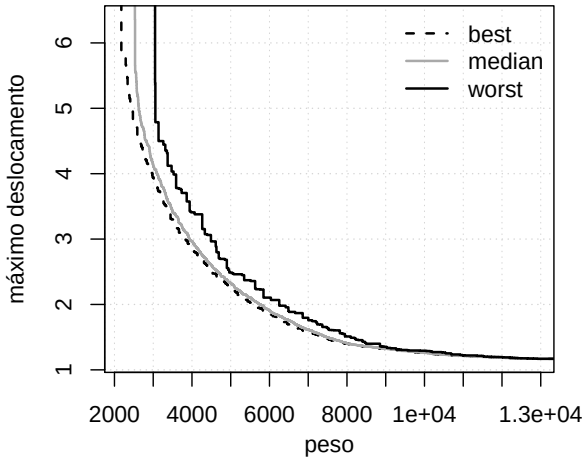
(a) sem r.c.



(b) $m = 2$

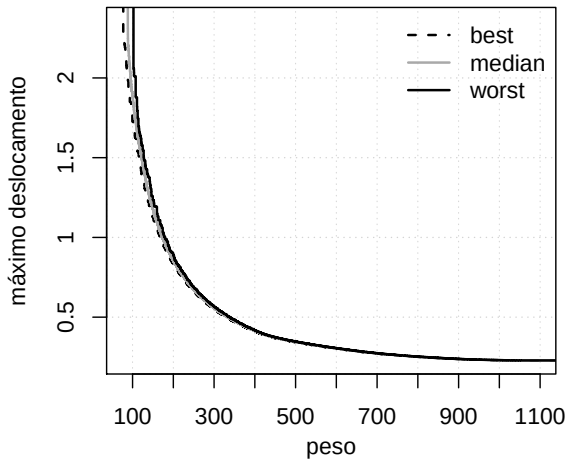


(c) $m = 4$

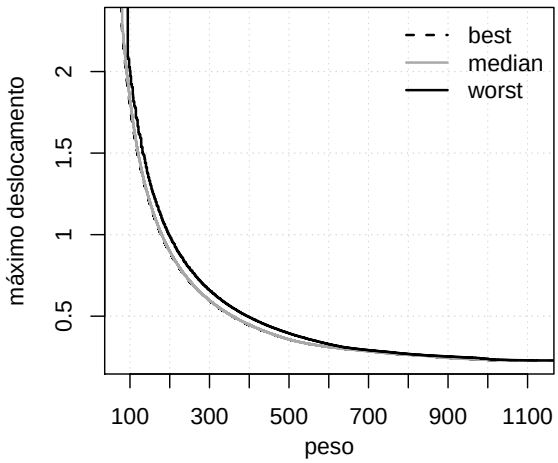


(d) $m = 8$

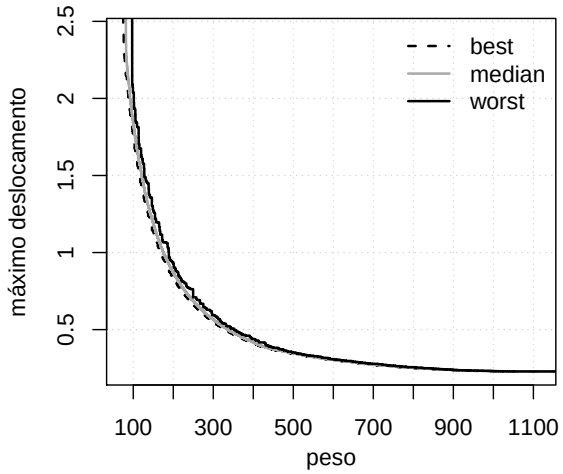
Figura 35 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 10 barras caso discreto.



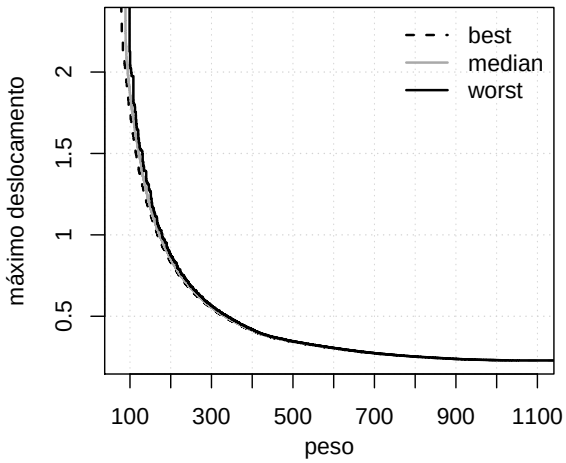
(a) sem r.c.



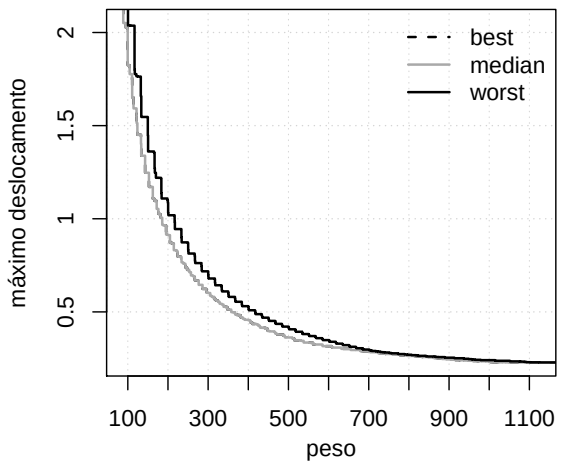
(b) $m = 2$



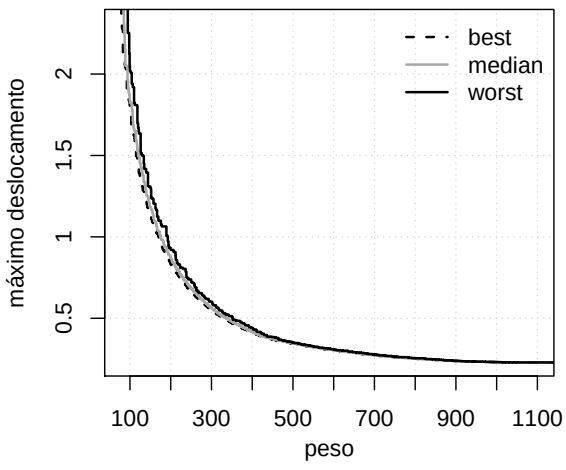
(c) $m = 4$



(d) sem r.c.

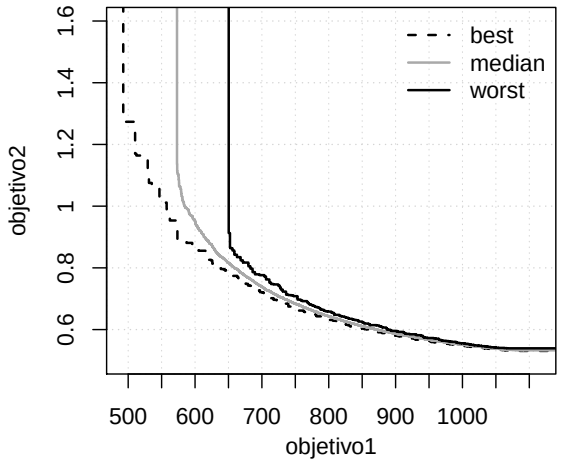


(e) $m = 2$

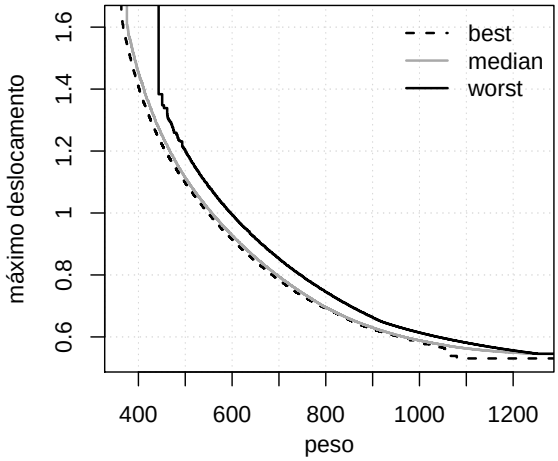


(f) $m = 4$

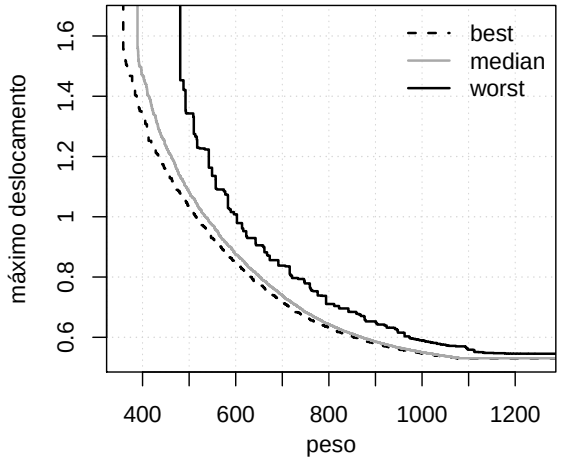
Figura 36 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para as treliças de 25 barras caso contínuo (figuras a, b, c) e caso discreto (figuras d, e, f).



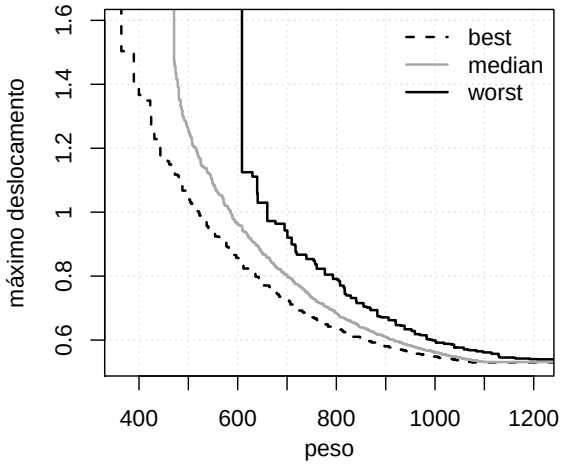
(a) sem r.c.



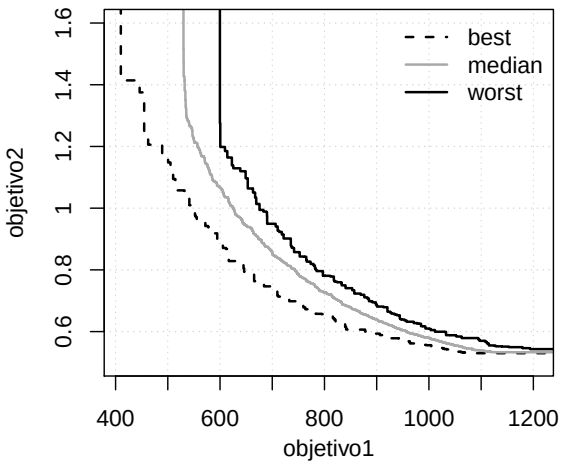
(b) $m = 2$



(c) $m = 4$

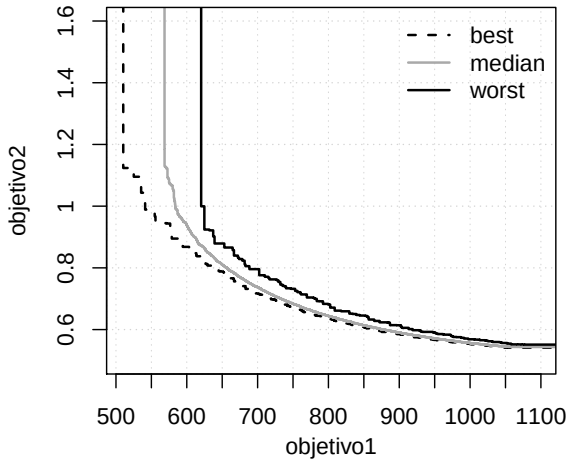


(d) $m = 8$

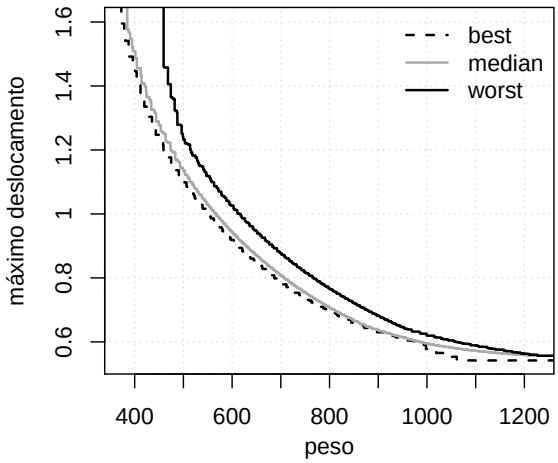


(e) $m = 16$

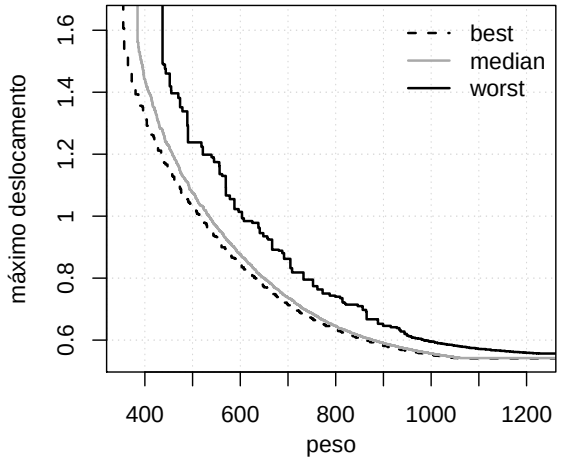
Figura 37 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 60 barras caso contínuo.



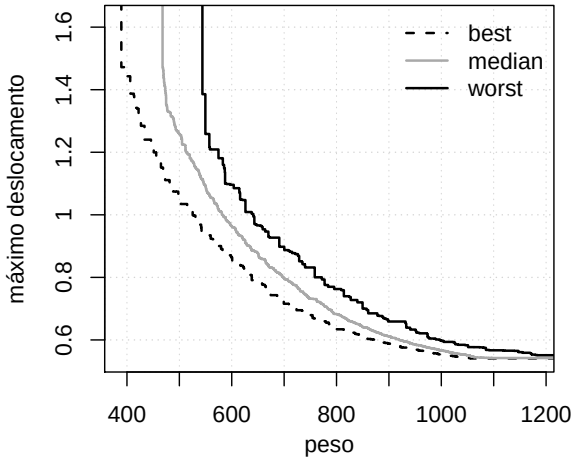
(a) sem r.c.



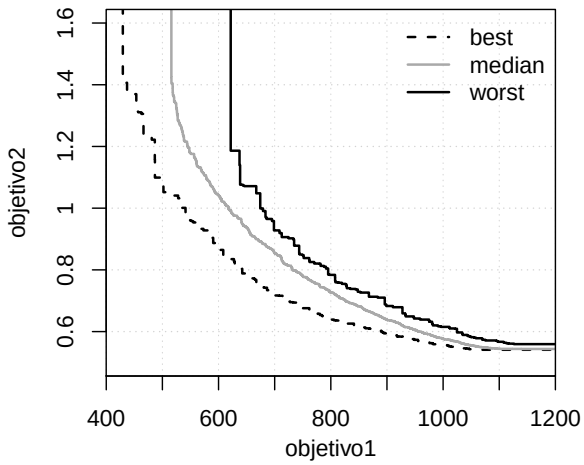
(b) $m = 2$



(c) $m = 4$

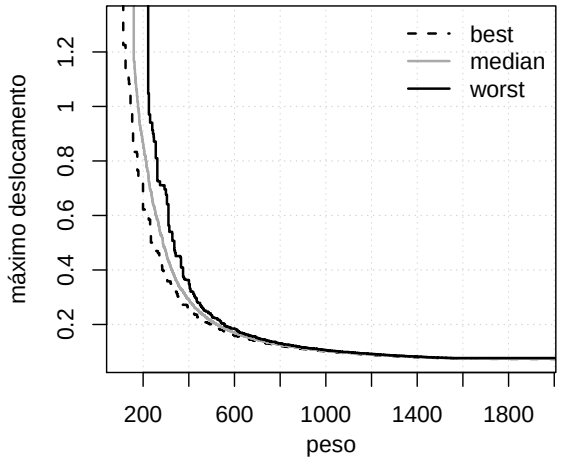


(d) $m = 8$

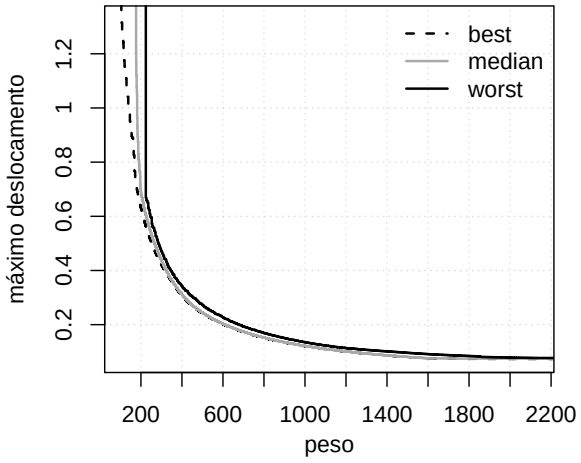


(e) $m = 16$

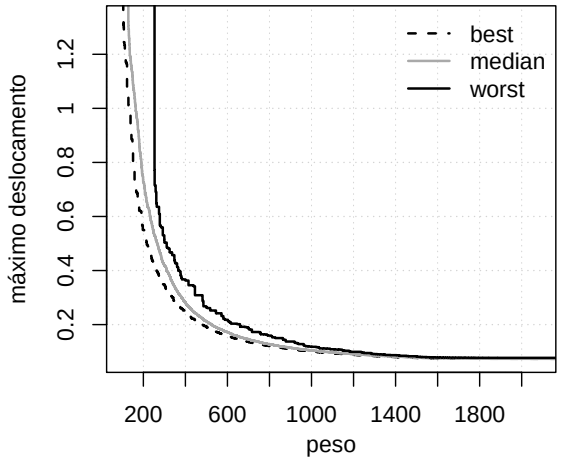
Figura 38 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 60 barras caso discreto.



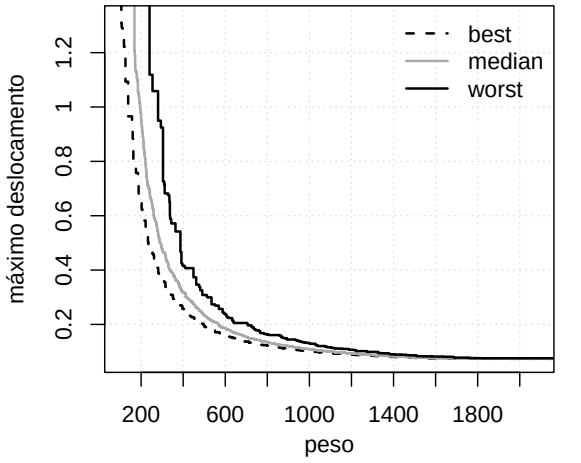
(a) sem r.c.



(b) $m = 2$

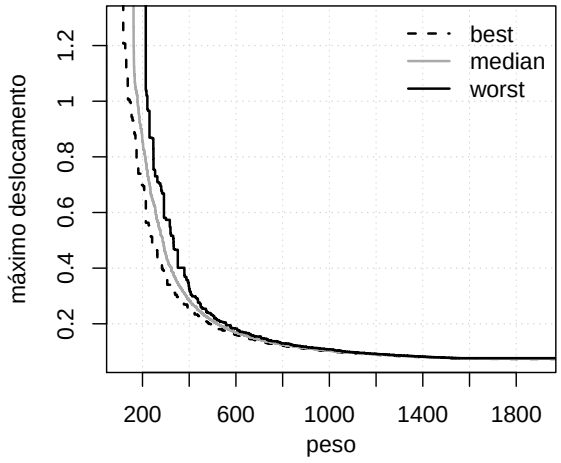


(c) $m = 4$

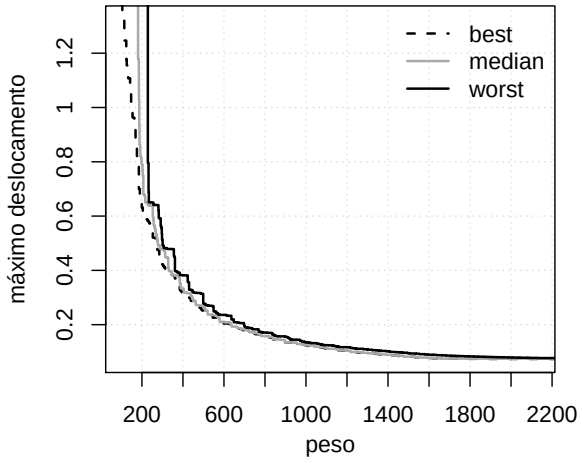


(d) $m = 8$

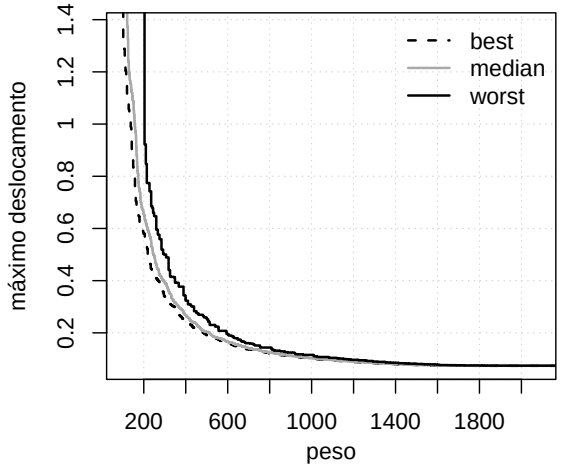
Figura 39 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 72 barras caso contínuo.



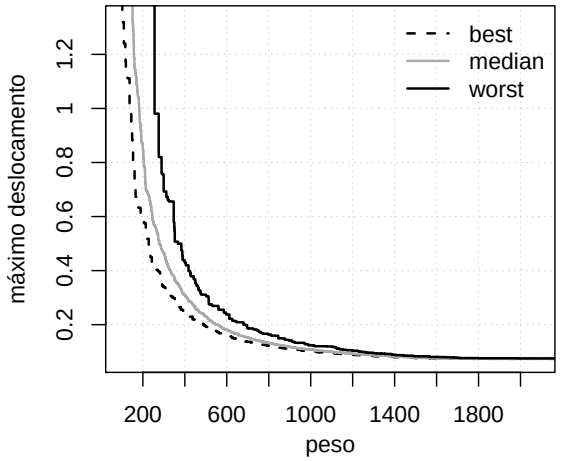
(a) sem r.c.



(b) $m = 2$

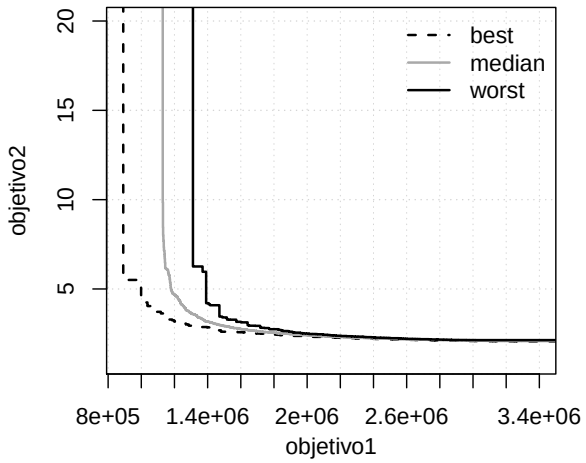


(c) $m = 4$

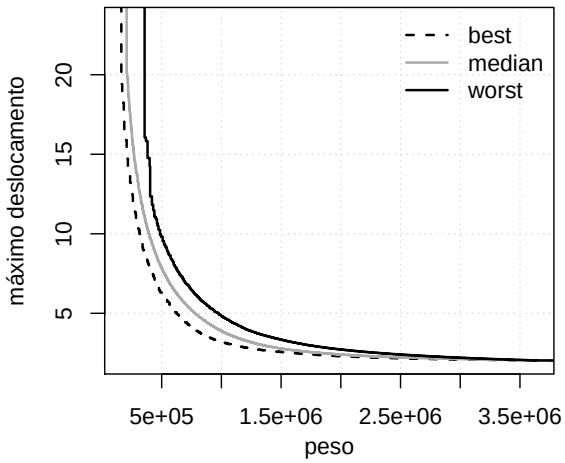


(d) $m = 8$

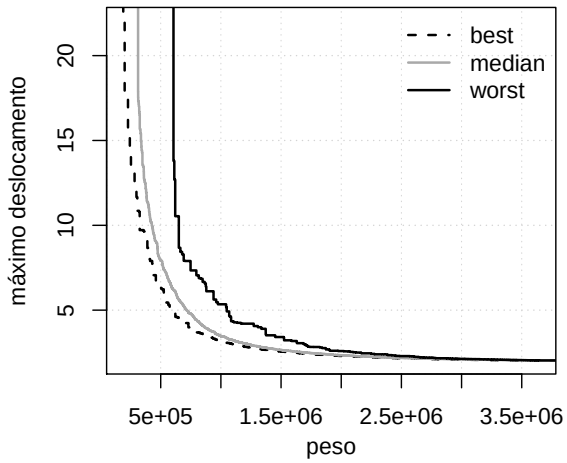
Figura 40 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 72 barras caso discreto.



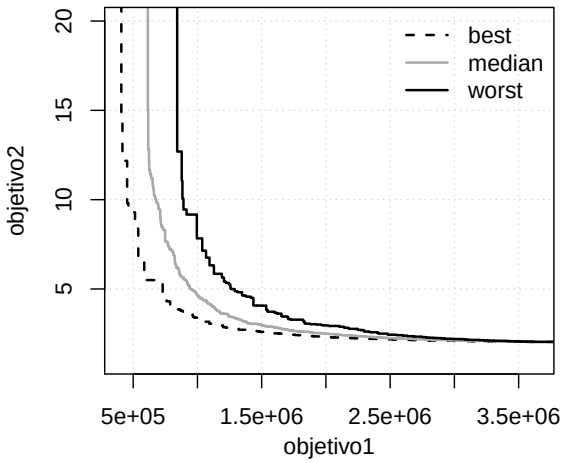
(a) sem r.c.



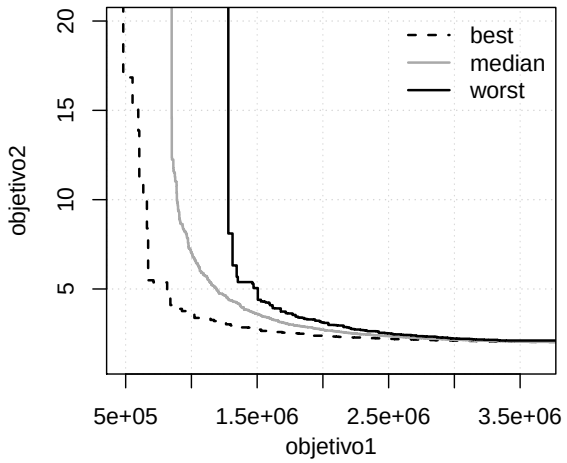
(b) $m = 2$



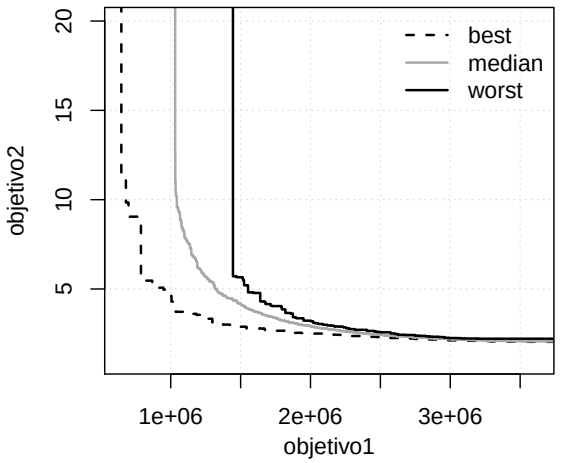
(c) $m = 4$



(d) $m = 8$

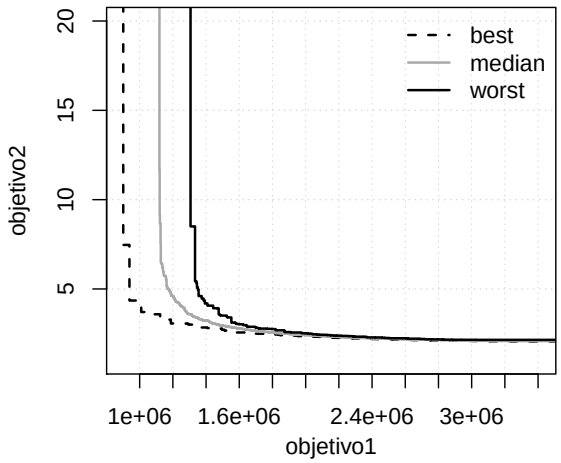


(e) $m = 16$

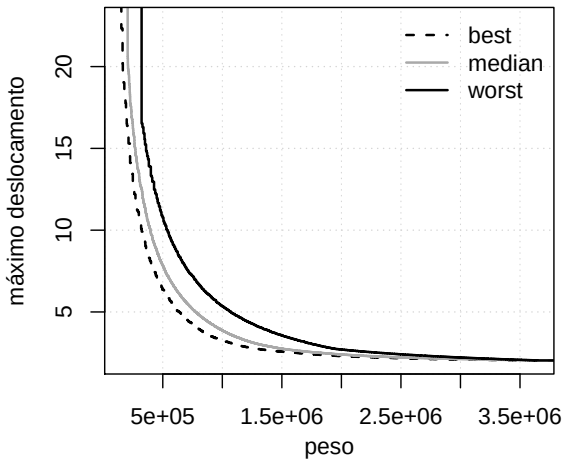


(f) $m = 32$

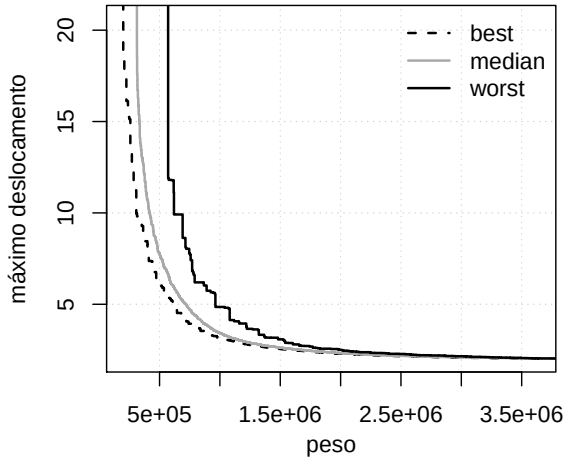
Figura 41 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 942 barras caso contínuo.



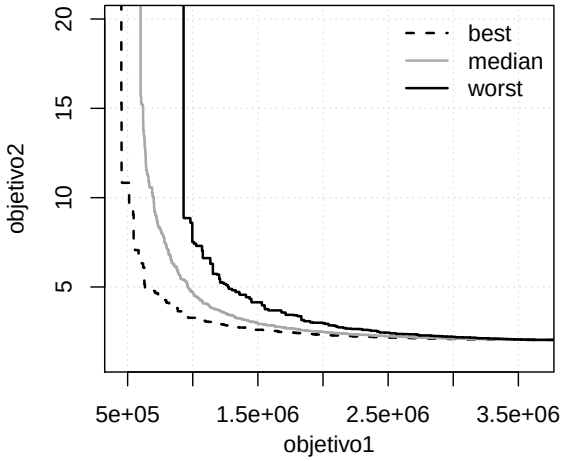
(a) sem r.c.



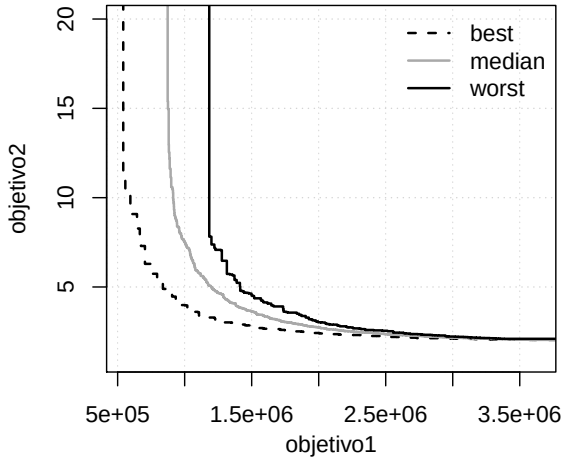
(b) $m = 2$



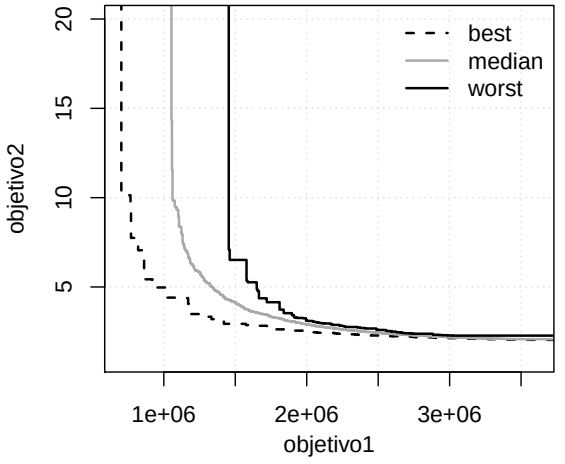
(c) $m = 4$



(d) $m = 8$



(e) $m = 16$



(f) $m = 32$

Figura 42 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 942 barras caso discreto.

Uma comparação envolvendo esses resultados ainda no segundo conjunto de experimentos foi produzida utilizando dois pares de algoritmos multiobjetivo encontrados na literatura. Inicialmente, as cinco treliças foram comparadas com os resultados dos algoritmos GDE3 e GDE3-APM, apresentados por Vargas *et al.* [6], utilizando variáveis contínuas e discretas sem restrições de cardinalidade. Em seguida, as mesmas estruturas utilizando somente variáveis discretas e com restrições de cardinalidade foram comparadas com os resultados dos algoritmos MOAS e MOACS, apresentados por Angelo *et al.* [14]. Para cada grupo de algoritmos serão apresentadas as curvas EAF comparativas. É importante reforçar que o orçamento computacional utilizado em ambos os pares de algoritmos foi o mesmo utilizado pelo MOCRPSO.

6.2.1.1 Comparação I

As Figs. 46 a 50 apresentam as curvas EAF *best*, *median* e *worst* comparativas dos algoritmos GDE3, GDE3-APM e MOCRPSO e seus respectivos valores de hipervolume normalizados. Além disso, o resultado mono-objetivo de cada problema também é apresentado nos gráficos. Esses resultados foram extraídos de [15] e correspondem à melhor solução do problema de otimização mono-objetivo (para o caso de minimização do peso da estrutura) e seu respectivo deslocamento máximo permitido. A Tabela 10 apresenta os valores obtidos da literatura para o peso e o máximo deslocamento para cada treliça.

Tabela 10 – Valores referentes ao peso e ao máximo deslocamento para o problema de otimização mono-objetivo.

Problema	peso | deslocamento
Treliça de 10 barras (contínuo)	5060.85 | 2
Treliça de 10 barras (discreto)	5490.74 | 2
Treliça de 25 barras (contínuo)	484.05 | 0.35
Treliça de 25 barras (discreto)	484.85 | 0.35
Treliça de 60 barras (contínuo)	308.85 | 1.75
Treliça de 60 barras (discreto)	312.71 | 1.75
Treliça de 72 barras (contínuo)	379.62 | 0.25
Treliça de 72 barras (discreto)	385.54 | 0.25
Treliça de 942 barras (contínuo)	147.57 | 15
Treliça de 942 barras (discreto)	141.24 | 15

Para a treliça de 10 barras casos contínuo e discreto (Fig. 46), as curvas EAF *best* dos três algoritmos foram capazes de alcançar os valores do problema mono-objetivo. Os valores do hipervolume apresentados pelo MOCRPSO são inferiores comparados aos valores apresentados pelos algoritmos GDE3 e GDE3-APM. As curvas EAF *best*, *median* e *worst* da treliça de 25 barras casos contínuo e discreto (Fig. 47) também alcançaram os valores

encontrados do problema mono-objetivo. Os valores do hipervolume apresentados pelo MOCRPSO para o caso contínuo são competitivos em relação aos outros dois algoritmos. Para o caso discreto, os valores são iguais, diferenciando-se apenas na terceira casa decimal.

Os casos contínuo e discreto das curvas EAF da treliça de 60 barras (Fig. 48) não foram capazes de alcançar os valores encontrados no problema mono-objetivo. O MOCRPSO não apresentou um desempenho satisfatório para esta estrutura quando comparado ao GDE3 e GDE3-APM, o que pode ser confirmado pelos valores de hipervolume apresentados pelos algoritmos. Para a treliça de 72 barras casos contínuo e discreto (Fig. 49), as curvas EAF *best* dos algoritmos GDE3 e GDE3-APM foram capazes de alcançar os valores do problema mono-objetivo. O algoritmo MOCRPSO nesta estrutura obteve melhores valores de hipervolume para seu caso discreto.

Por fim, as curvas EAF da treliça de 942 barras (Fig. 50) em ambos os casos e nos três algoritmos analisados não foram capazes de alcançar os valores obtidos no problema mono-objetivo. Em relação aos valores de hipervolume, o MOCRPSO não apresentou bons resultados quando comparado aos algoritmos GDE3 e GDE3-APM.

As Figs. 43, 44 e 45 apresentam os perfis de desempenho dos três algoritmos analisados no primeiro grupo de comparações para curvas EAF *best*, *median* e *worst*, respectivamente. Nas duas primeiras figuras, o GDE3 obteve um desempenho superior que os demais algoritmos, alcançando o menor valor de τ, tal que $\rho(\tau) = 1$. Na Fig. 45, o GDE3-APM obteve o menor valor de τ, tal que $\rho(\tau) = 1$. As Tabelas 15, 16 e 17 apresentam o valor da área normalizada sob as curvas dos perfis de desempenho. O GDE3 alcançou o maior valor da área na análise para as curvas EAF *best* e *median* e o GDE3-APM o maior valor para a curva *worst.*

O MOCRPSO não obteve um desempenho superior comparado aos algoritmos GDE3 e GDE3-APM. Contudo, ele mostrou ser um bom algoritmo obtendo valores próximos aos dos dois algoritmos. É importante mencionar que a Evolução Diferencial (*Differential Evolution* - DE), base dos algoritmos multiobjetivo GDE3 e GDE3-APM, é uma técnica de otimização bastante robusta e largamente testada na literatura, apresentando sempre bons resultados [164, 6, 165].

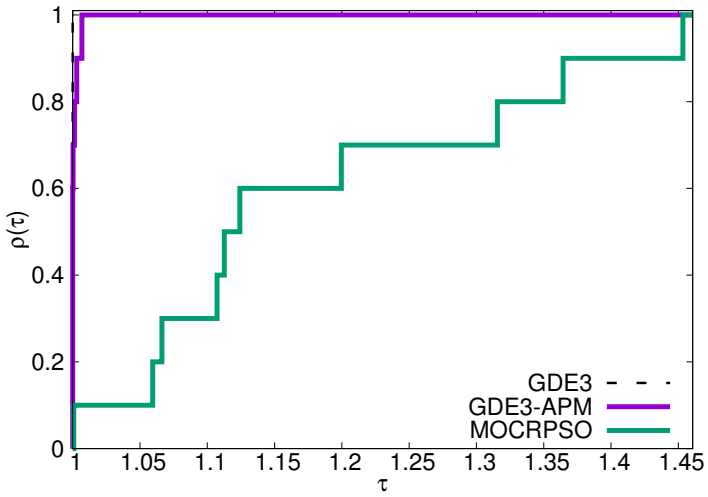


Figura 43 – Perfis de desempenho dos algoritmos GDE3, GDE3-APM e MOCRPSO para a curva EAF *best*.

Tabela 11 – Área normalizada sob as curvas dos perfis de desempenho do algoritmos GDE3, GDE3-APM e MOCRPSO para a curva EAF *best*.

Algoritmo	GDE3	GDE3-APM	MOCRPSO
Área	1.0	0.9974	0.6022

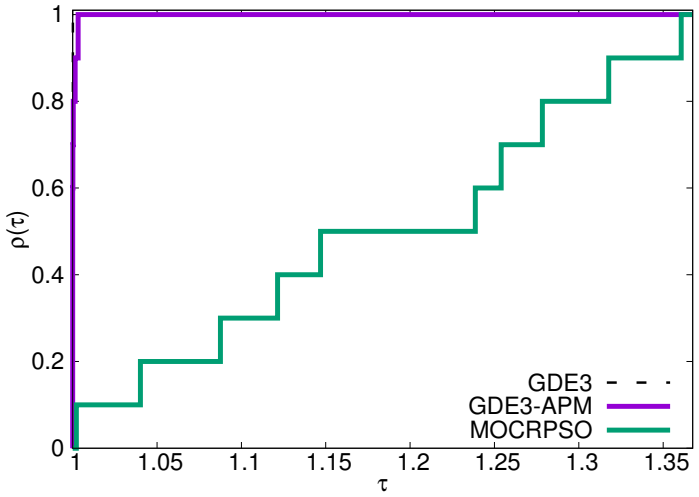


Figura 44 – Perfis de desempenho dos algoritmos GDE3, GDE3-APM e MOCRPSO para a curva EAF *median*.

Tabela 12 – Área normalizada sob as curvas dos perfis de desempenho do algoritmos GDE3, GDE3-APM e MOCRPSO para a curva EAF *median*.

Algoritmo	GDE3	GDE3-APM	MOCRPSO
Área	1.0	0.9986	0.4877

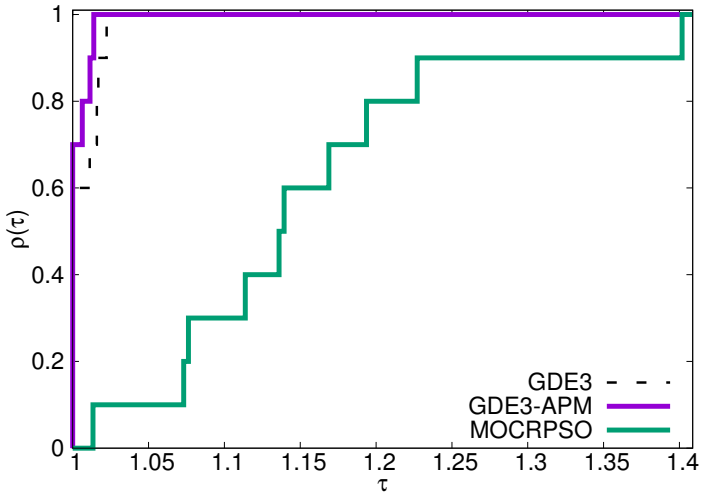


Figura 45 – Perfis de desempenho dos algoritmos GDE3, GDE3-APM e MOCRPSO para a curva EAF *worst*.

Tabela 13 – Área normalizada sob as curvas dos perfis de desempenho do algoritmos GDE3, GDE3-APM e MOCRPSO para a curva EAF *worst*.

Algoritmo	GDE3	GDE3-APM	MOCRPSO
Área	0.9911	1.0	0.6206

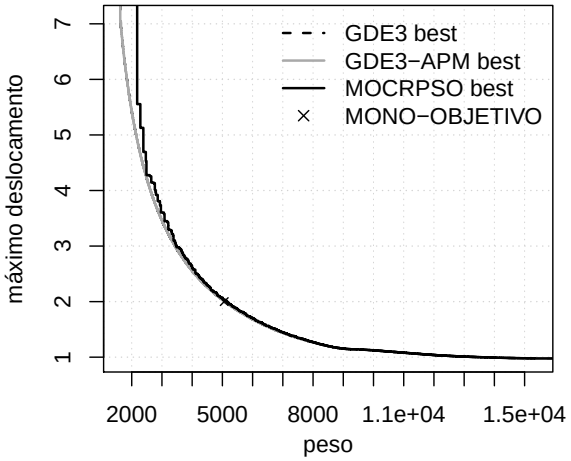
A Tabela 14 apresenta a média e o desvio padrão dos valores de hipervolumes apresentados nas Figs. 34 a 42 para os algoritmos GDE3, GDE3-APM e MOCRPSO. O sinal (+) indica que a diferença entre as medidas calculadas para cada uma das execuções realizadas é estatisticamente significativa (p-valor ≤ 0.05) quando comparado com os mesmos resultados obtidos pelo algoritmo com a melhor média. Para cada problema, o melhor resultado está destacado em negrito. As melhores médias foram obtidos pelos algoritmos GDE3 e GDE3-APM.

Tabela 14 – Média e desvio padrão dos resultados dos hipervolumes para a comparação I.

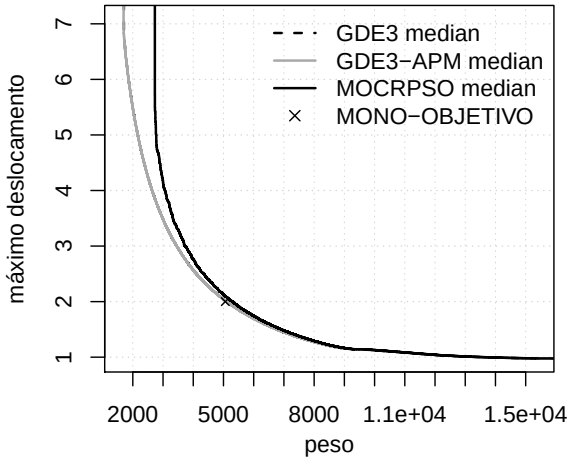
	GDE3	GDE3-APM	MOCRPSO
Treliça de 10 barras (contínuo)			
Média	0.866240	**0.866590**	0.829380(+)
Desvio padrão	0.002987	0.000535	0.009305
Treliça de 10 barras (discreto)			
Média	0.839910	**0.840190**	0.823630(+)
Desvio padrão	0.002142	0.000588	0.002749
Treliça de 25 barras (contínuo)			
Média	**0.892020**	**0.892020**	0.885290(+)
Desvio padrão	0.000123	0.000123	0.001208
Treliça de 25 barras (discreto)			
Média	**0.888690**	0.888680	0.883440(+)
Desvio padrão	0.000066	0.000065	0.000792
Treliça de 60 barras (contínuo)			
Média	0.760360	**0.760510**	0.612030(+)
Desvio padrão	0.004413	0.005238	0.021632
Treliça de 60 barras (discreto)			
Média	**0.758710**	0.757560	0.601130(+)
Desvio padrão	0.005358	0.006547	0.021127
Treliça de 72 barras (contínuo)			
Média	**0.921280**	0.921260	0.889270(+)
Desvio padrão	0.002617	0.002474	0.007005
Treliça de 72 barras (discreto)			
Média	**0.919360**	**0.919360**	0.888180(+)
Desvio padrão	0.001484	0.001516	0.006113
Treliça de 942 barras (contínuo)			
Média	0.927590	**0.927910**	0.696200(+)
Desvio padrão	0.008693	0.007107	0.025394
Treliça de 942 barras (discreto)			
Média	**0.939420**	0.937610	0.702780(+)
Desvio padrão	0.009193	0.007186	0.025959

6.2.1.2 Comparação II

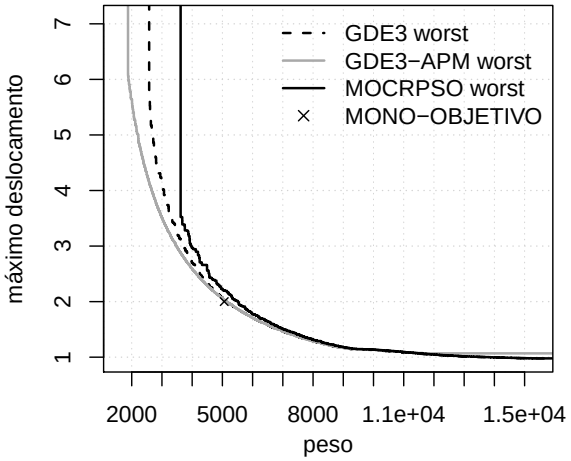
No segundo grupo de comparações, as Figs. 46 a 50 apresentam as curvas EAF *best*, *median* e *worst* comparativas entre os algoritmos MOACS, MOAS e MOCRPSO e seus respectivos valores de hipervolume normalizados. Não foi possível adicionar nos gráficos o resultado mono-objetivo de cada problema, pois até então não existe na literatura trabalhos que apresentam os valores mono-objetivo para todos os casos apresentados. O valor da cardinalidade (m) utilizada está indicado na legenda das figuras. Vale lembrar que os resultados apresentados referem-se apenas ao caso discreto, pois a referência utilizada na comparação apresentou somente resultados para este caso.



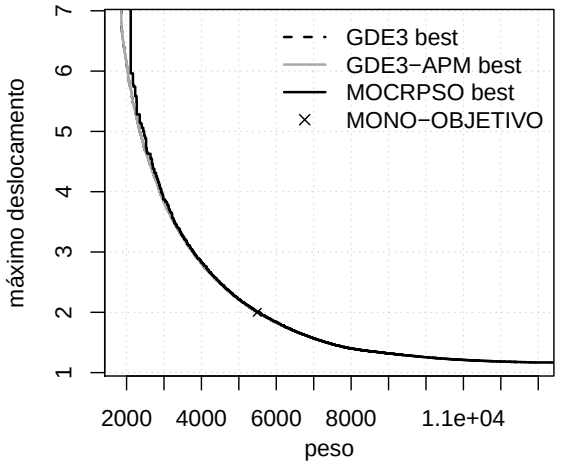
(a) $H_1 = 0.8678$, $H_2 = 0.8678$ e $H_3 = 0.7234$



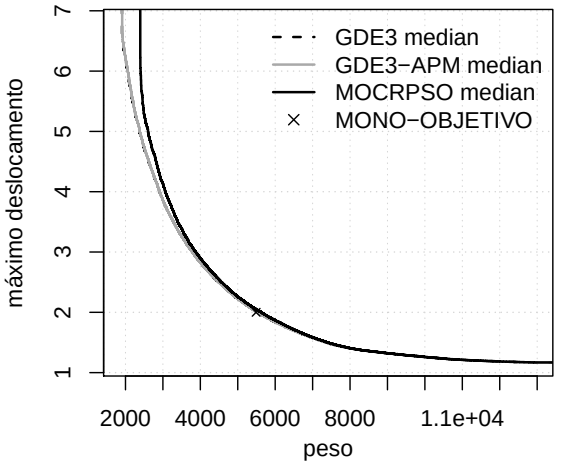
(b) $H_1 = 0.8700$, $H_2 = 0.8701$ e $H_3 = 0.6603$



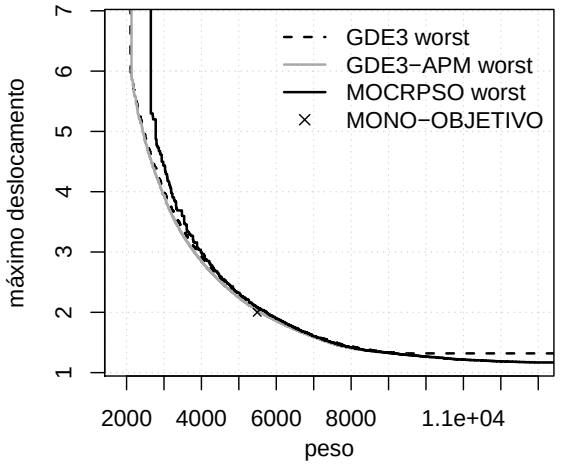
(c) $H_1 = 0.8314$, $H_2 = 0.8500$ e $H_3 = 0.7482$



(d) $H_1 = 0.8420$, $H_2 = 0.8420$ e $H_3 = 0.7490$

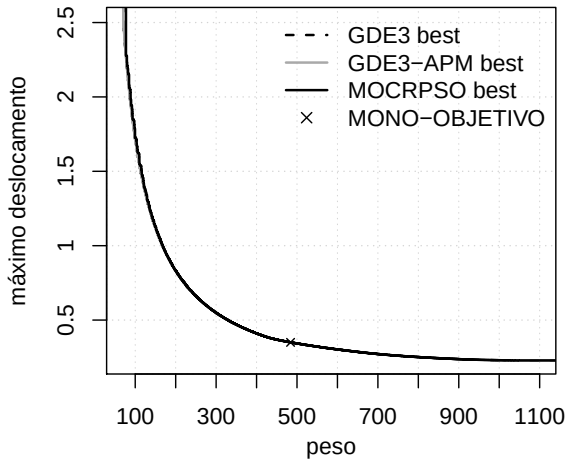


(e) $H_1 = 0.8394$, $H_2 = 0.8391$ e $H_3 = 0.7718$

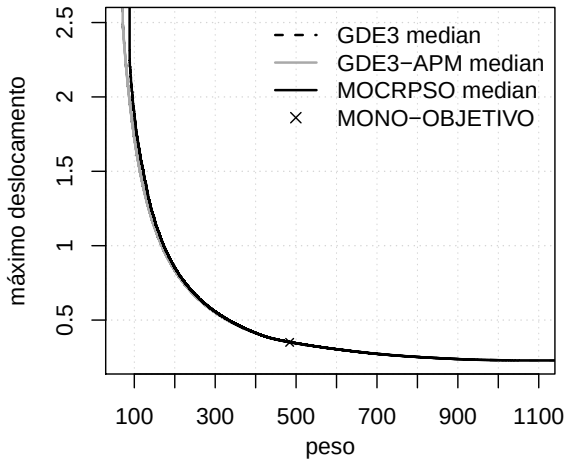


(f) $H_1 = 0.8208$, $H_2 = 0.8339$ e $H_3 = 0.7134$

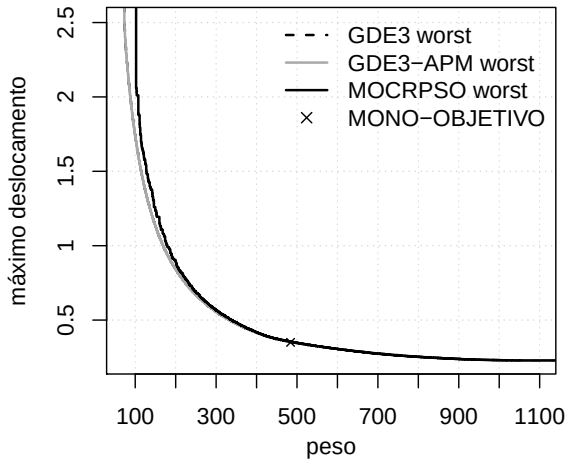
Figura 46 – Curvas EAF da treliça de 10 barras caso contínuo (figuras a, b, c) e discreto (figuras d, e, f). Valores de hipervolume normalizados: H_1 para GDE3, H_2 para GDE3-APM e H_3 para MOCRPSO.



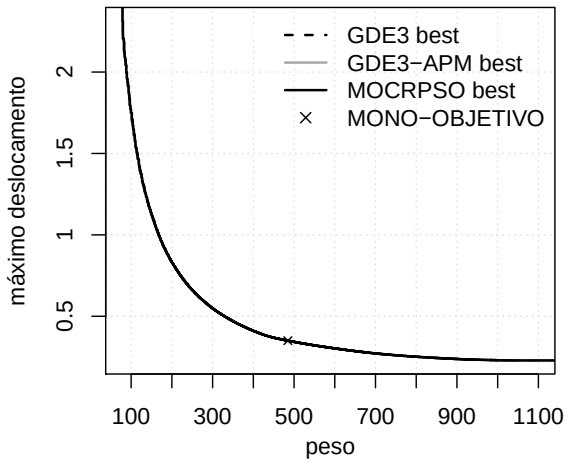
(a) $H_1 = 0.8876$, $H_2 = 0.8876$ e $H_3 = 0.8324$



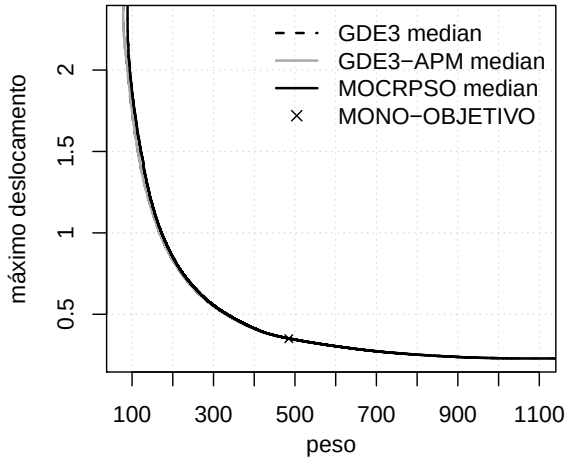
(b) $H_1 = 0.8875$, $H_2 = 0.8875$ e $H_3 = 0.7738$



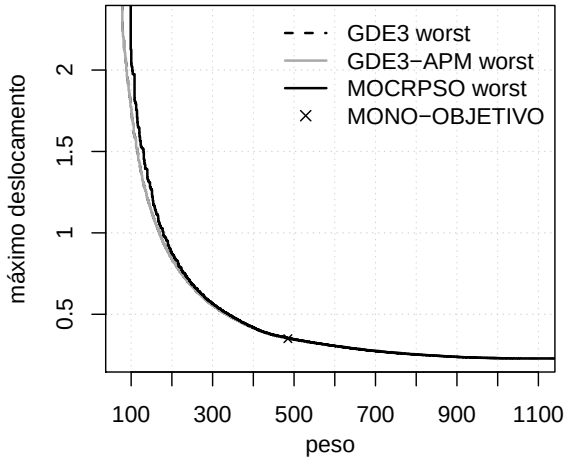
(c) $H_1 = 0.8873$, $H_2 = 0.8873$ e $H_3 = 0.8244$



(d) $H_1 = 0.8834$, $H_2 = 0.8834$ e $H_3 = 0.8827$

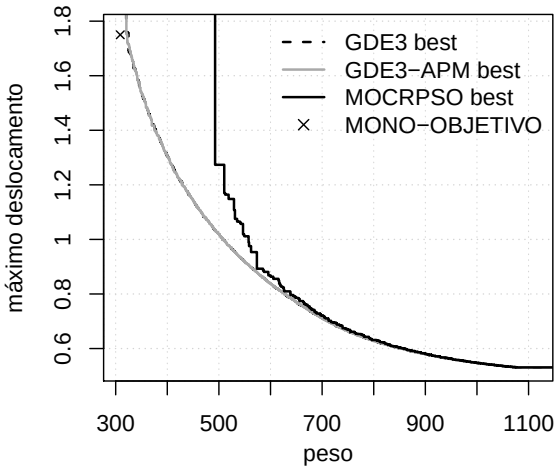


(e) $H_1 = 0.8830$, $H_2 = 0.8829$ e $H_3 = 0.8811$

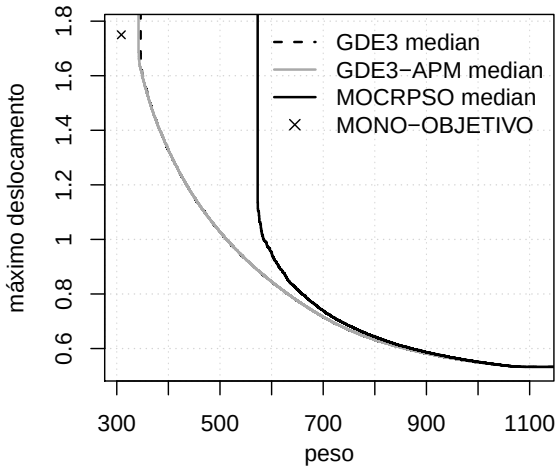


(f) $H_1 = 0.8778$, $H_2 = 0.8778$ e $H_3 = 0.8761$

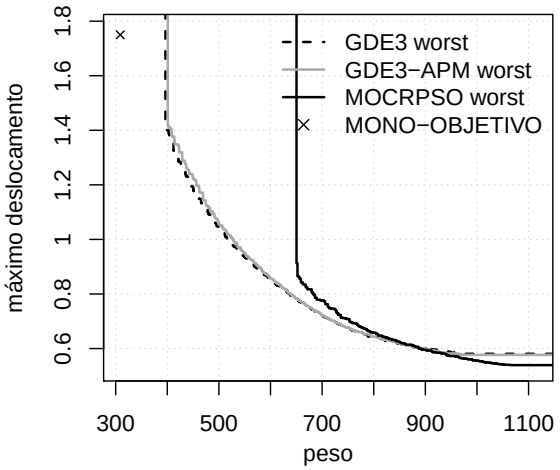
Figura 47 – Curvas EAF da treliça de 25 barras caso contínuo (figuras a, b, c) e discreto (figuras d, e, f). Valores de hipervolume normalizados: H_1 para GDE3, H_2 para GDE3-APM e H_3 para MOCRPSO.



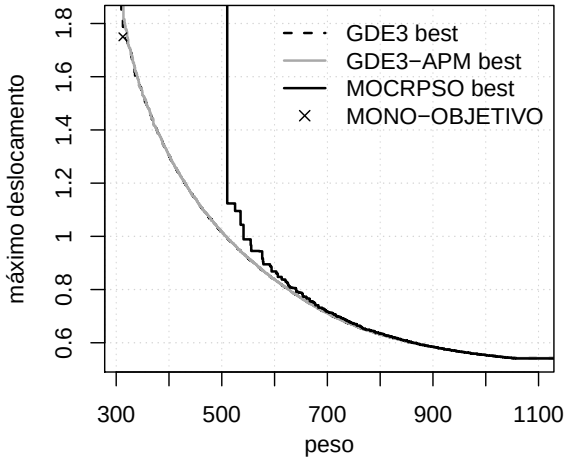
(a) $H_1 = 0.7564$, $H_2 = 0.7562$ e $H_3 = 0.5544$



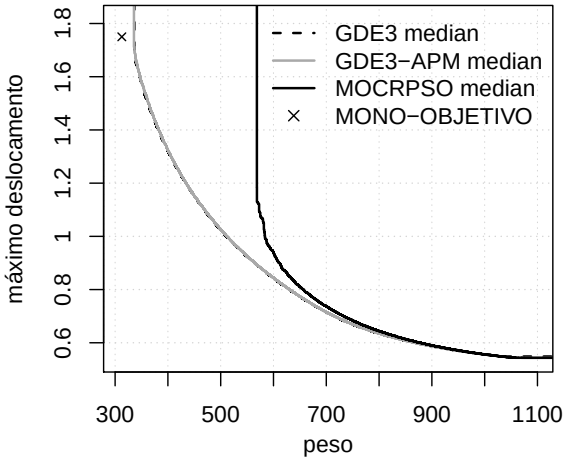
(b) $H_1 = 0.7370$, $H_2 = 0.7371$ e $H_3 = 0.5417$



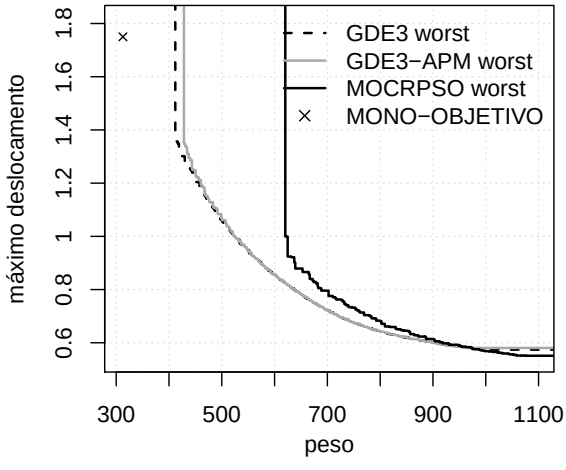
(c) $H_1 = 0.5987$, $H_2 = 0.5905$ e $H_3 = 0.4879$



(d) $H_1 = 0.7549$, $H_2 = 0.7539$ e $H_3 = 0.5194$

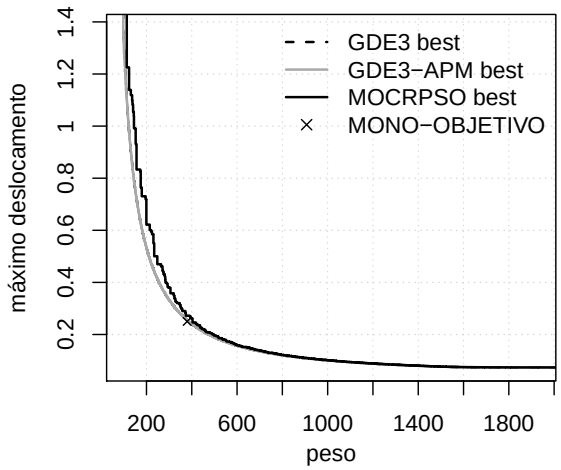


(e) $H_1 = 0.74401$, $H_2 = 0.7429$ e $H_3 = 0.5933$

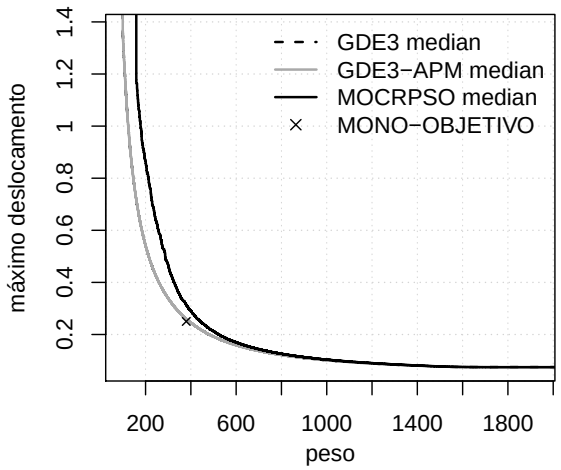


(f) $H_1 = 0.6176$, $H_2 = 0.6137$ e $H_3 = 0.4406$

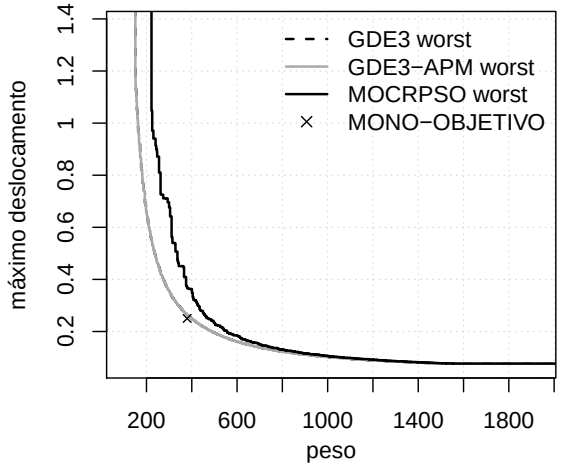
Figura 48 – Curvas EAF da treliça de 60 barras caso contínuo (figuras a, b, c) e discreto (figuras d, e, f). Valores de hipervolume normalizados: H_1 para GDE3, H_2 para GDE3-APM e H_3 para MOCRPSO.



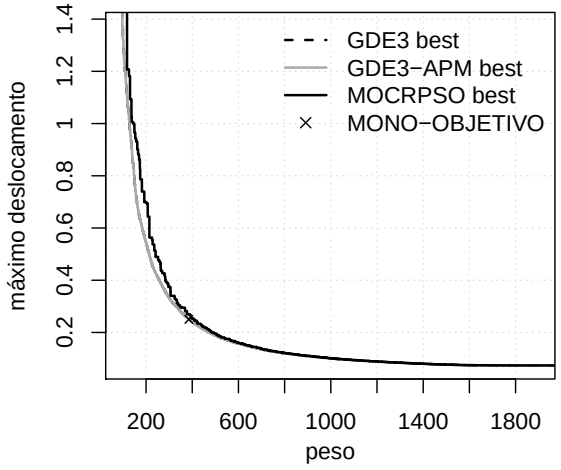
(a) $H_1 = 0.8805$, $H_2 = 0.8805$ e $H_3 = 0.6693$



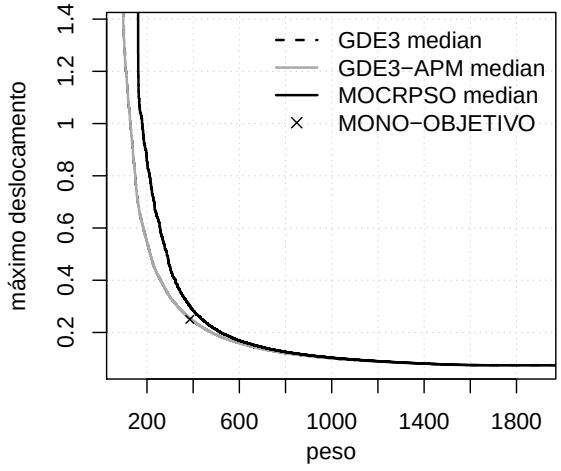
(b) $H_1 = 0.8500$, $H_2 = 0.8500$ e $H_3 = 0.6649$



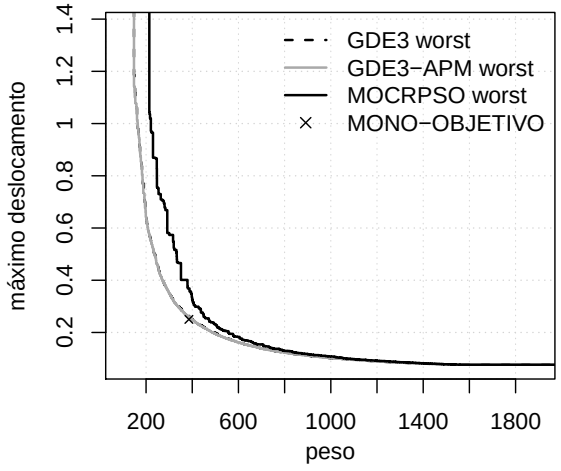
(c) $H_1 = 0.9037$, $H_2 = 0.9037$ e $H_3 = 0.8421$



(d) $H_1 = 0.9146$, $H_2 = 0.9146$ e $H_3 = 0.8633$

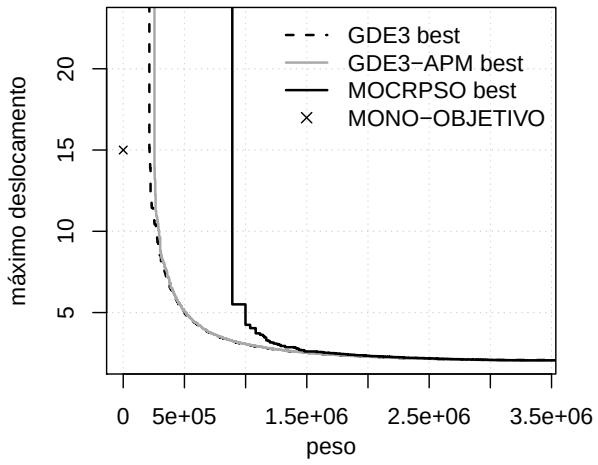


(e) $H_1 = 0.9148$, $H_2 = 0.9148$ e $H_3 = 0.8795$

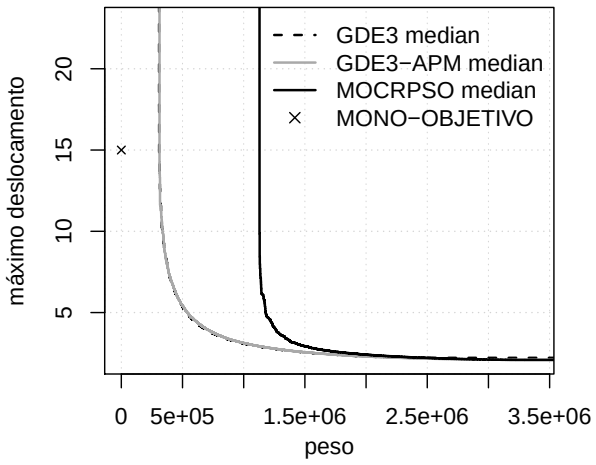


(f) $H_1 = 0.8979$, $H_2 = 0.8984$ e $H_3 = 0.7885$

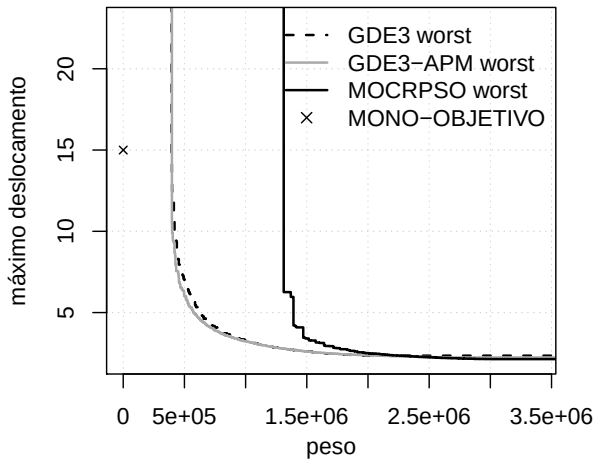
Figura 49 – Curvas EAF da treliça de 72 barras caso contínuo (figuras a, b, c) e discreto (figuras d, e, f). Valores de hipervolume normalizados: H_1 para GDE3, H_2 para GDE3-APM e H_3 para MOCRPSO.



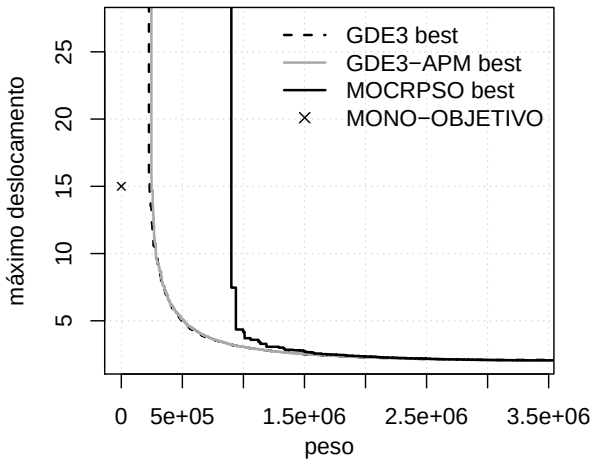
(a) $H_1 = 0.8782$, $H_2 = 0.8722$ e $H_3 = 0.7931$



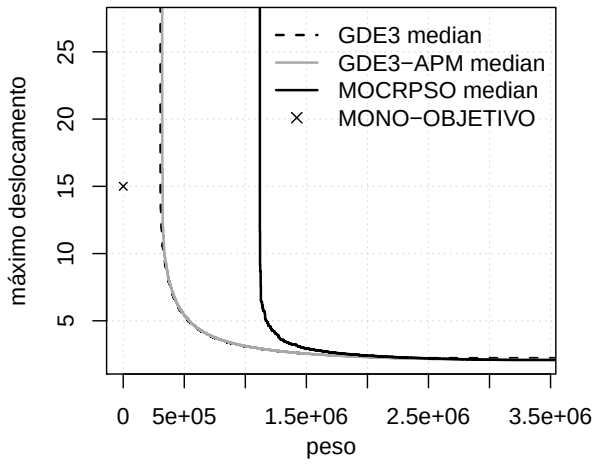
(b) $H_1 = 0.8515$, $H_2 = 0.8515$ e $H_3 = 0.7593$



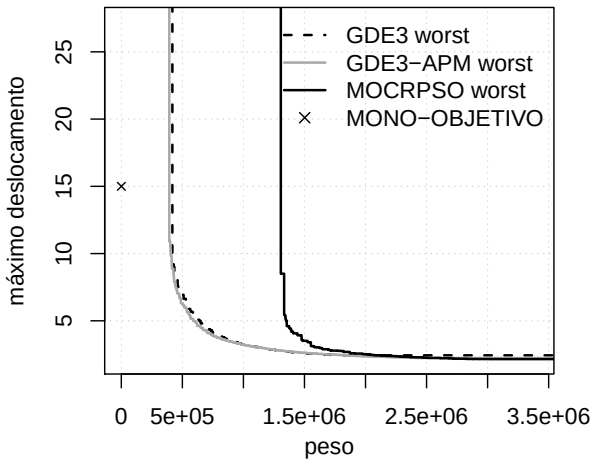
(c) $H_1 = 0.8081$, $H_2 = 0.8172$ e $H_3 = 0.7337$



(d) $H_1 = 0.9050$, $H_2 = 0.9023$ e $H_3 = 0.8134$



(e) $H_1 = 0.8377$, $H_2 = 0.8350$ e $H_3 = 0.6763$



(f) $H_1 = 0.6696$, $H_2 = 0.6809$ e $H_3 = 0.5704$

Figura 50 – Curvas EAF da treliça de 942 barras caso contínuo (figuras a, b, c) e discreto (figuras d, e, f). Valores de hipervolume normalizados: H_1 para GDE3, H_2 para GDE3-APM e H_3 para MOCRPSO.

Os resultados comparativos para a treliça de 10 barras sem restrição de cardinalidade, $m = 2$, 4 e 8 são apresentados nas Figs. 54 e 55. O MOCRPSO obteve valor de hipervolume superior aos dos algoritmos MOACS e MOAS para a curva *worst* sem restrição de cardinalidade e $m = 2$. Em alguns casos, o MOCRPSO obteve valores de hipervolume igual ou superior aos do algoritmo MOAS.

As curvas EAF *best*, *median* e *worst* comparativas para a treliça de 25 barras sem restrição de cardinalidade, $m = 2$ e 4 são apresentadas nas Figs. 56 e 57. Em todos os gráficos o MOCRPSO obteve valores de hipervolume competitivos ou superiores aos obtidos pelos algoritmos MOACS e MOAS.

Para a treliça de 60 barras, os gráficos com as curvas EAF *best*, *median* e *worst* comparativas sem restrição de cardinalidade, $m = 2$, 4, 8 e 16 são apresentadas nas Figs. 58, 59 e 60. Os algoritmos MOACS e MOAS obtiveram valores de hipervolume superiores ao do MOCRPSO para as curvas *best* sem restrição de cardinalidade e $m = 2$. Para as curvas *median* e *worst*, os valores obtidos pelo MOCRPSO foram competitivos ou superiores aos do MOACS e MOAS. Na maioria das curvas EAF comparativas para $m = 4$, 8 e 16, o MOCRPSO obteve melhores resultados quando comparado ao MOAS. Entretando, apenas na curva *best* para $m = 16$ o MOCRPSO obteve valor de hipervolume superior ao do MOACS.

As Figs. 61 e 62 apresentam as curvas EAF comparativas da treliça de 72 barras sem restrição de cardinalidade, $m = 2$, 4 e 8. Em 6 dos 12 gráficos o MOCRPSO obteve valores de hipervolume maior que os dos algoritmos MOACS e MOAS. Em 8 dos 12 gráficos o MOCRPSO obteve valores de hipervolume maior que os do algoritmo MOAS. E em 2 dos 12 gráficos, o três algoritmos obtiveram valores iguais, com diferenças apenas a partir da terceira casa decimal.

A última estrutura analisada refere-se à treliça de 942 barras, que apresenta as curvas EAF comparativas nas Figs. 63, 64 e 65. Os resultados sem restrição de cardinalidade e $m = 2$ mostraram que o MOCRPSO foi superior, apresentando valores de hipervolume maior em 4 dos 6 gráficos. Para $m = 4$, 8, 16 e 32 há um equilíbrio entre os valores de hipervolume obtidos pelos três algoritmos.

As Figs. 51, 52 e 53 apresentam os perfis de desempenho dos três algoritmos analisados no segundo grupo de comparações para curvas EAF *best*, *median* e *worst*, respectivamente. Nas duas primeiras figuras, o MOAS obteve um desempenho superior que os demais algoritmos, alcançando o menor valor de τ, tal que $\rho(\tau) = 1$. Na Fig. 53, o MOACS obteve o menor valor de τ, tal que $\rho(\tau) = 1$. As Tabelas 11, 12 e 13 apresentam o valor da área normalizada sob as curvas dos perfis de desempenho. Os três algoritmos obtiveram valores da área muito próximos. O MOCRPSO obteve o segundo maior valor da área em 2 das 3 tabelas apresentadas.

Após as análises apresentadas, O MOCRPSO mostrou ser um algoritmo robusto e competitivo em relação aos algoritmos MOACS e MOAS.

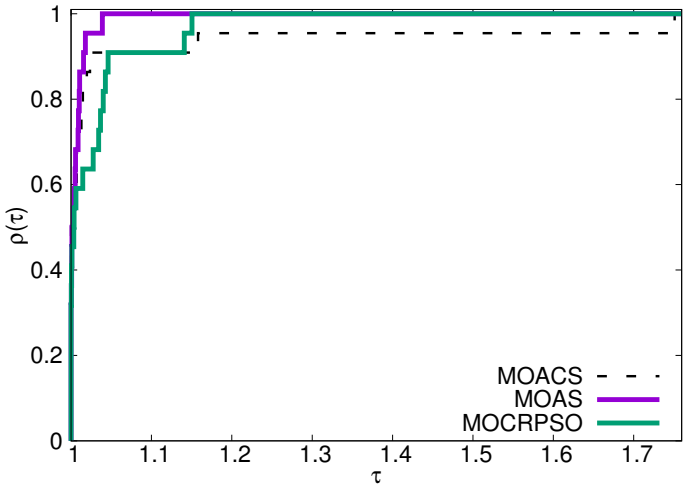


Figura 51 – Perfis de desempenho dos algoritmos MOACS, MOAS e MOCRPSO para a curva EAF *best*.

Tabela 15 – Área normalizada sob as curvas dos perfis de desempenho do algoritmos MOACS, MOAS e MOCRPSO para a curva EAF *best*.

Algoritmo	MOACS	MOAS	MOCRPSO
Área	0.9466	1.0	0.9745

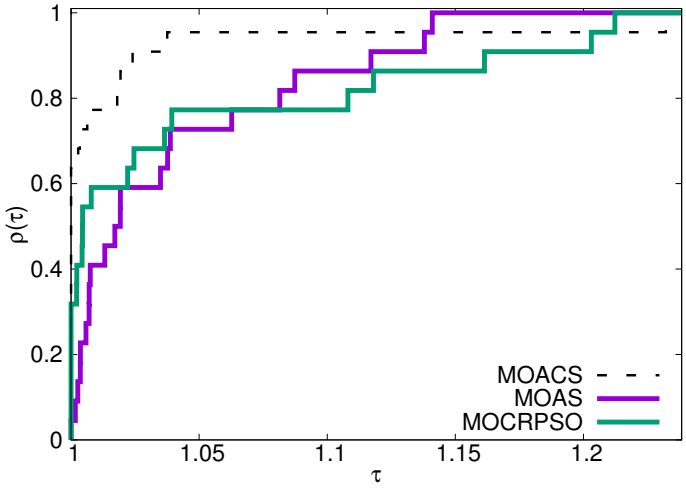


Figura 52 – Perfis de desempenho dos algoritmos MOACS, MOAS e MOCRPSO para a curva EAF *median*.

Tabela 16 – Área normalizada sob as curvas dos perfis de desempenho do algoritmos MOACS, MOAS e MOCRPSO para a curva EAF *median*.

Algoritmo	MOACS	MOAS	MOCRPSO
Área	1.0	0.8944	0.8725

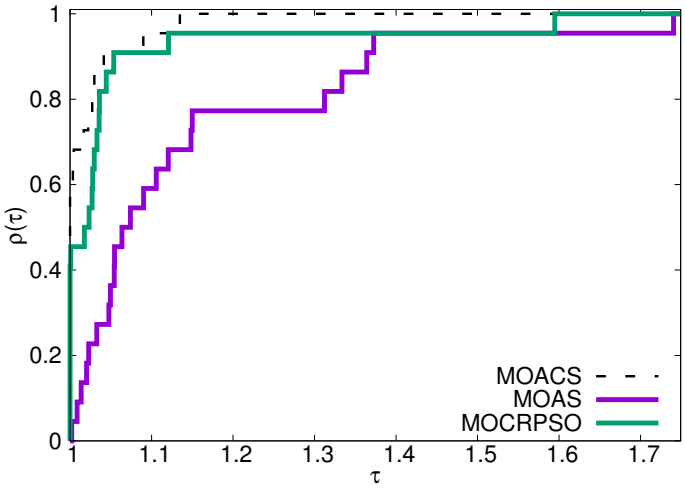


Figura 53 – Perfis de desempenho dos algoritmos MOACS, MOAS e MOCRPSO para a curva EAF *worst*.

Tabela 17 – Área normalizada sob as curvas dos perfis de desempenho do algoritmos MOACS, MOAS e MOCRPSO para a curva EAF *worst*.

Algoritmo	MOACS	MOAS	MOCRPSO
Área	1.0	0.8233	0.9576

As Tabelas 18 e 19 apresentam a média e o desvio padrão dos valores de hipervolumes apresentados nas Figs. 54 a 65 para os algoritmos MOACS, MOAS e MOCRPSO. O sinal (+) indica que a diferença entre as medidas calculadas para cada uma das execuções realizadas é estatisticamente significativa (p-valor ≤ 0.05) quando comparado com os mesmos resultados obtidos pelo algoritmo com a melhor média. Para cada problema, o melhor resultado está destacado em negrito. Na Tabela 18, as melhores médias foram obtidas pelo algoritmo MOACS. Na Tabela 19, o MOCRPSO apresentou melhor desempenho que o MOACS e desempenho igual ao do algoritmo MOAS. Em sintése, o algoritmo MOCRPSO apresentou um desempenho satisfatório nessa comparação, destacando-se principalmente em estruturas com maior número de barras.

Tabela 18 – Média e desvio padrão dos resultados dos hipervolumes das treliças de 10, 25 e 60 barras para a comparação 2.

	MOACS	MOAS	MOCRPSO
Treliça de 10 barras sem r.c.			
Média	0.807620(+)	0.707400(+)	**0.826590**
Desvio padrão	0.007955	0.031431	0.002777
Treliça de 10 barras $m = 2$			
Média	**0.813110**	0.800090	0.805610(+)
Desvio padrão	0.003985	0.006825	0.014638
Treliça de 10 barras $m = 4$			
Média	**0.834260**	0.797660	0.828830(+)
Desvio padrão	0.003578	0.020209	0.005172
Treliça de 10 barras $m = 8$			
Média	**0.824650**	0.760840	0.813430(+)
Desvio padrão	0.004438	0.027111	0.009286
Treliça de 25 barras sem r.c.			
Média	**0.884110**	0.878630	0.883750(+)
Desvio padrão	0.001177	0.003401	0.000797
Treliça de 25 barras $m = 2$			
Média	**0.863380**	0.859050	0.866220(+)
Desvio padrão	0.002430	0.002492	0.003456
Treliça de 25 barras $m = 4$			
Média	0.879020(+)	0.876500(+)	**0.883930**
Desvio padrão	0.005239	0.003621	0.001909
Treliça de 60 barras sem r.c.			
Média	**0.671070**	0.625850	0.613600(+)
Desvio padrão	0.008966	0.021645	0.020351
Treliça de 60 barras $m = 2$			
Média	**0.766750**	0.738970	0.753320(+)
Desvio padrão	0.003073	0.014231	0.012626
Treliça de 60 barras $m = 4$			
Média	0.770480(+)	0.727950	**0.771290**
Desvio padrão	0.009835	0.017054	0.017906
Treliça de 60 barras $m = 8$			
Média	**0.748720**	0.689720	0.701320(+)
Desvio padrão	0.009541	0.024649	0.030800
Treliça de 60 barras $m = 16$			
Média	**0.727360**	0.644720	0.641560
Desvio padrão	0.011328	0.026839	0.033175

Tabela 19 – Média e desvio padrão dos resultados dos hipervolumes das treliças de 72 e 942 barras para a comparação 2 (continuação).

	MOACS	MOAS	MOCRPSO
Treliça de 72 barras sem r.c.			
Média	**0.897340**	0.887260	0.888180(+)
Desvio padrão	0.003145	0.003630	0.006113
Treliça de 72 barras $m = 2$			
Média	**0.892430**	0.889710	0.886720(+)
Desvio padrão	0.003269	0.002288	0.005370
Treliça de 72 barras $m = 4$			
Média	0.899810(+)	0.898630(+)	**0.911690**
Desvio padrão	0.005124	0.004519	0.006489
Treliça de 72 barras $m = 8$			
Média	0.887050(+)	0.888720(+)	**0.894230**
Desvio padrão	0.019502	0.004835	0.013506
Treliça de 942 barras sem r.c.			
Média	0.773570	**0.872320**	0.682280(+)
Desvio padrão	0.009370	0.005874	0.025300
Treliça de 942 barras $m = 2$			
Média	0.883650(+)	0.879770(+)	**0.905070**
Desvio padrão	0.004625	0.012341	0.01229
Treliça de 942 barras $m = 4$			
Média	0.877130(+)	0.882850(+)	**0.896960**
Desvio padrão	0.004462	0.010755	0.017562
Treliça de 942 barras $m = 8$			
Média	0.869810	**0.872360**	0.821980(+)
Desvio padrão	0.005467	0.011335	0.025485
Treliça de 942 barras $m = 16$			
Média	0.868550	**0.869850**	0.756510(+)
Desvio padrão	0.007138	0.008966	0.031910
Treliça de 942 barras $m = 32$			
Média	0.859130	**0.859950**	0.704640(+)
Desvio padrão	0.009056	0.009381	0.038502

6.3 TERCEIRO CONJUNTO DE EXPERIMENTOS

No terceiro conjunto de experimentos, a frequência natural de vibração é introduzida no problema multiobjetivo como o seu segundo objetivo. Três diferentes modelagens envolvendo as frequências foram definidas para as treliças de 10, 25, 72 e 200 barras e serão apresentadas como casos I, II e III. Em todos os casos as restrições de cardinalidade são adicionadas à formulação do problema. A descrição de cada caso é apresentada a seguir.

6.3.1 **Caso I**

O primeiro caso consiste em encontrar um conjunto de variáveis de projeto que correspondem às áreas da seção transversal das barras $(A_1, \ldots, A_{n_a})$ da estrutura que minimiza seu peso e que maximiza a menor frequência natural de vibração. O problema é formulado como

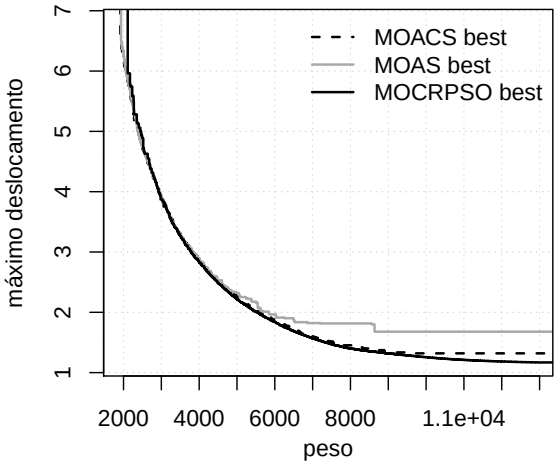
$$\begin{gathered} \min \quad W = \sum_{i=1}^{n_a} \rho A_i L_i \\ \text{e} \\ \max \quad \text{mínimo } (\omega_j),\ j = 1, \ldots, n_f, \end{gathered} \tag{6.9}$$

sujeito a

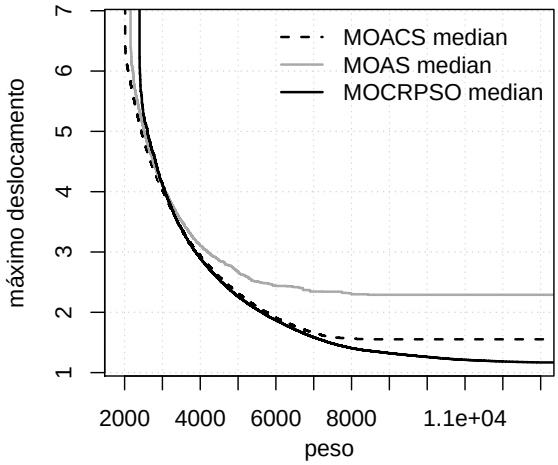
$$\begin{gathered} \frac{\sigma_{il}}{\bar{\sigma}} - 1 \leq 0, \quad i = 1, \ldots, n_a,\ l = 1, \ldots, n_l \\ \text{e} \\ \frac{u_{ql}}{\bar{u}} - 1 \leq 0, \quad q = 1, \ldots, n_d,\ l = 1, \ldots, n_l, \end{gathered} \tag{6.10}$$

onde ρ é a massa específica, L é o comprimento da barra e σ e u são as restrições de tensões e deslocamentos, respectivamente. As variáveis n_a, n_f, n_d e n_l representam, respectivamente, o número de variáveis de projeto, de frequências naturais de vibração, de restrições de deslocamento e de casos de carregamento.

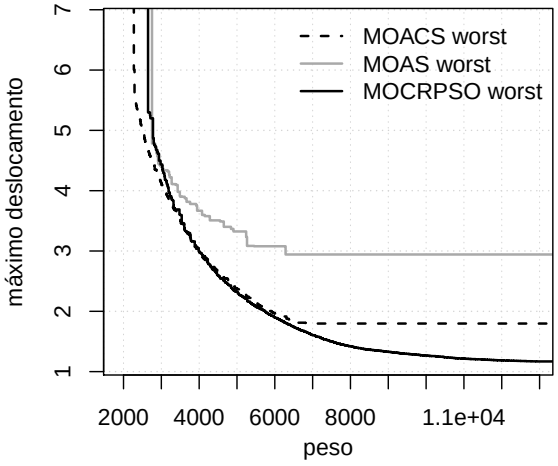
As curvas EAF *best*, *median* e *worst* do algoritmo MOCRPSO para as estruturas analisadas com restrições de cardinalidade utilizando variáveis contínuas e discretas são apresentadas nas Figs. 66 a 72 para o caso I. Uma análise envolvendo os resultados para todos os 3 casos estudados aqui será feita ao final desse conjunto de experimentos.



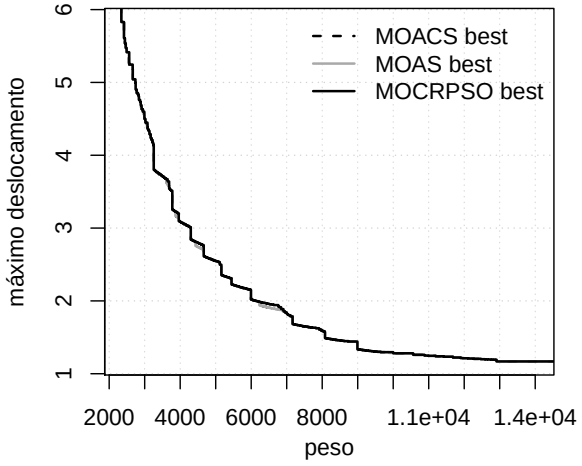
(a) $H_1 = 0.5231$, $H_2 = 0.5040$ e $H_3 = 0.5236$



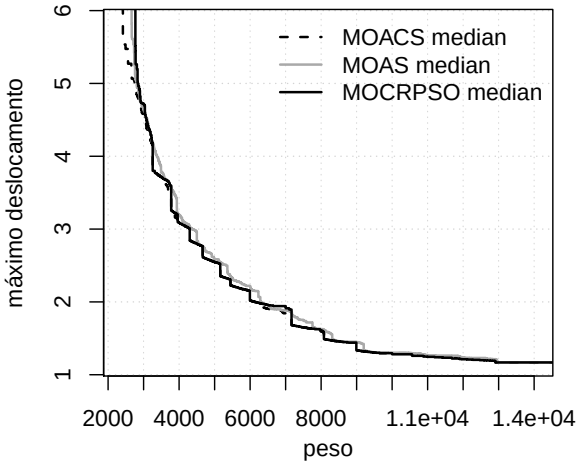
(b) $H_1 = 0.5355$, $H_2 = 0.4706$ e $H_3 = 0.5332$



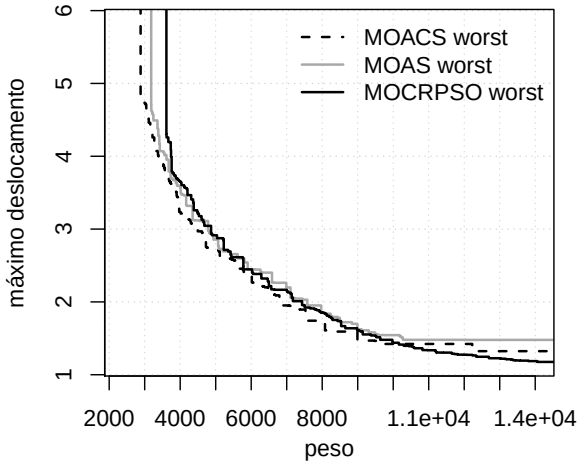
(c) $H_1 = 0.3514$, $H_2 = 0.2569$ e $H_3 = 0.3526$



(d) $H_1 = 0.8012$, $H_2 = 0.8012$ e $H_3 = 0.7704$

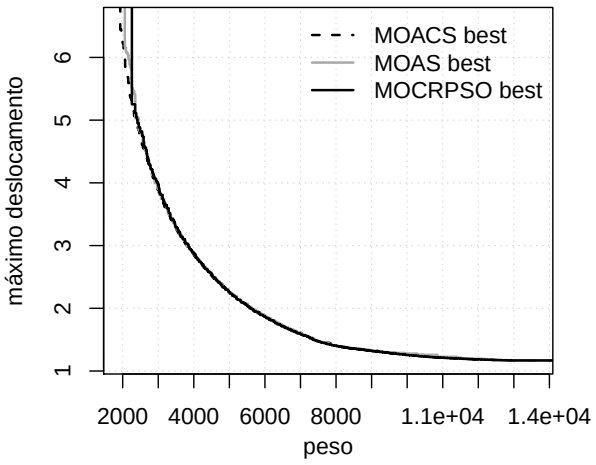


(e) $H_1 = 0.8059$, $H_2 = 0.7907$ e $H_3 = 0.7208$

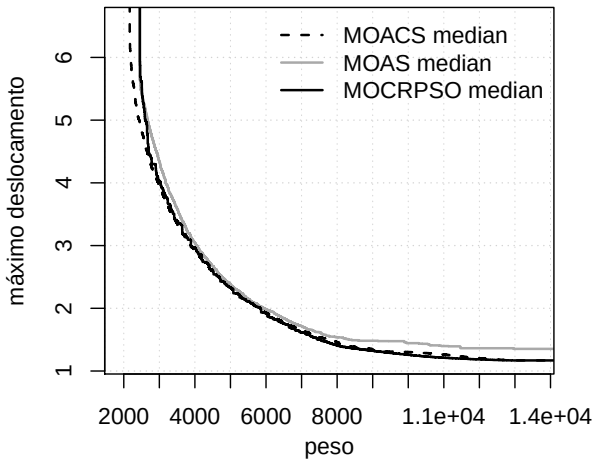


(f) $H_1 = 0.6909$, $H_2 = 0.6557$ e $H_3 = 0.7530$

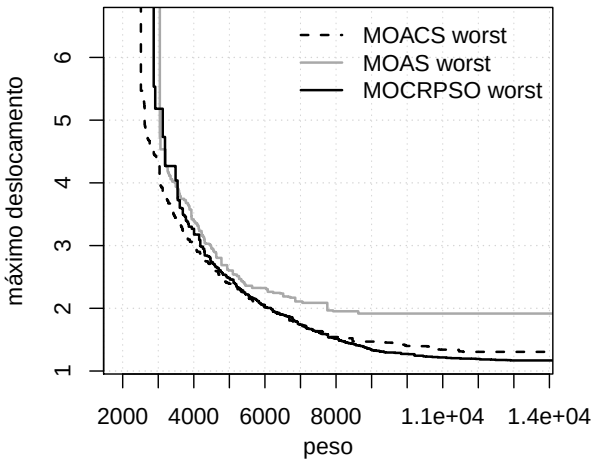
Figura 54 – Curvas EAF da treliça de 10 barras caso discreto, sem restrição de cardinalidade (figuras a, b, c) e $m = 2$ (figuras d, e, f). Valores de hipervolume normalizados: H_1 para MOACS, H_2 para MOAS e H_3 para MOCRPSO.



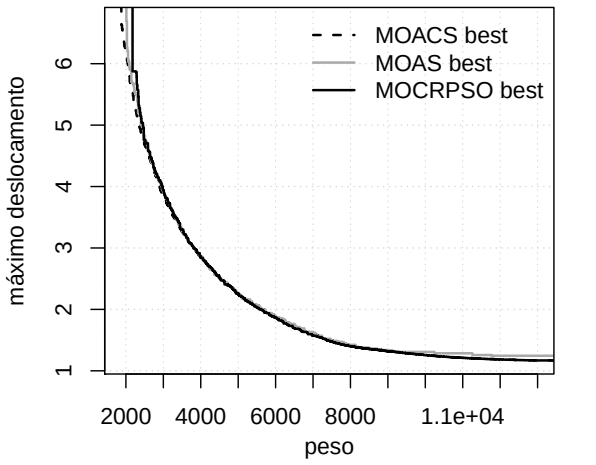
(a) $H_1 = 0.8320$, $H_2 = 0.8278$ e $H_3 = 0.8292$



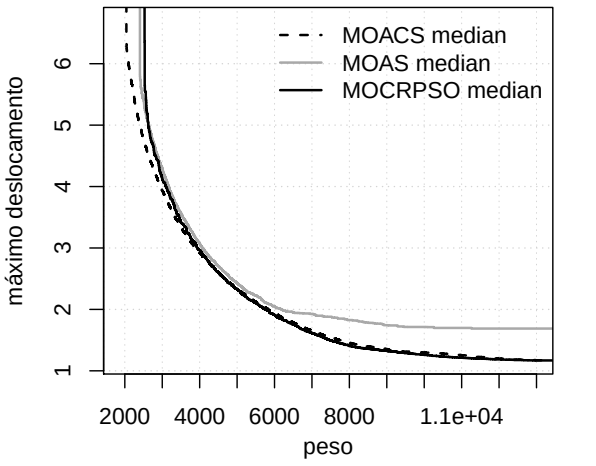
(b) $H_1 = 0.8389$, $H_2 = 0.8076$ e $H_3 = 0.6920$



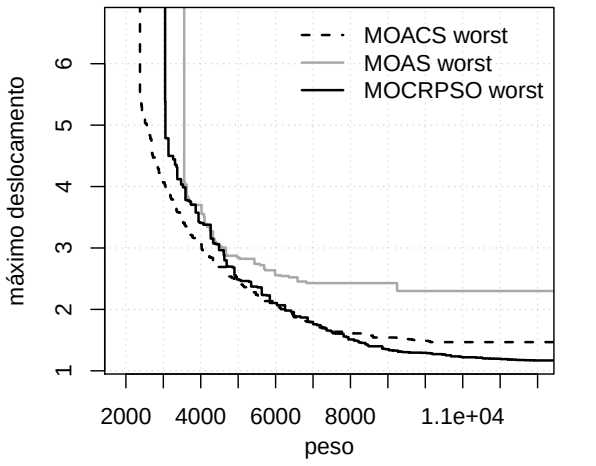
(c) $H_1 = 0.6664$, $H_2 = 0.5795$ e $H_3 = 0.6472$



(d) $H_1 = 0.8366$, $H_2 = 0.8290$ e $H_3 = 0.8335$

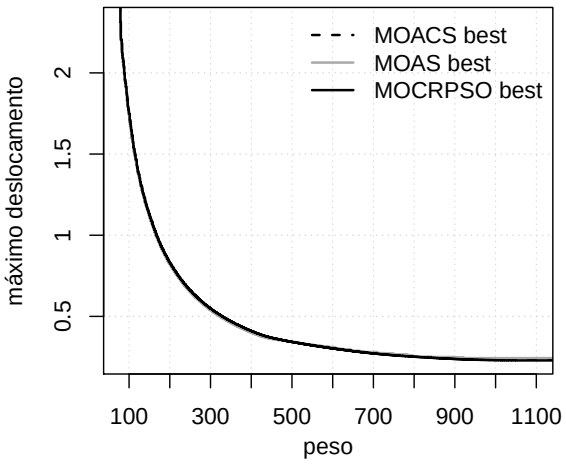


(e) $H_1 = 0.8388$, $H_2 = 0.7756$ e $H_3 = 0.7570$

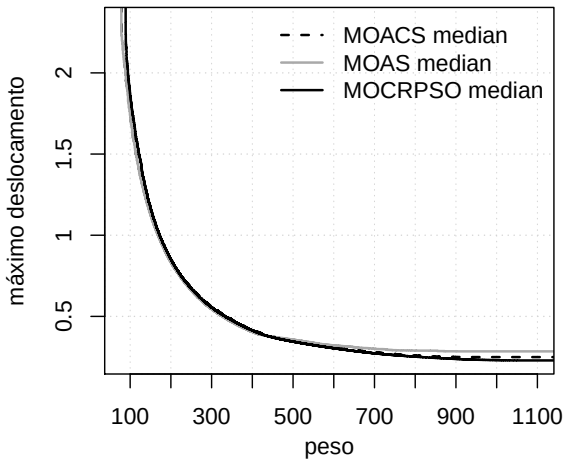


(f) $H_1 = 0.5308$, $H_2 = 0.4045$ e $H_3 = 0.5138$

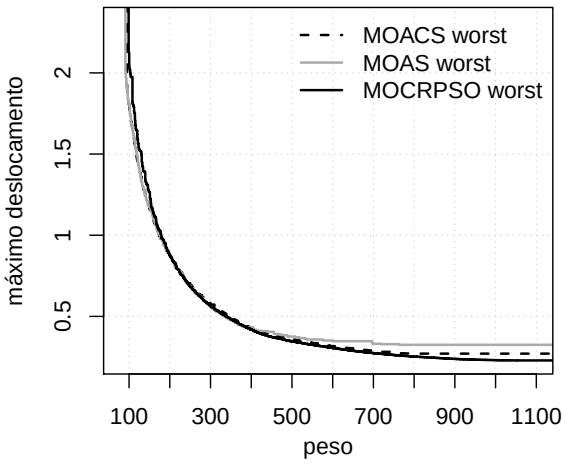
Figura 55 – Curvas EAF da treliça de 10 barras caso discreto, $m = 4$ (figuras a, b, c) e $m = 8$ (figuras d, e, f). Valores de hipervolume normalizados: H_1 para MOACS, H_2 para MOAS e H_3 para MOCRPSO.



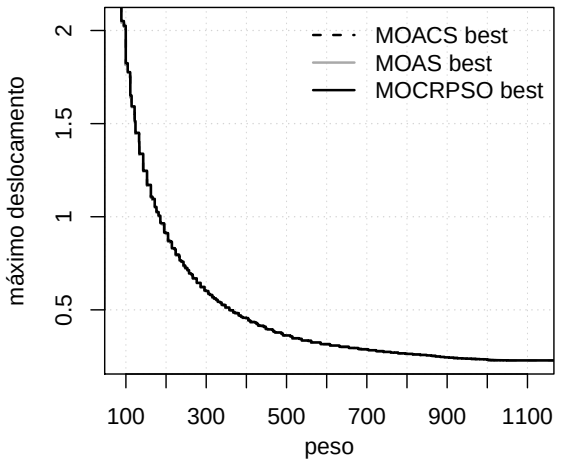
(a) $H_1 = 0.7847$, $H_2 = 0.7841$ e $H_3 = 0.7838$



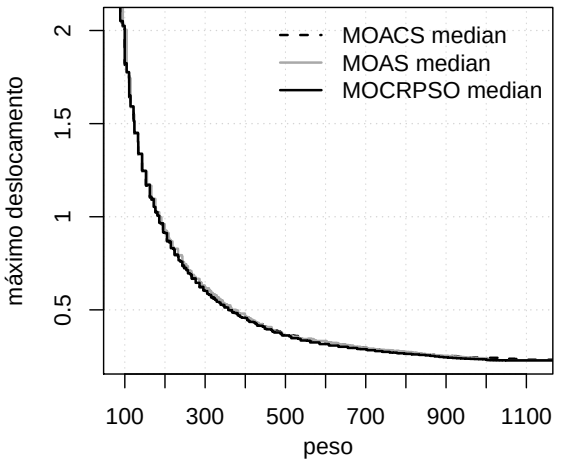
(b) $H_1 = 0.7930$, $H_2 = 0.7896$ e $H_3 = 0.7952$



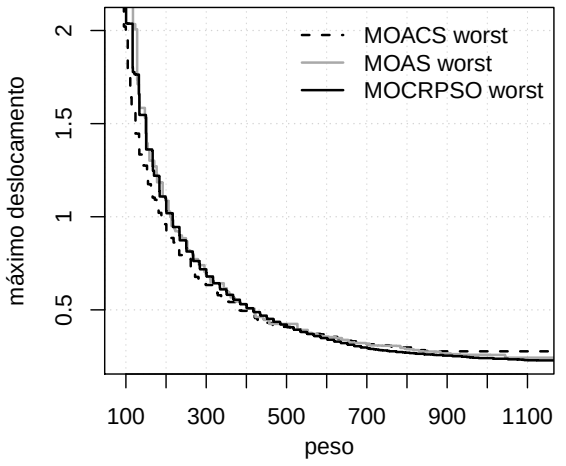
(c) $H_1 = 0.5977$, $H_2 = 0.5923$ e $H_3 = 0.6004$



(d) $H_1 = 0.8579$, $H_2 = 0.8579$ e $H_3 = 0.8579$

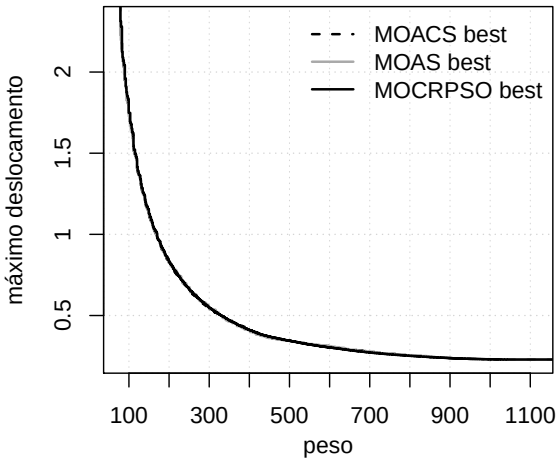


(e) $H_1 = 0.8603$, $H_2 = 0.8572$ e $H_3 = 0.8417$

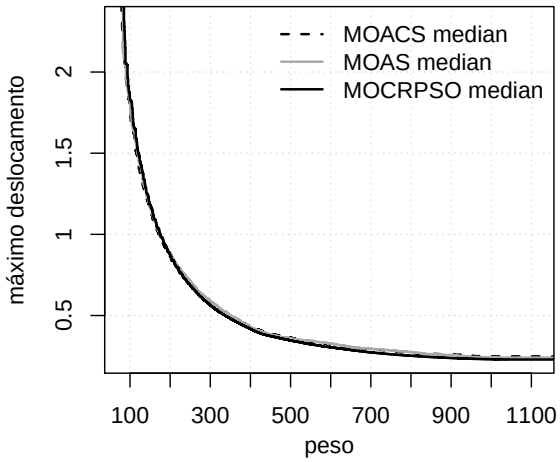


(f) $H_1 = 0.8400$, $H_2 = 0.8205$ e $H_3 = 0.8650$

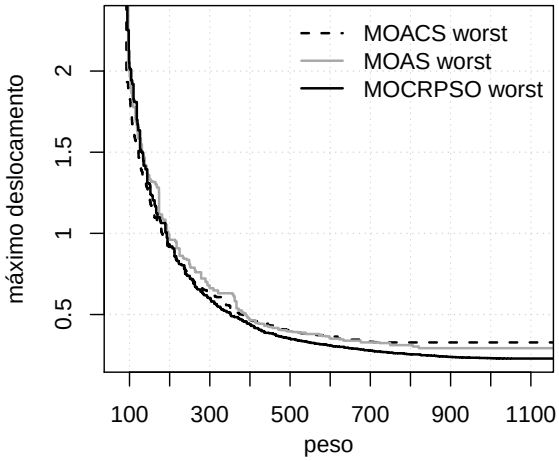
Figura 56 – Curvas EAF da treliça de 25 barras caso discreto, sem restrição de cardinalidade (figuras a, b, c) e $m = 2$ (figuras d, e, f). Valores de hipervolume normalizados: H_1 para MOACS, H_2 para MOAS e H_3 para MOCRPSO.



(a) $H_1 = 0.8830$, $H_2 = 0.8823$ e $H_3 = 0.8827$

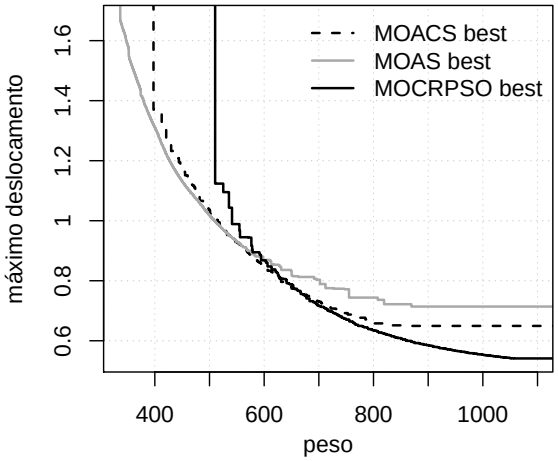


(b) $H_1 = 0.8755$, $H_2 = 0.8732$ e $H_3 = 0.8736$

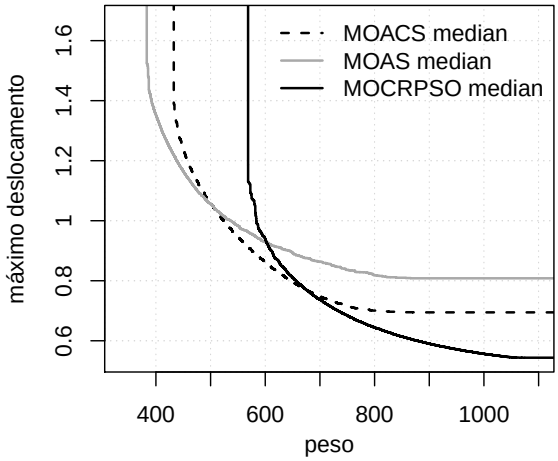


(c) $H_1 = 0.6261$, $H_2 = 0.6137$ e $H_3 = 0.6256$

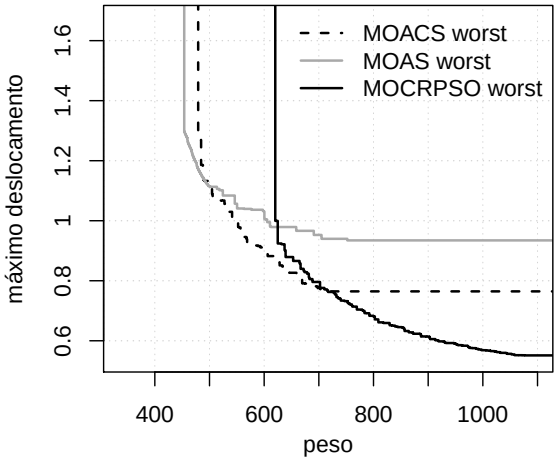
Figura 57 – Curvas EAF da treliça de 25 barras caso discreto, $m = 4$ (figuras a, b, c). Valores de hipervolume normalizados: H_1 para MOACS, H_2 para MOAS e H_3 para MOCRPSO.



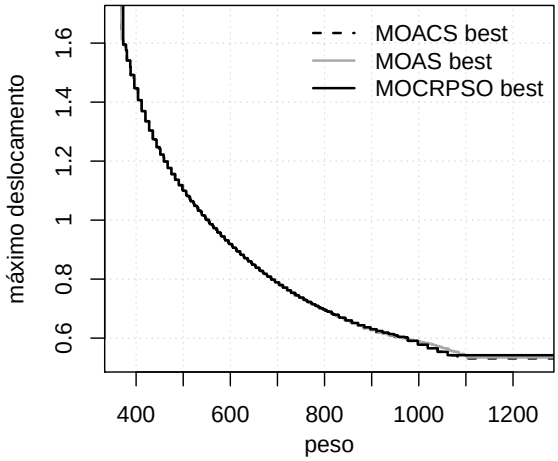
(a) $H_1 = 0.4665$, $H_2 = 0.4660$ e $H_3 = 0.4055$



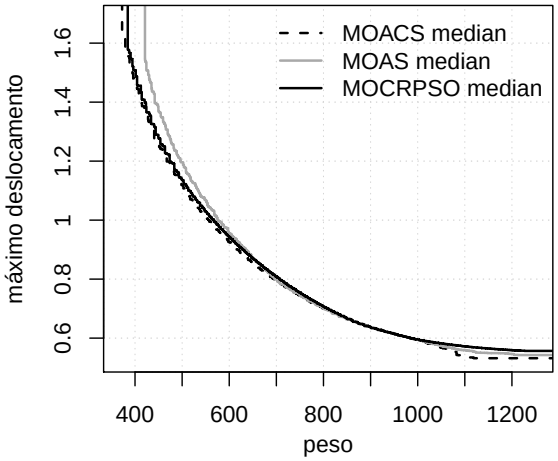
(b) $H_1 = 0.4580$, $H_2 = 0.4254$ e $H_3 = 0.4752$



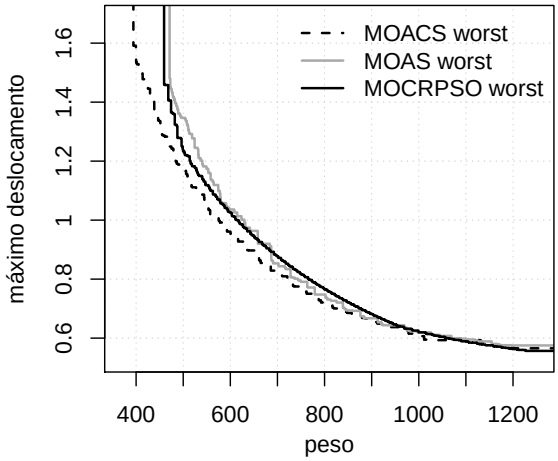
(c) $H_1 = 0.3502$, $H_2 = 0.2626$ e $H_3 = 0.3422$



(d) $H_1 = 0.7286$, $H_2 = 0.7266$ e $H_3 = 0.6965$

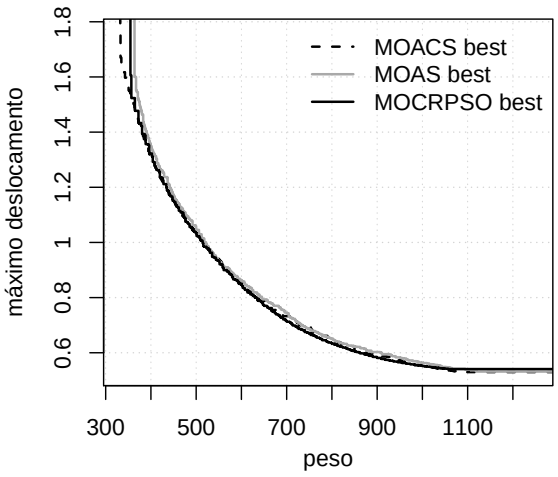


(e) $H_1 = 0.7349$, $H_2 = 0.7101$ e $H_3 = 0.7173$

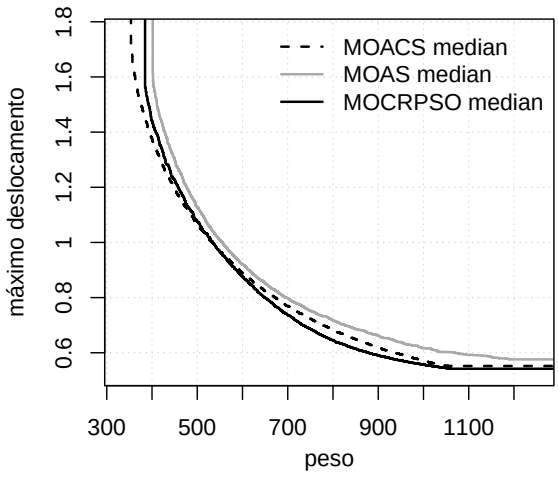


(f) $H_1 = 0.7377$, $H_2 = 0.6868$ e $H_3 = 0.6582$

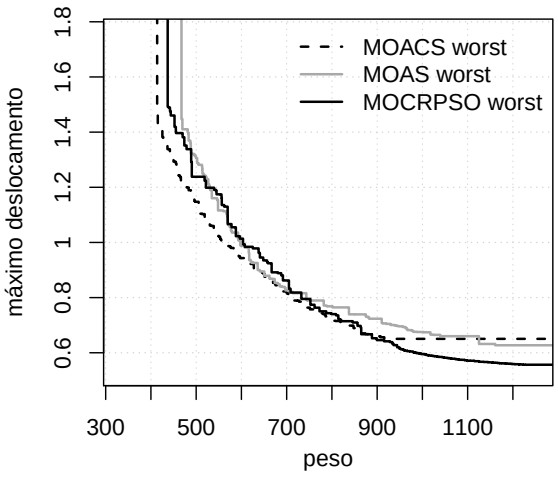
Figura 58 – Curvas EAF da treliça de 60 barras caso discreto, sem restrição de cardinalidade (figuras a, b, c) e $m = 2$ (figuras d, e, f). Valores de hipervolume normalizados: H_1 para MOACS, H_2 para MOAS e H_3 para MOCRPSO.



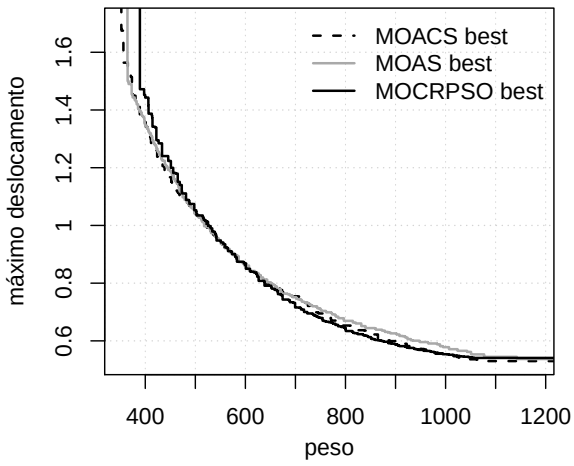
(a) $H_1 = 0.7689$, $H_2 = 0.7554$ e $H_3 = 0.7579$



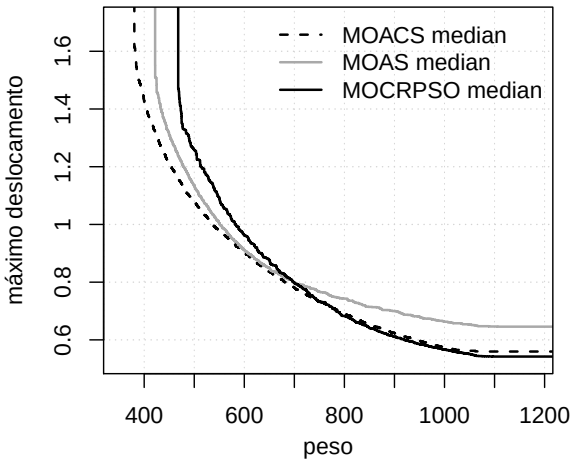
(b) $H_1 = 0.7728$, $H_2 = 0.7272$ e $H_3 = 0.7456$



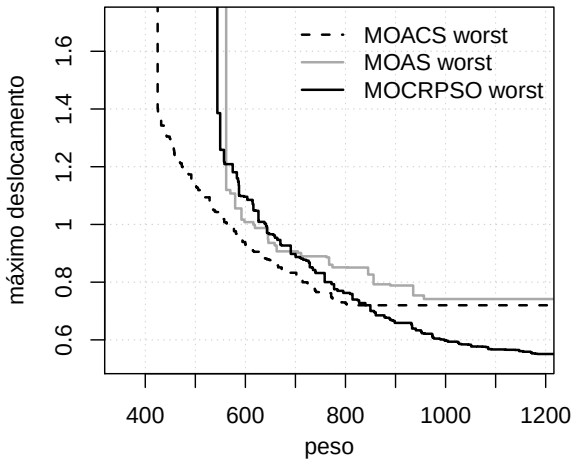
(c) $H_1 = 0.6548$, $H_2 = 0.6006$ e $H_3 = 0.6215$



(d) $H_1 = 0.7688$, $H_2 = 0.7570$ e $H_3 = 0.7482$

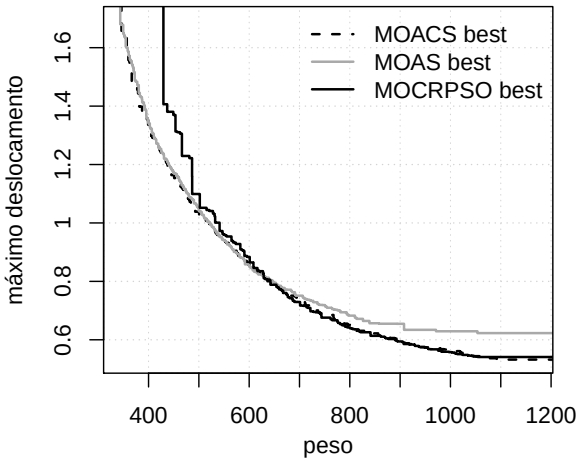


(e) $H_1 = 0.7446$, $H_2 = 0.6848$ e $H_3 = 0.6189$

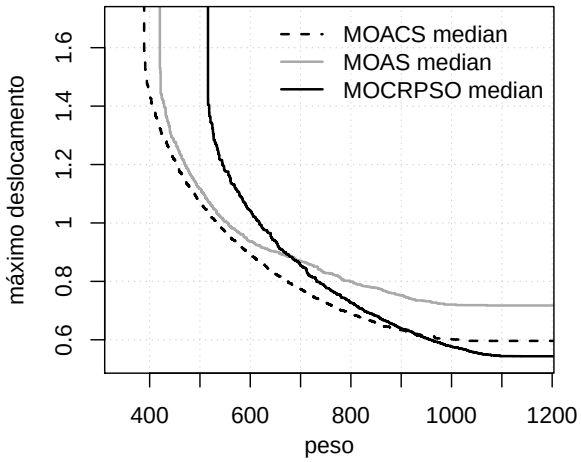


(f) $H_1 = 0.4199$, $H_2 = 0.3162$ e $H_3 = 0.4131$

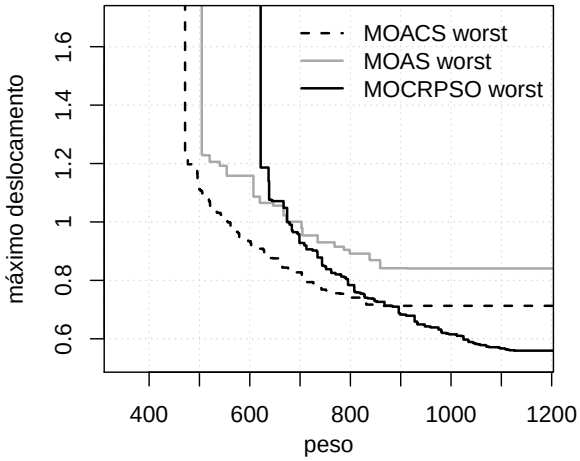
Figura 59 – Curvas EAF da treliça de 60 barras caso discreto, $m = 4$ (figuras a, b, c) e $m = 8$ (figuras d, e, f). Valores de hipervolume normalizados: H_1 para MOACS, H_2 para MOAS e H_3 para MOCRPSO.



(a) $H_1 = 0.4131$, $H_2 = 0.7233$ e $H_3 = 0.6340$

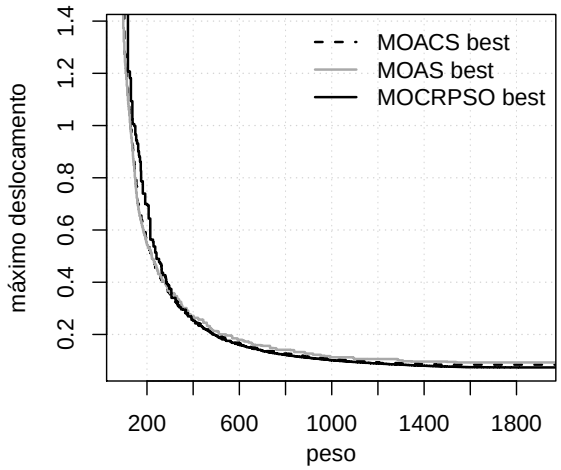


(b) $H_1 = 0.6643$, $H_2 = 0.5822$ e $H_3 = 0.5720$

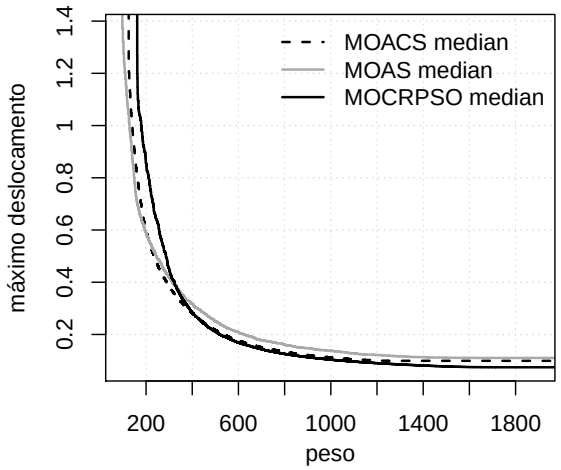


(c) $H_1 = 0.3723$, $H_2 = 0.2139$ e $H_3 = 0.3625$

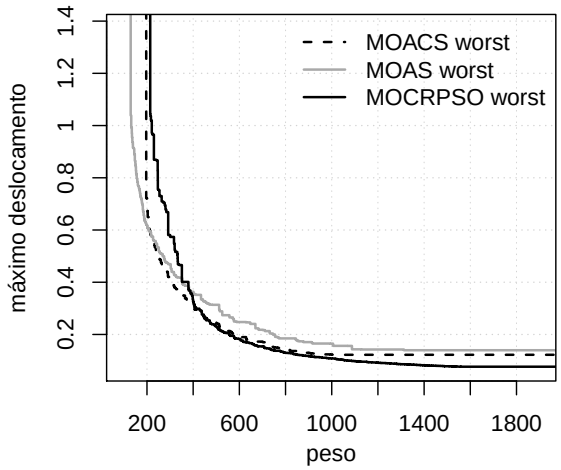
Figura 60 – Curvas EAF da treliça de 60 barras caso discreto, $m = 16$ (figuras a, b, c). Valores de hipervolume normalizados: H_1 para MOACS, H_2 para MOAS e H_3 para MOCRPSO.



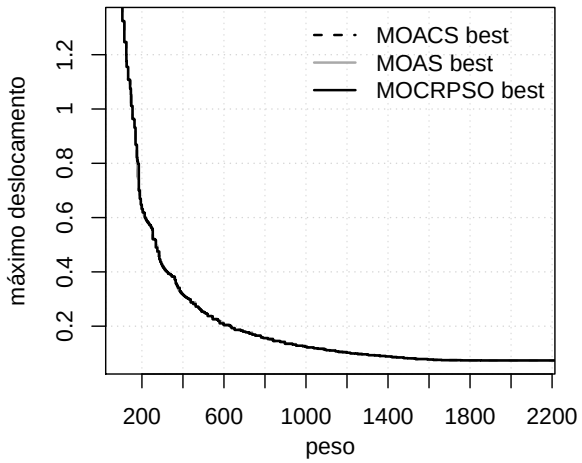
(a) $H_1 = 0.7514$, $H_2 = 0.7454$ e $H_3 = 0.7534$



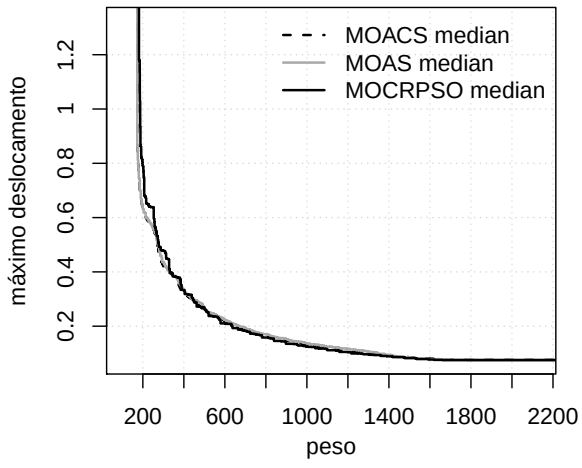
(b) $H_1 = 0.8092$, $H_2 = 0.7989$ e $H_3 = 0.8143$



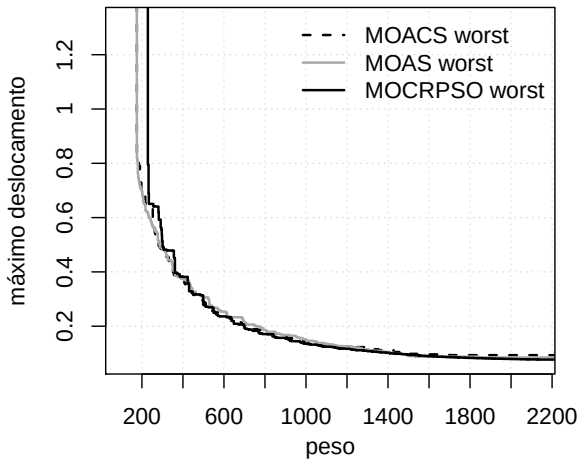
(c) $H_1 = 0.6722$, $H_2 = 0.6524$ e $H_3 = 0.6835$



(d) $H_1 = 0.8858$, $H_2 = 0.8855$ e $H_3 = 0.8854$

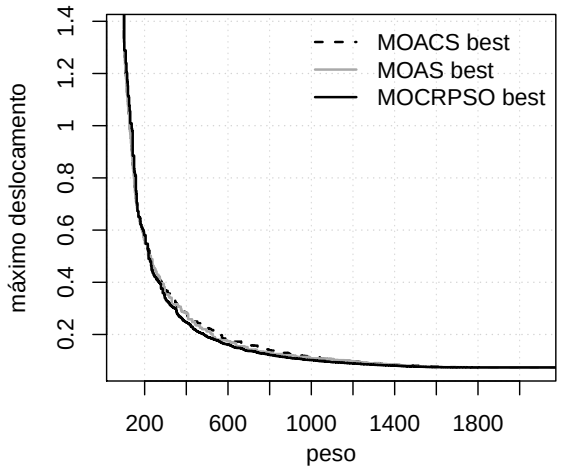


(e) $H_1 = 0.8482$, $H_2 = 0.8451$ e $H_3 = 0.8161$

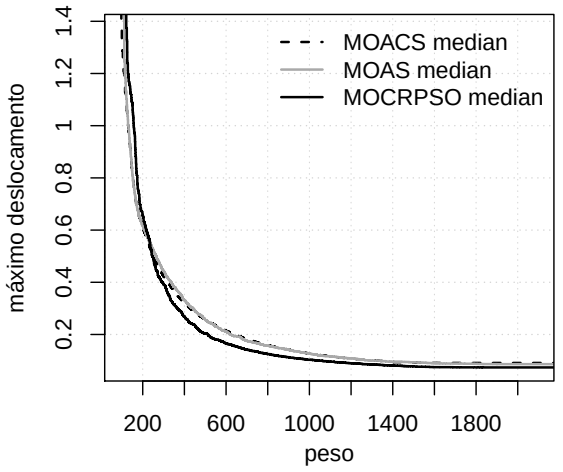


(f) $H_1 = 0.8693$, $H_2 = 0.8688$ e $H_3 = 0.8887$

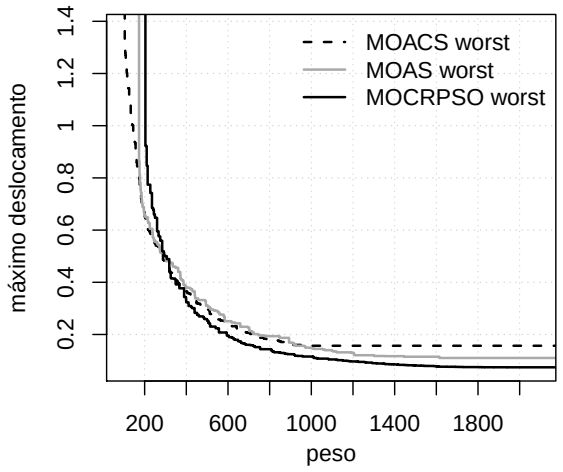
Figura 61 – Curvas EAF da treliça de 72 barras caso discreto, sem restrição de cardinalidade (figuras a, b, c) e $m = 2$ (figuras d, e, f). Valores de hipervolume normalizados: H_1 para MOACS, H_2 para MOAS e H_3 para MOCRPSO.



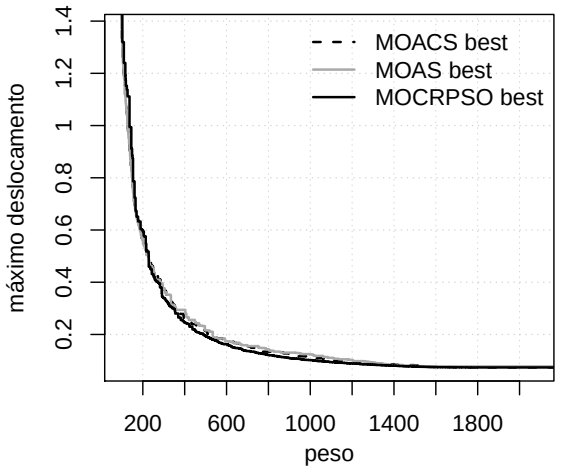
(a) $H_1 = 0.9093$, $H_2 = 0.9123$ e $H_3 = 0.9174$



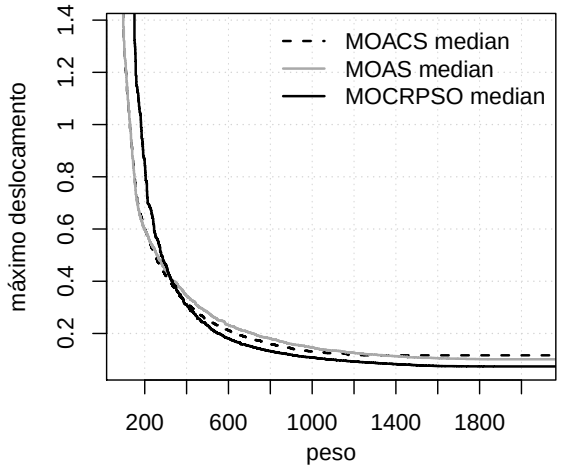
(b) $H_1 = 0.8953$, $H_2 = 0.8938$ e $H_3 = 0.8933$



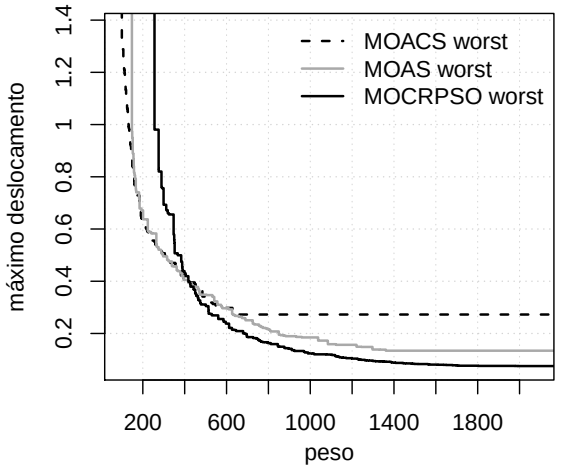
(c) $H_1 = 0.6885$, $H_2 = 0.6826$ e $H_3 = 0.6766$



(d) $H_1 = 0.9074$, $H_2 = 0.9048$ e $H_3 = 0.9140$

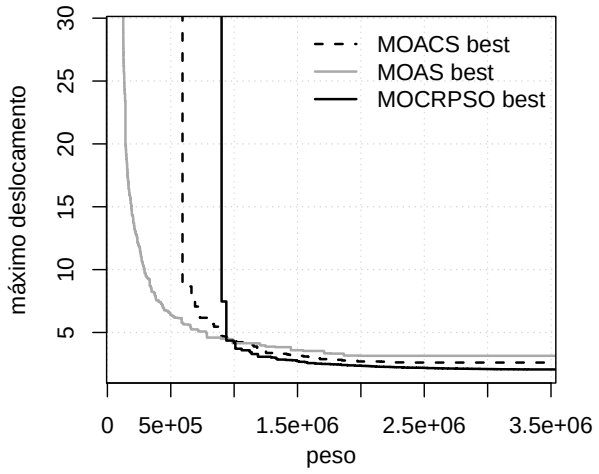


(e) $H_1 = 0.8590$, $H_2 = 0.8530$ e $H_3 = 0.8552$

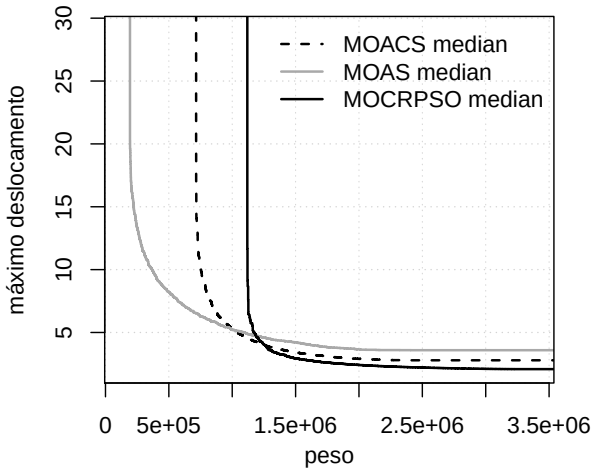


(f) $H_1 = 0.5078$, $H_2 = 0.5277$ e $H_3 = 0.5288$

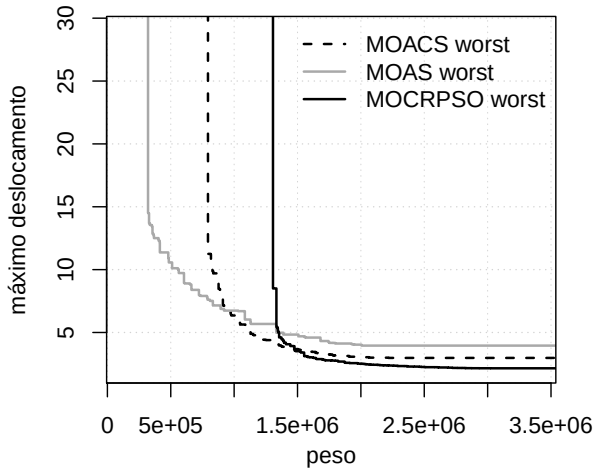
Figura 62 – Curvas EAF da treliça de 72 barras caso discreto, $m = 4$ (figuras a, b, c) e $m = 8$ (figuras d, e, f). Valores de hipervolume normalizados: H_1 para MOACS, H_2 para MOAS e H_3 para MOCRPSO.



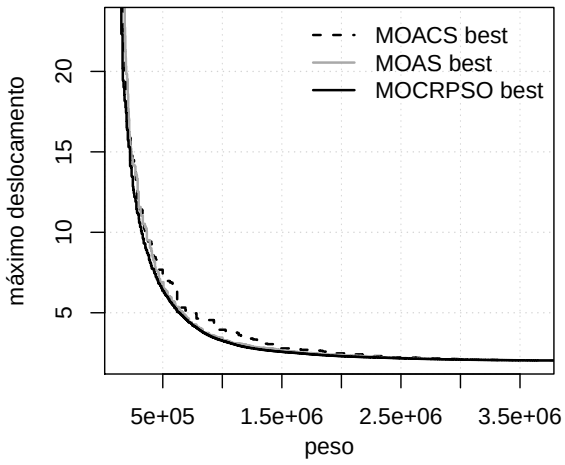
(a) $H_1 = 0.6372$, $H_2 = 0.7379$ e $H_3 = 0.7018$



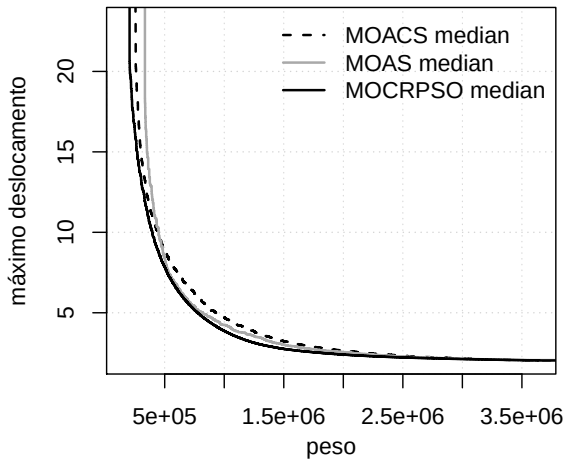
(b) $H_1 = 0.4940$, $H_2 = 0.5866$ e $H_3 = 0.6087$



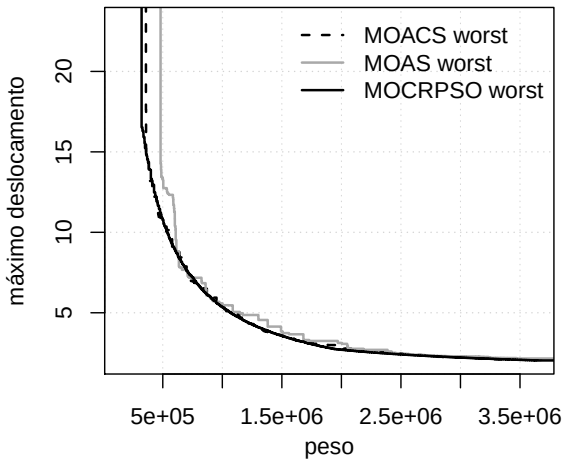
(c) $H_1 = 0.4289$, $H_2 = 0.4615$ e $H_3 = 0.4867$



(d) $H_1 = 0.9118$, $H_2 = 0.9217$ e $H_3 = 0.9297$

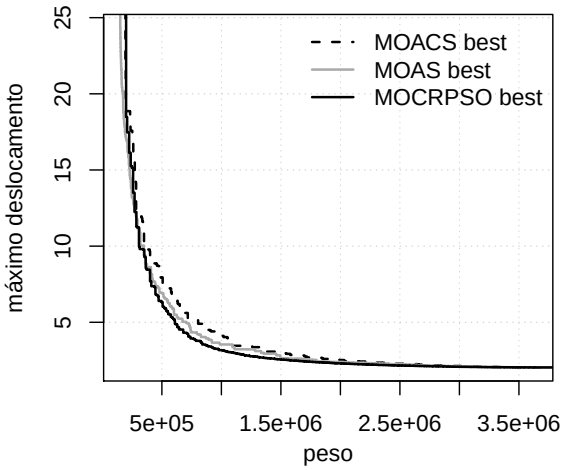


(e) $H_1 = 0.8972$, $H_2 = 0.9007$ e $H_3 = 0.9145$

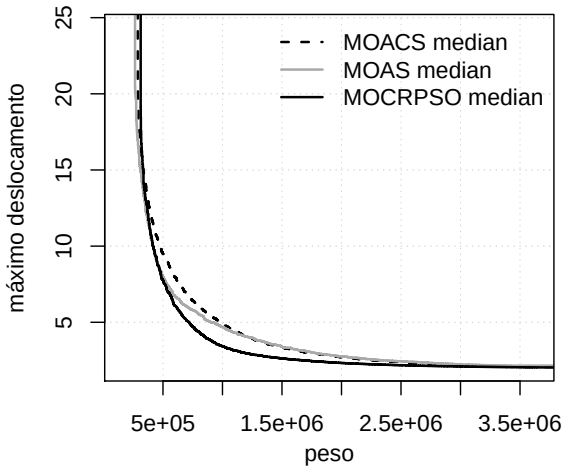


(f) $H_1 = 0.8737$, $H_2 = 0.8460$ e $H_3 = 0.8436$

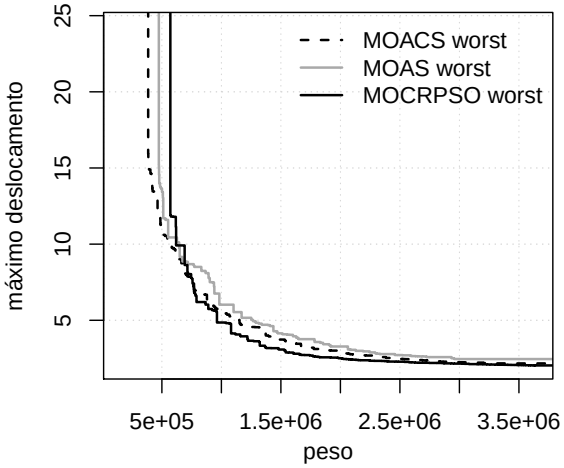
Figura 63 – Curvas EAF da treliça de 942 barras caso discreto, sem restrição de cardinalidade (figuras a, b, c) e $m = 2$ (figuras d, e, f). Valores de hipervolume normalizados: H_1 para MOACS, H_2 para MOAS e H_3 para MOCRPSO.



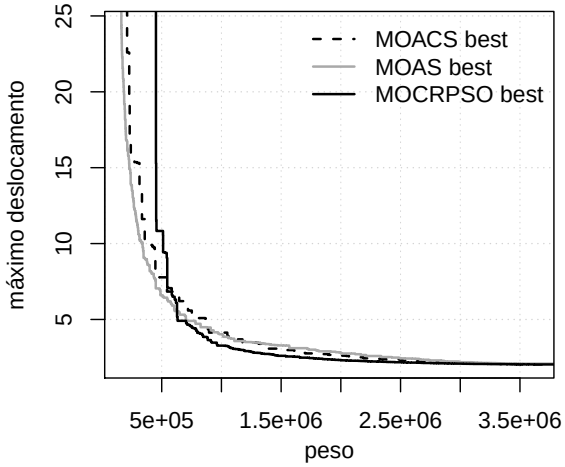
(a) $H_1 = 0.9083$, $H_2 = 0.9259$ e $H_3 = 0.9297$



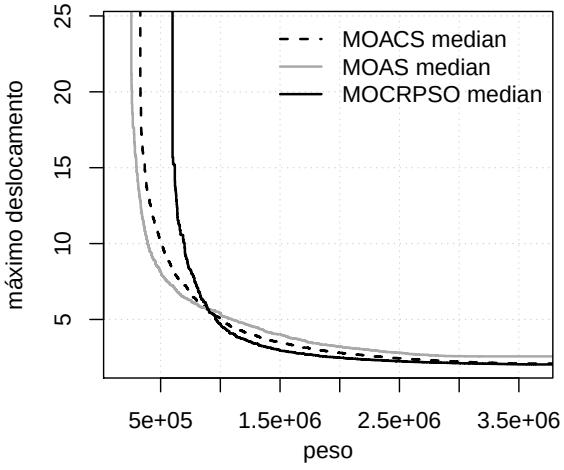
(b) $H_1 = 0.8863$, $H_2 = 0.8958$ e $H_3 = 0.9076$



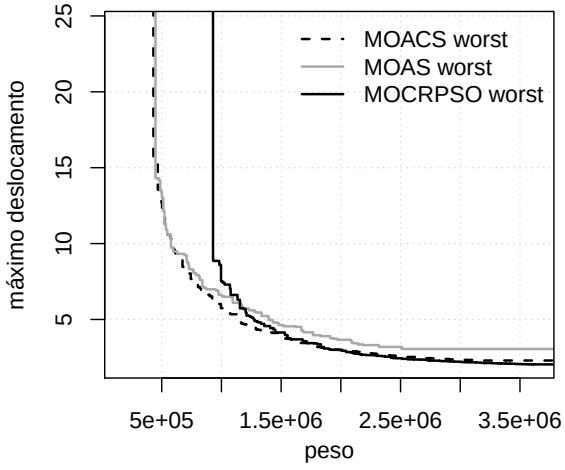
(c) $H_1 = 0.7771$, $H_2 = 0.7404$ e $H_3 = 0.75010$



(d) $H_1 = 0.9059$, $H_2 = 0.9175$ e $H_3 = 0.8799$

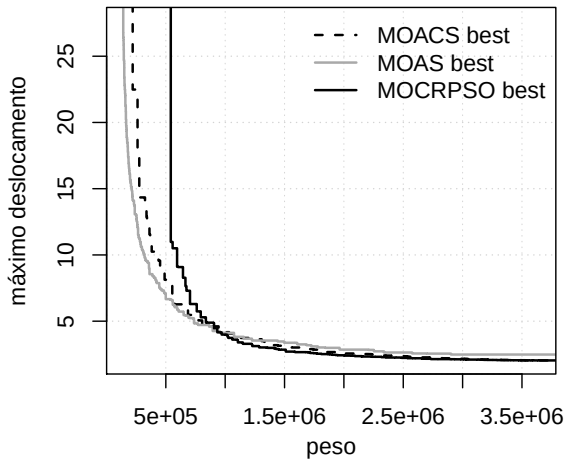


(e) $H_1 = 0.8472$, $H_2 = 0.8501$ e $H_3 = 0.8463$

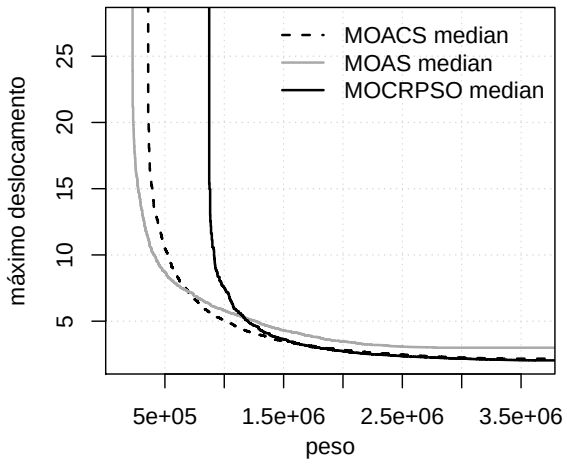


(f) $H_1 = 0.6872$, $H_2 = 0.6461$ e $H_3 = 0.6686$

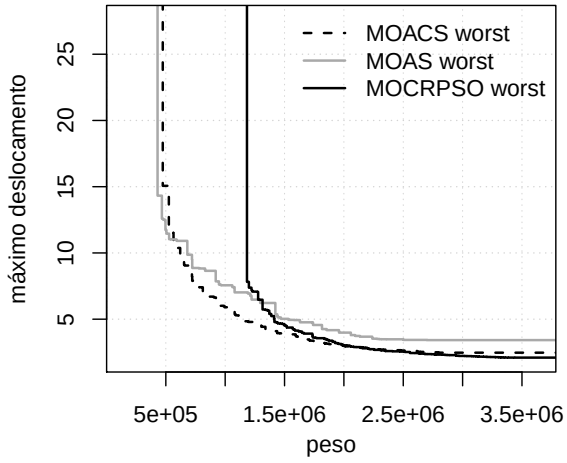
Figura 64 – Curvas EAF da treliça de 942 barras caso discreto, $m = 4$ (figuras (a, b, c) e $m = 8$ (figuras d, e, f). Valores de hipervolume normalizados: H_1 para MOACS, H_2 para MOAS e H_3 para MOCRPSO.



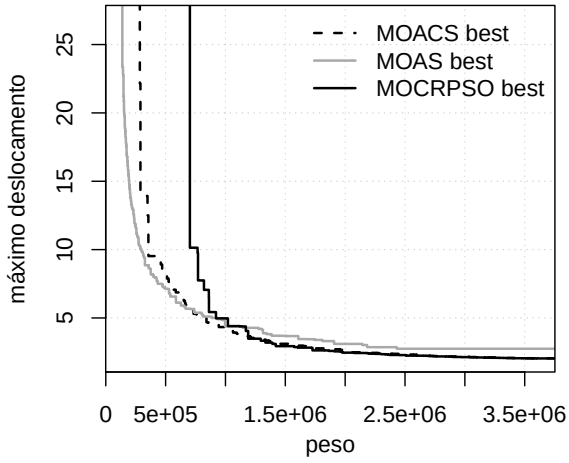
(a) $H_1 = 0.9029$, $H_2 = 0.9162$ e $H_3 = 0.9107$



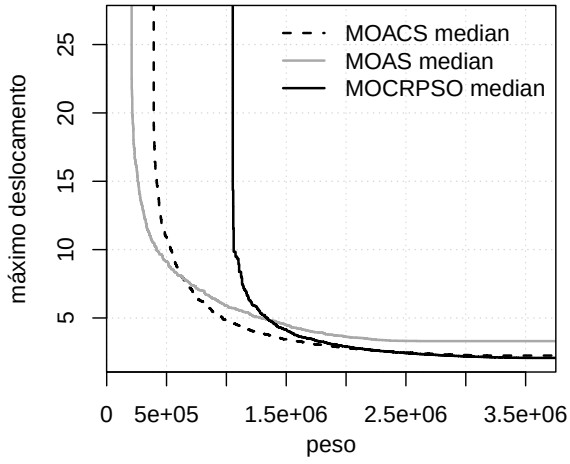
(b) $H_1 = 0.7887$, $H_2 = 0.7894$ e $H_3 = 0.8029$



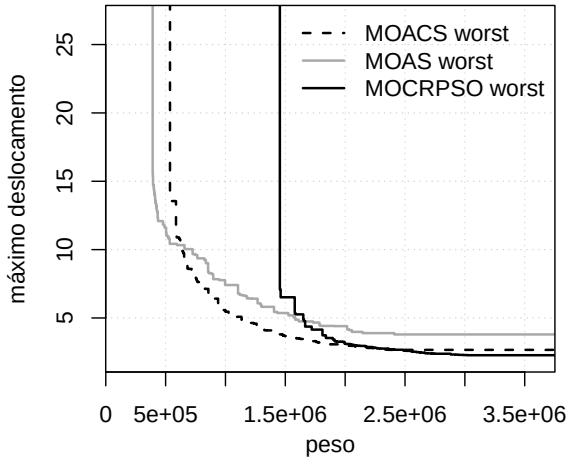
(c) $H_1 = 0.6451$, $H_2 = 0.5757$ e $H_3 = 0.6177$



(d) $H_1 = 0.8764$, $H_2 = 0.8826$ e $H_3 = 0.8533$

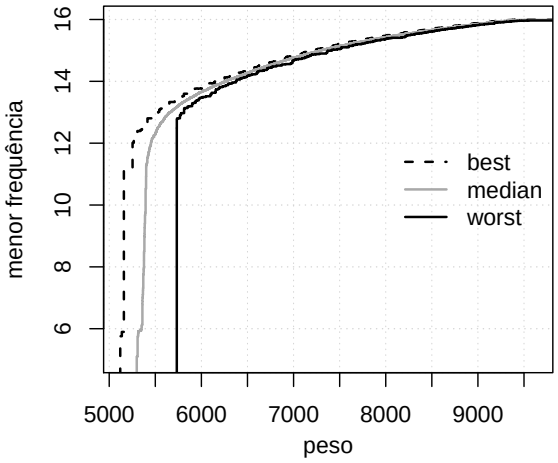


(e) $H_1 = 0.7788$, $H_2 = 0.7743$ e $H_3 = 0.7727$

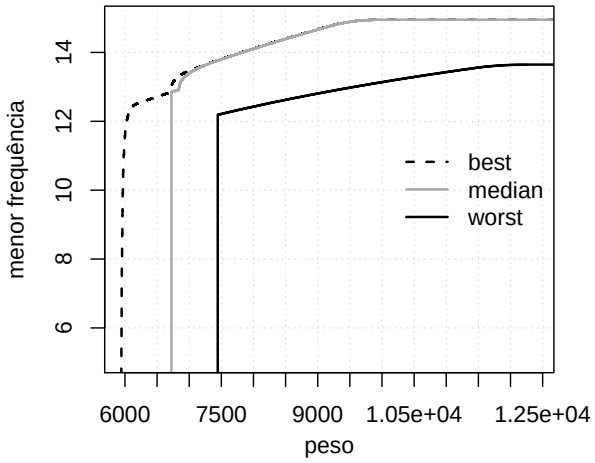


(f) $H_1 = 0.5800$, $H_2 = 0.5245$ e $H_3 = 0.3637$

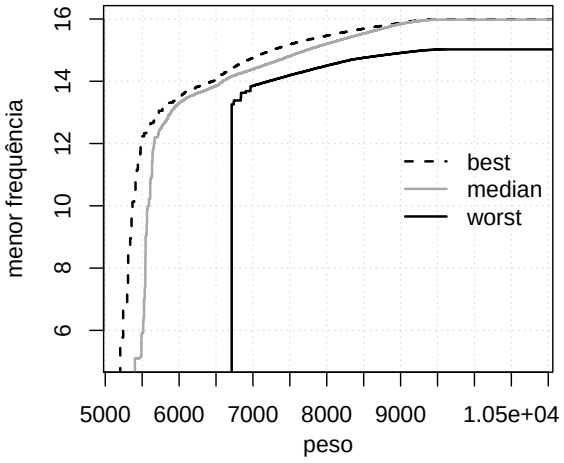
Figura 65 – Curvas EAF da treliça de 942 barras caso discreto, $m = 16$ (figuras a, b, c) e $m = 32$ (figuras (d, e, f)). Valores de hipervolume normalizados: H_1 para MOACS, H_2 para MOAS e H_3 para MOCRPSO.



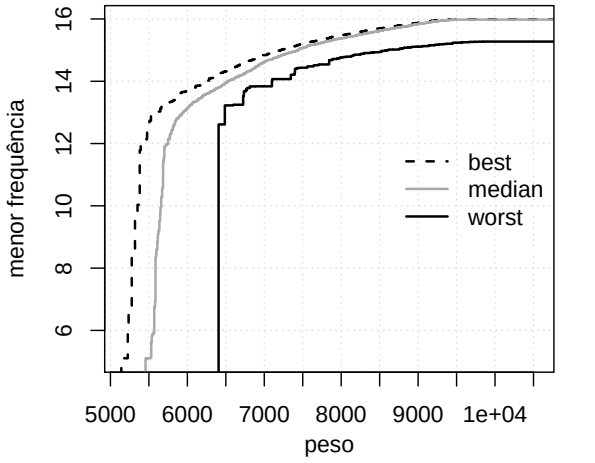
(a) sem r.c.



(b) $m = 2$

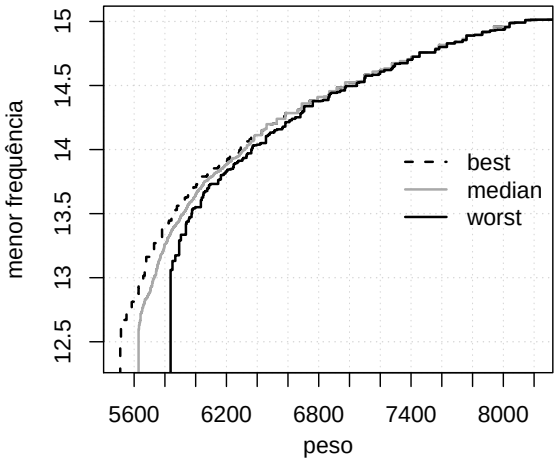


(c) $m = 4$

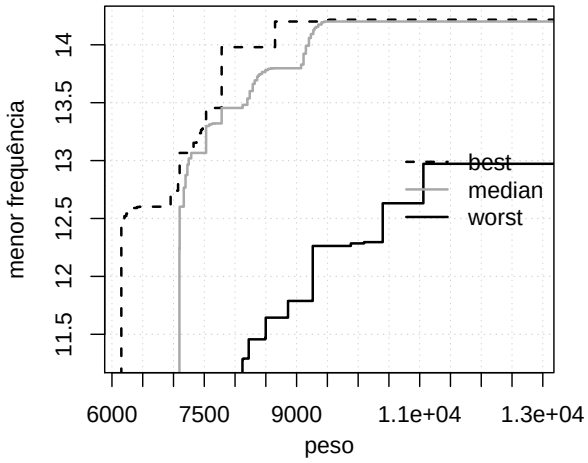


(d) $m = 8$

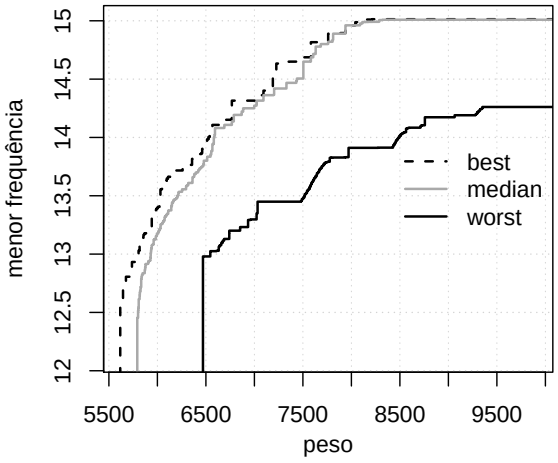
Figura 66 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 10 barras caso contínuo - caso I.



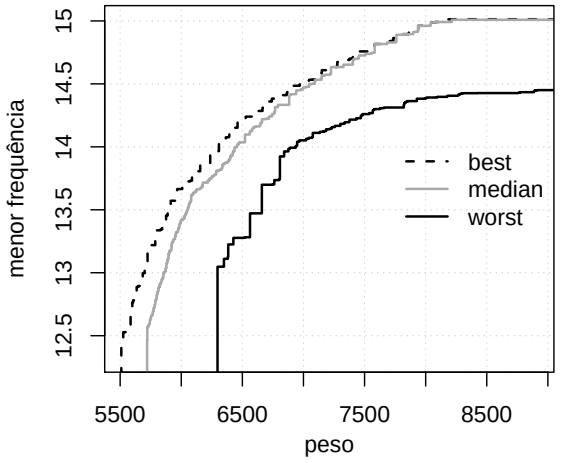
(a) sem r.c.



(b) $m = 2$

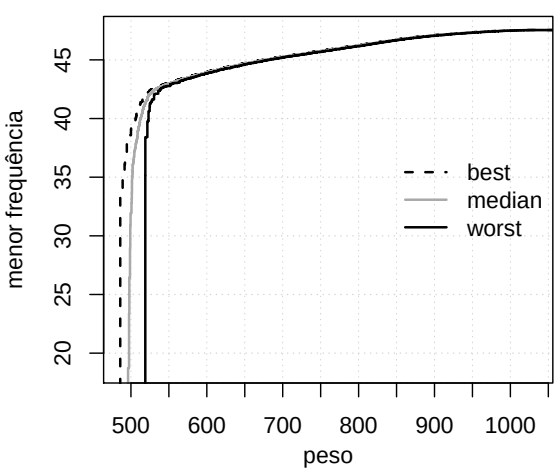


(c) $m = 4$

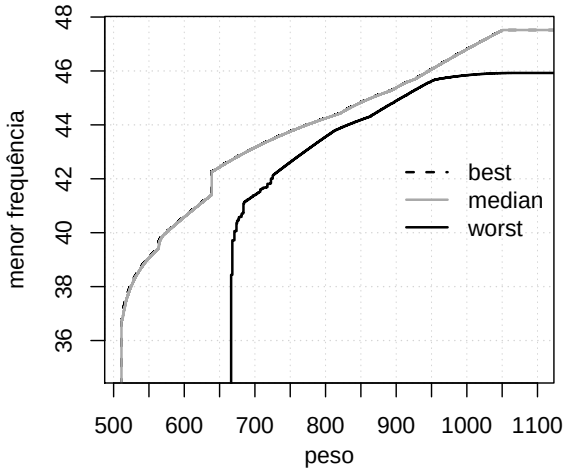


(d) $m = 8$

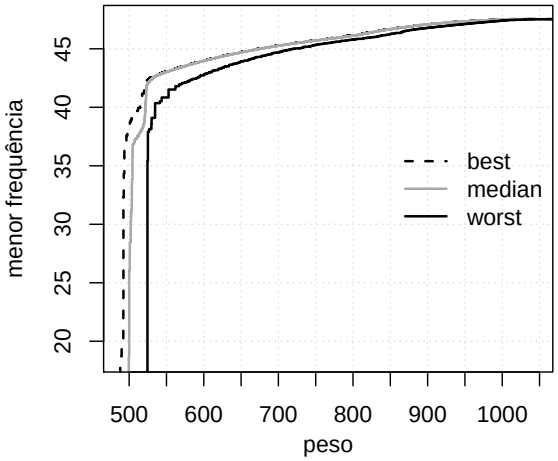
Figura 67 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 10 barras caso discreto - caso I.



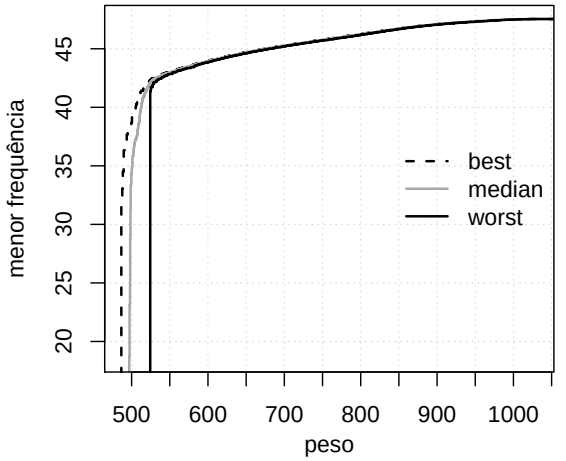
(a) sem r.c.



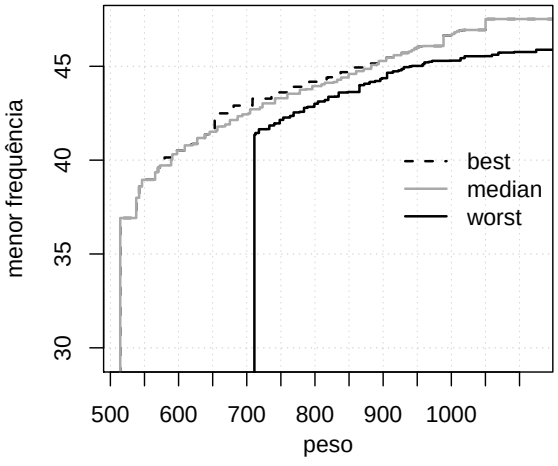
(b) $m = 2$



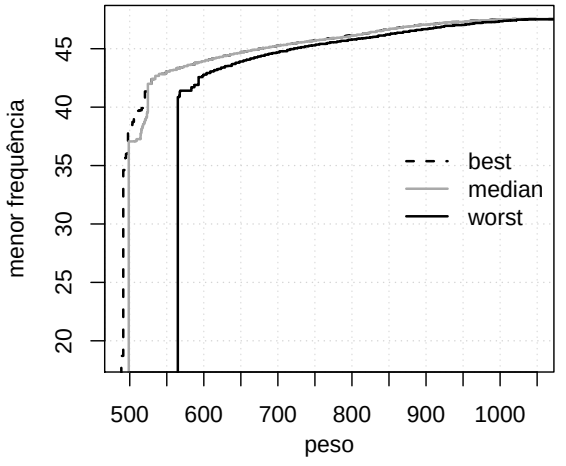
(c) $m = 4$



(d) sem r.c.

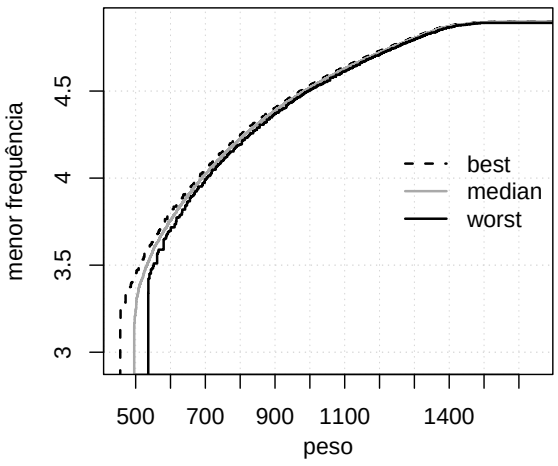


(e) $m = 2$

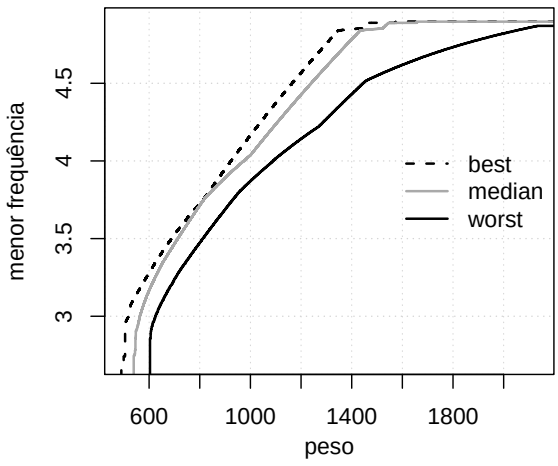


(f) $m = 4$

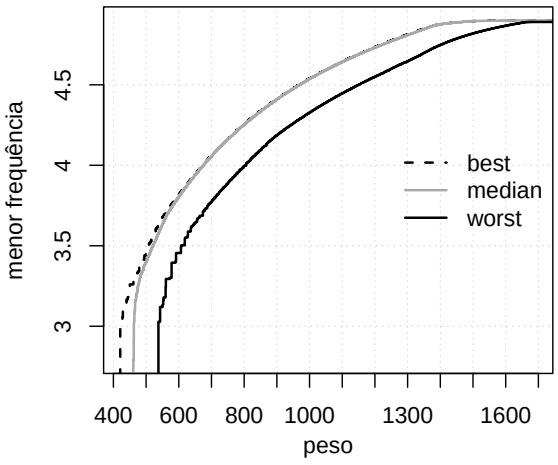
Figura 68 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para as treliças de 25 barras caso contínuo (figuras a, b, c) e caso discreto (figuras d, e, f) - caso I.



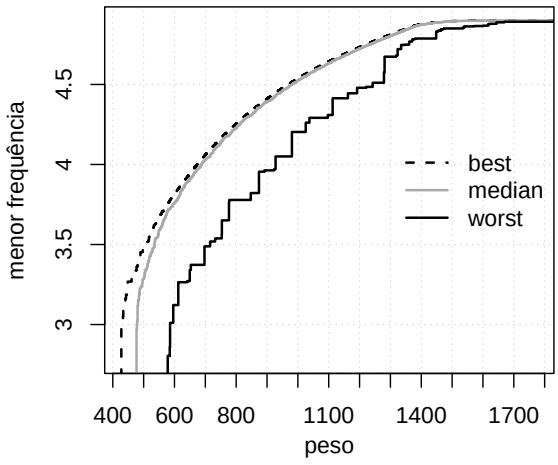
(a) sem r.c.



(b) $m = 2$

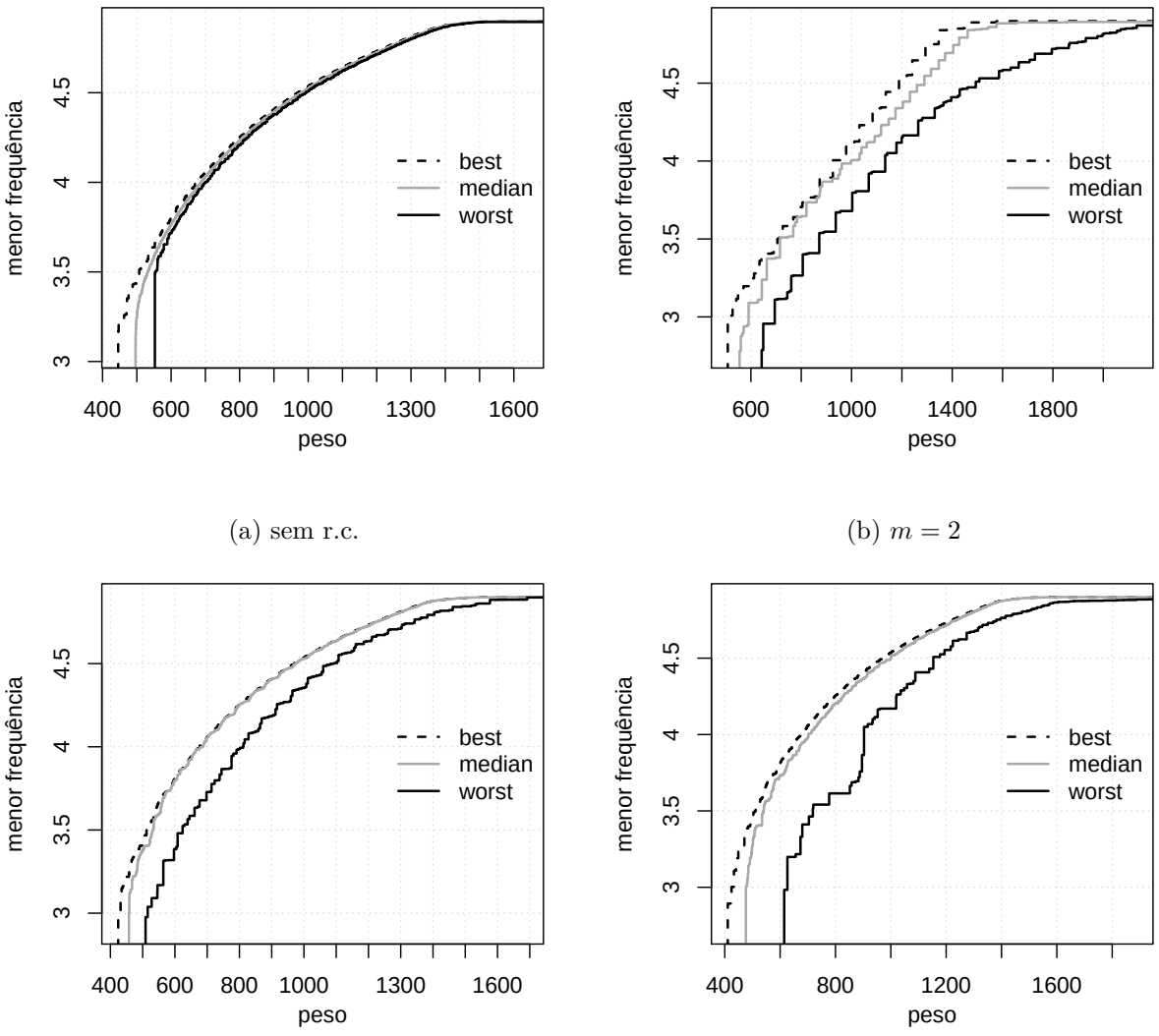


(c) $m = 4$



(d) $m = 8$

Figura 69 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 72 barras caso contínuo - caso I.



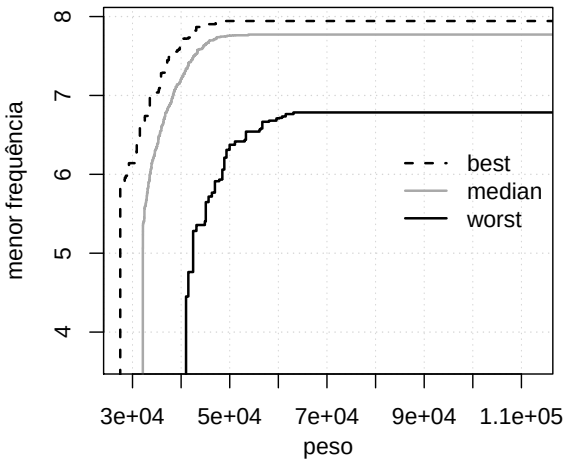
(a) sem r.c.

(b) $m = 2$

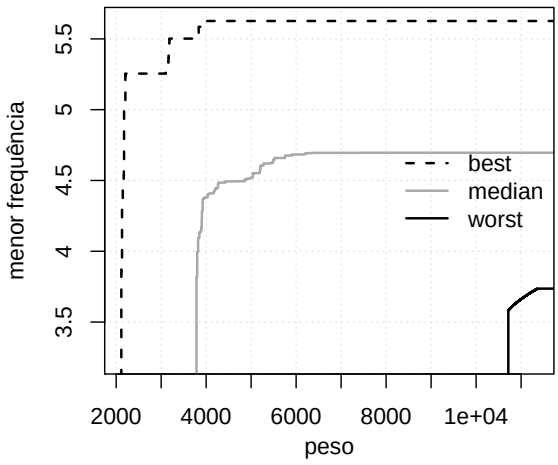
(c) $m = 4$

(d) $m = 8$

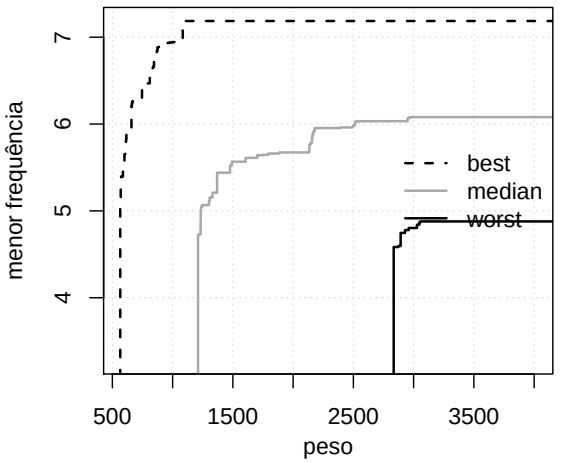
Figura 70 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 72 barras caso discreto - caso I.



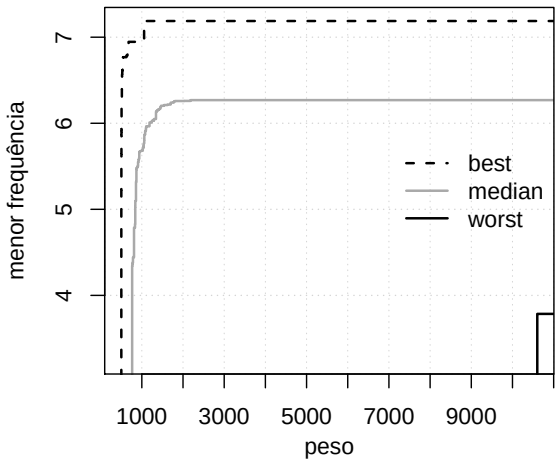
(a) sem r.c.



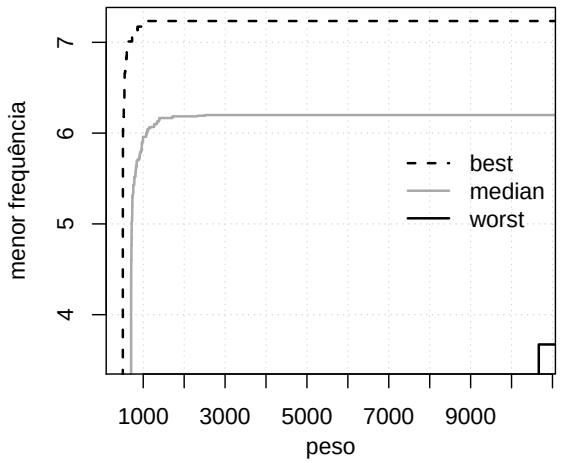
(b) $m = 2$



(c) $m = 4$

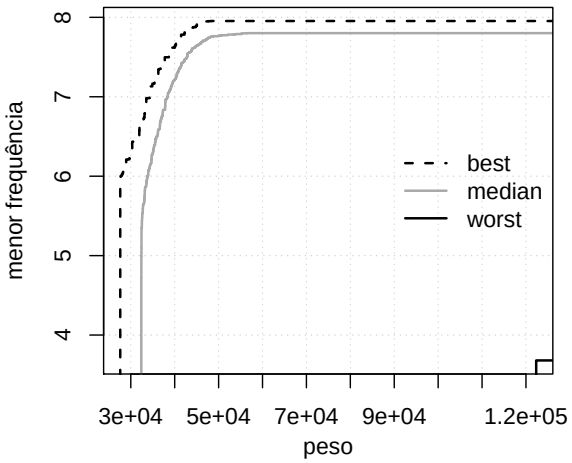


(d) $m = 8$

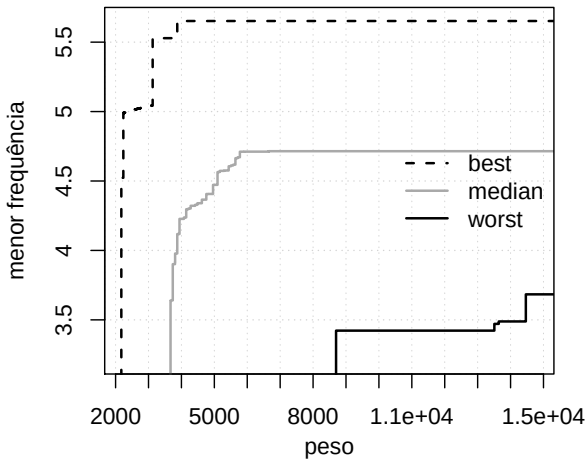


(e) $m = 16$

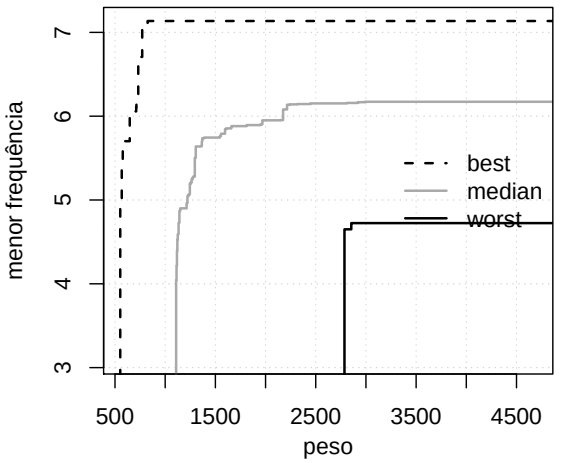
Figura 71 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 200 barras caso contínuo - caso I.



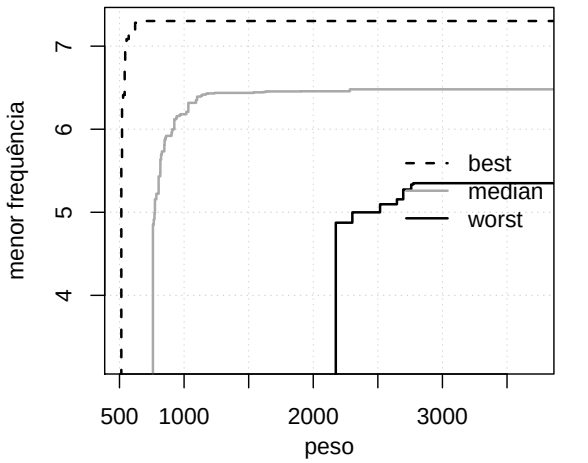
(a) sem r.c.



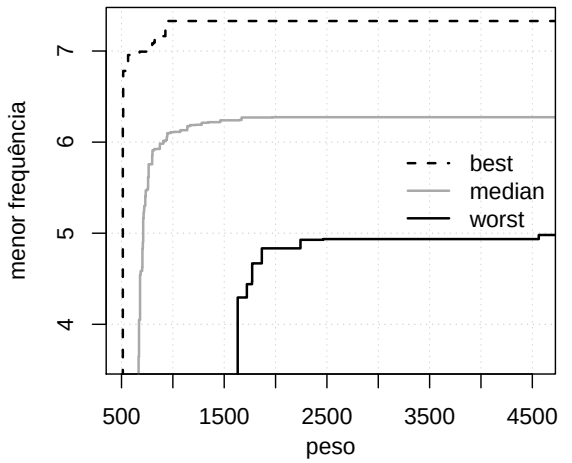
(b) $m = 2$



(c) $m = 4$



(d) $m = 8$



(e) $m = 16$

Figura 72 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 200 barras caso discreto - caso I.

6.3.2 **Caso II**

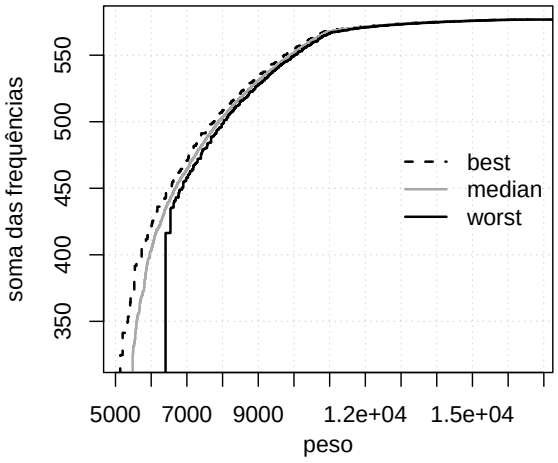
O caso II consiste em encontrar um conjunto de variáveis de projeto que correspondem às áreas da seções transversais das barras $(A_1, \ldots, A_{n_a})$ da estrutura que minimiza seu peso e que maximiza a soma das frequências naturais de vibração. Para estas análises, todas as frequências naturais da estrutura foram consideradas no somatório. O problema é formulado como

$$\begin{gathered} \min \quad W = \sum_{i=1}^{n_a} \rho A_i L_i \\ \text{e} \\ \max \quad \sum_{j=1}^{n_f} \omega_j \end{gathered} \tag{6.11}$$

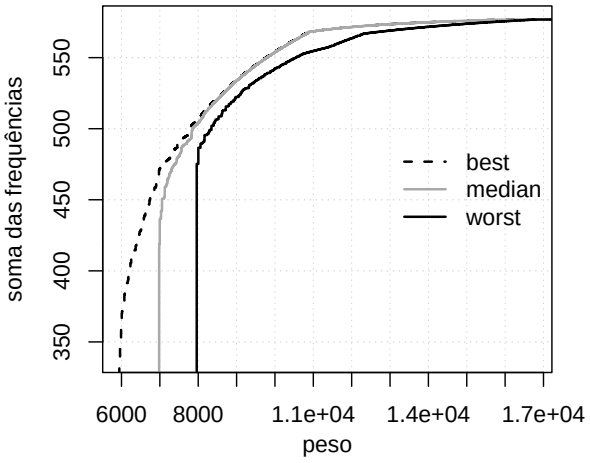
sujeito a

$$\begin{gathered} \frac{\sigma_{il}}{\bar{\sigma}} - 1 \leq 0, \quad i = 1, \ldots, n_a, \; l = 1, \ldots, n_l \\ \text{e} \\ \frac{u_{ql}}{\bar{u}} - 1 \leq 0, \quad q = 1, \ldots, n_d, \; l = 1, \ldots, n_l. \end{gathered} \tag{6.12}$$

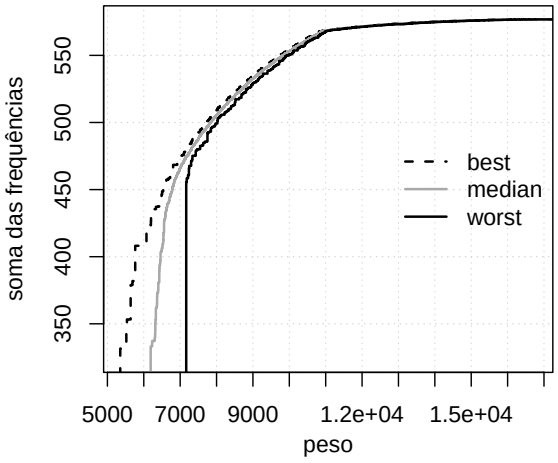
Para o caso II, as curvas EAF (*best*, *median* e *worst*) são apresentadas nas Figs. 73 a 79.



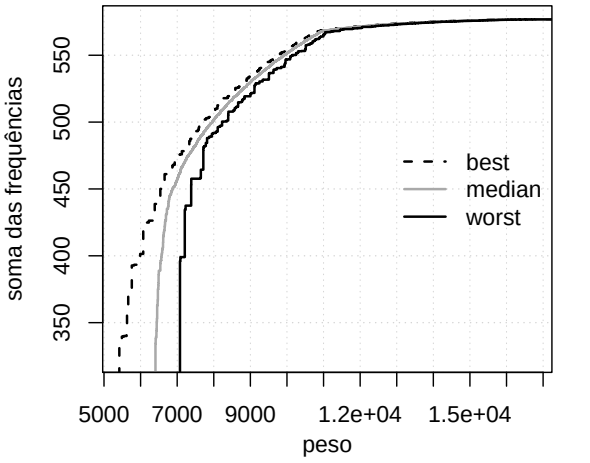
(a) sem r.c.



(b) $m = 2$

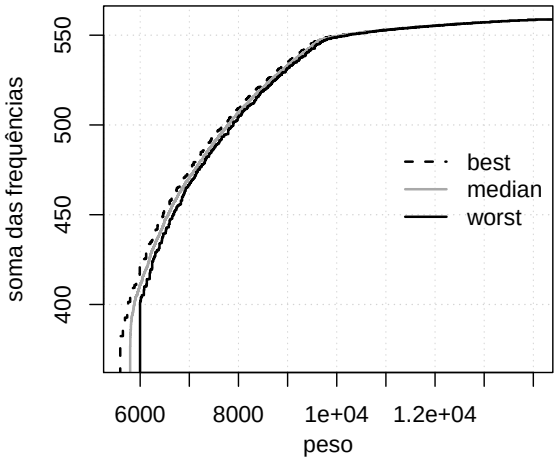


(c) $m = 4$

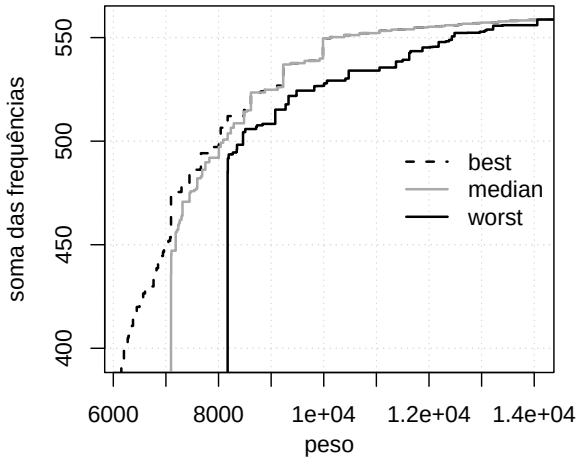


(d) $m = 8$

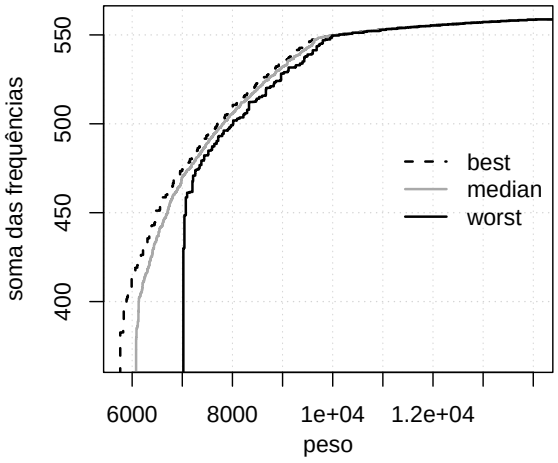
Figura 73 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 10 barras caso contínuo - caso II.



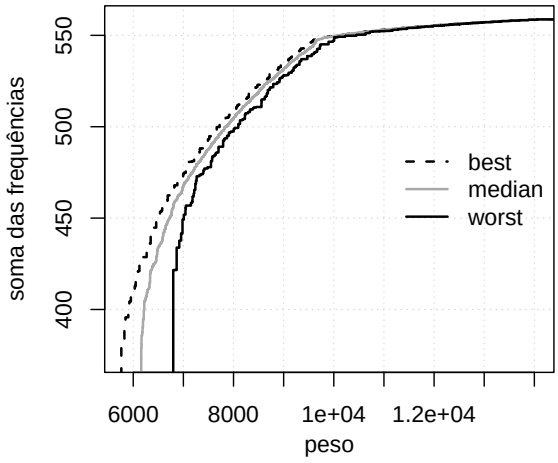
(a) sem r.c.



(b) $m = 2$

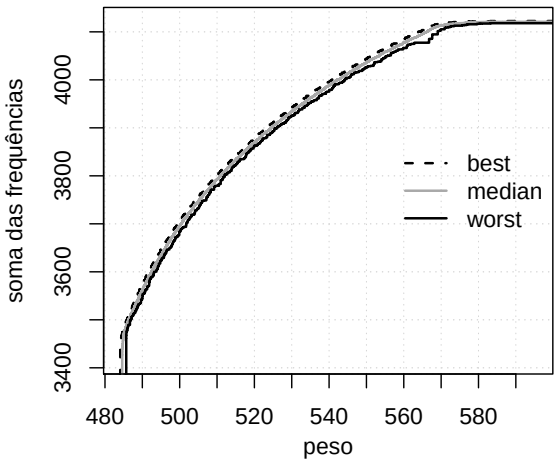


(c) $m = 4$

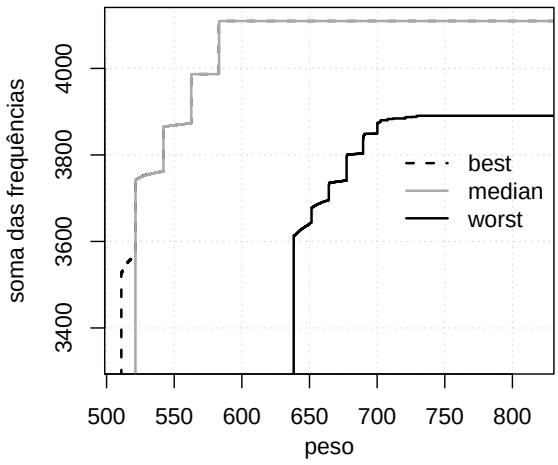


(d) $m = 8$

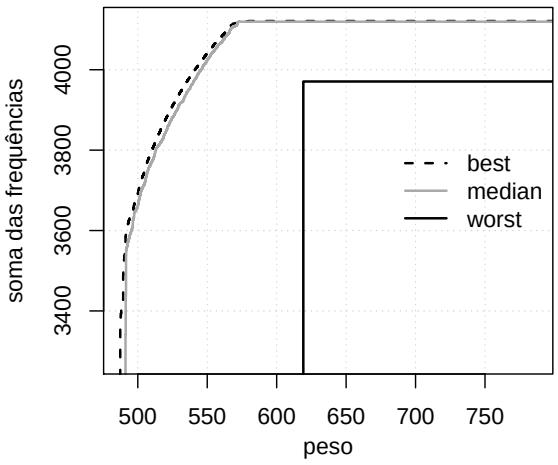
Figura 74 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 10 barras caso discreto - caso II.



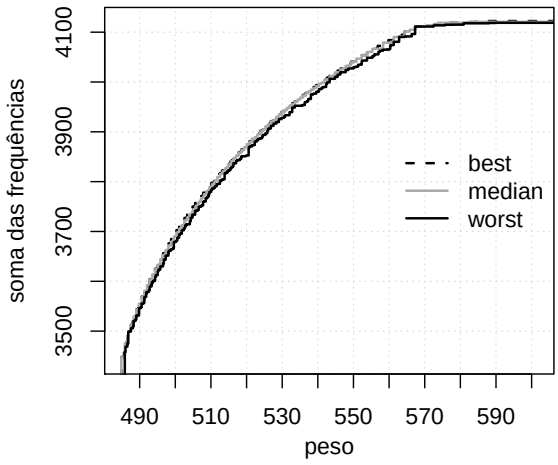
(a) sem r.c.



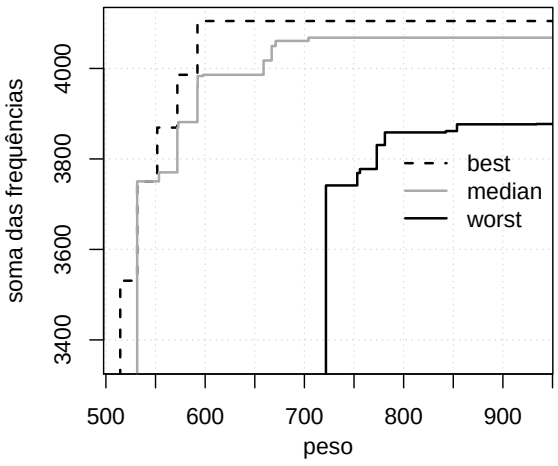
(b) $m = 2$



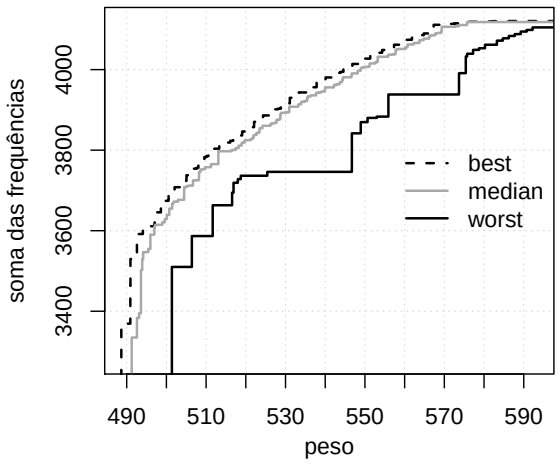
(c) $m = 4$



(d) sem r.c.

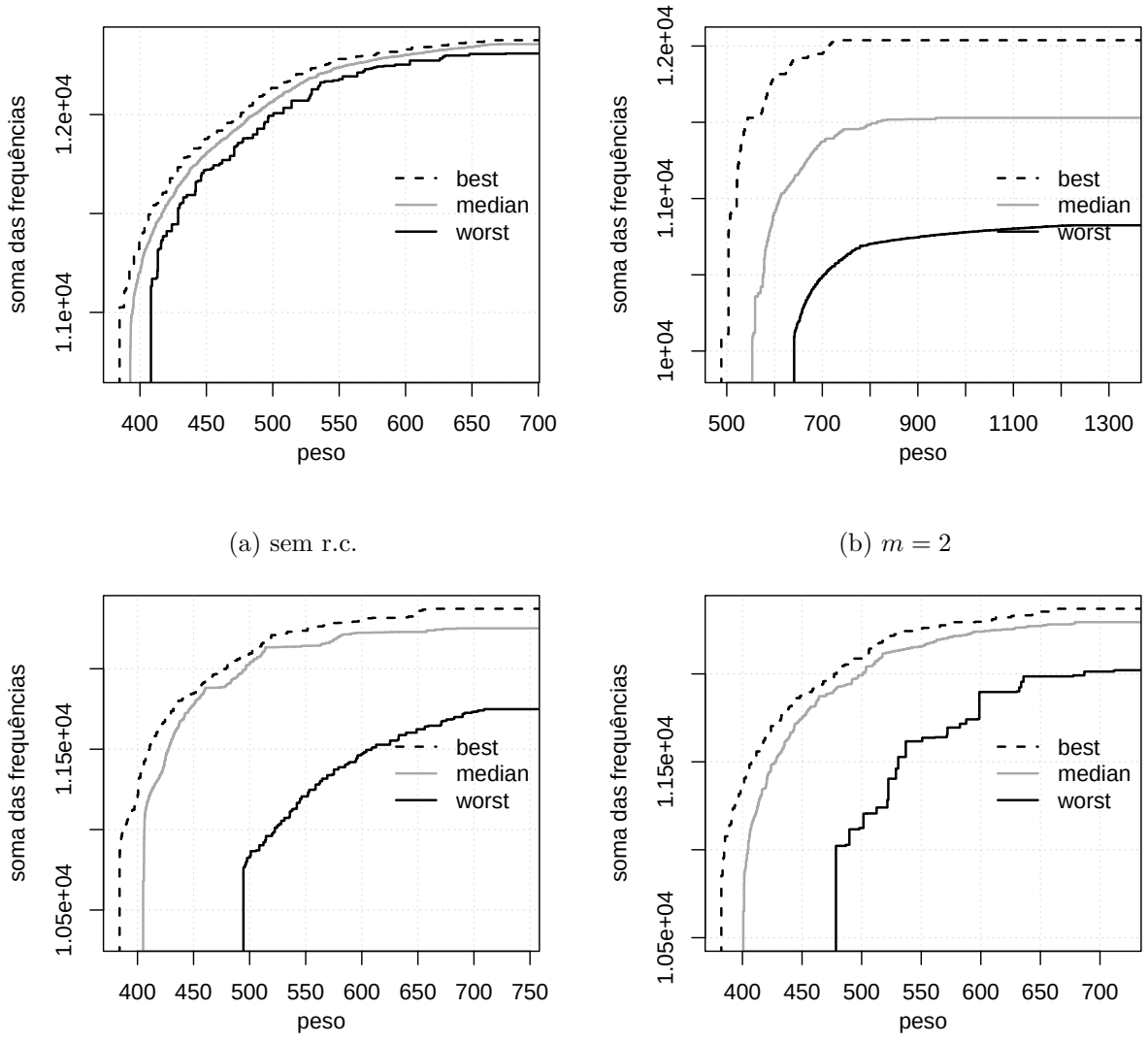


(e) $m = 2$



(f) $m = 4$

Figura 75 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para as treliças de 25 barras caso contínuo (figuras a, b, c) e caso discreto (figuras d, e, f) - caso II.



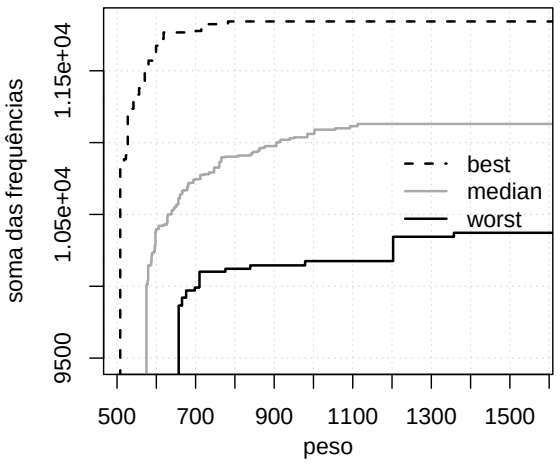
(a) sem r.c.

(b) $m = 2$

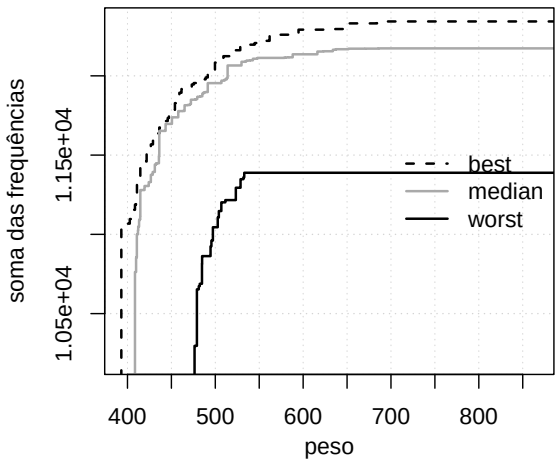
(c) $m = 4$

(d) $m = 8$

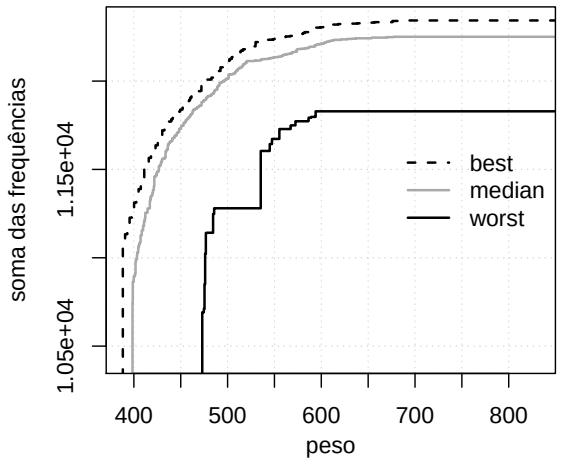
Figura 76 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 72 barras caso contínuo - caso II.



(a) $m = 2$

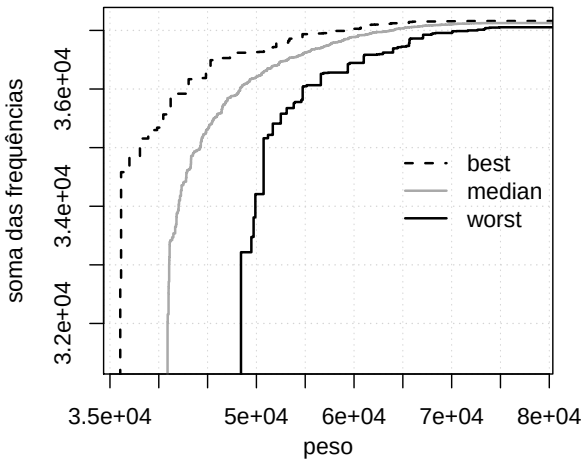


(b) $m = 4$

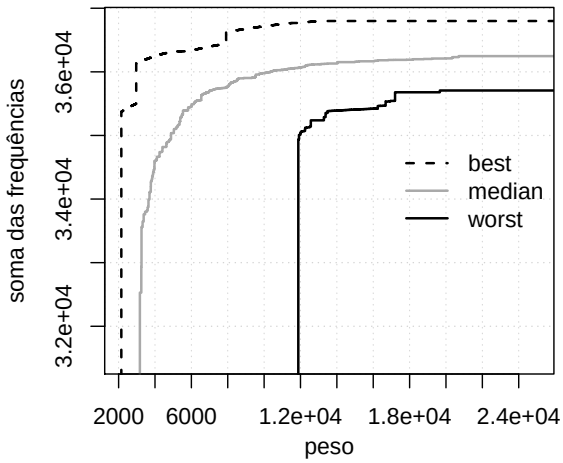


(c) $m = 8$

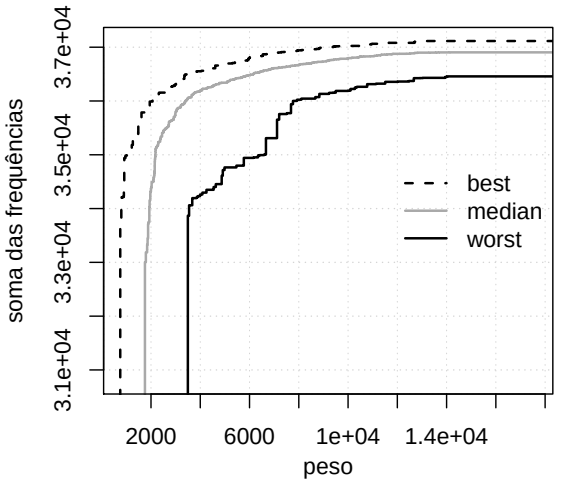
Figura 77 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 72 barras caso discreto - caso II.



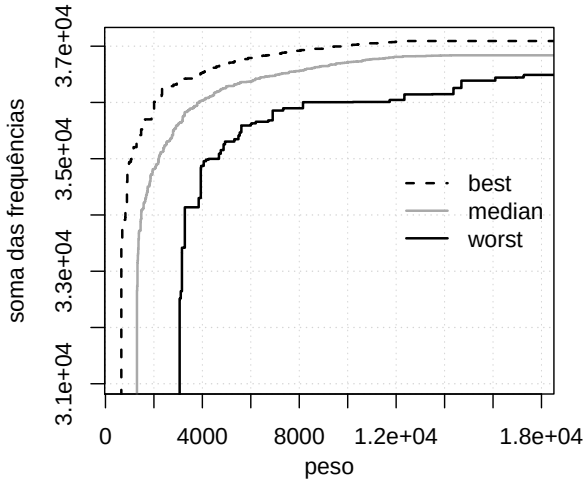
(a) sem r.c.



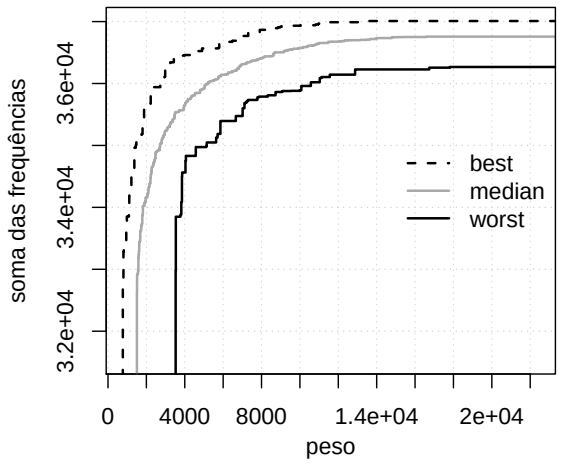
(b) $m = 2$



(c) $m = 4$

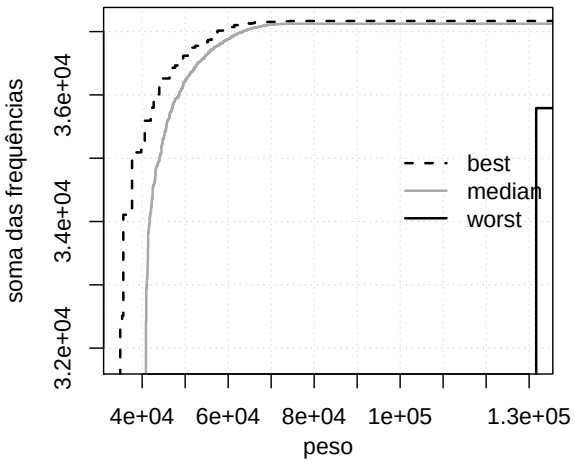


(d) $m = 8$

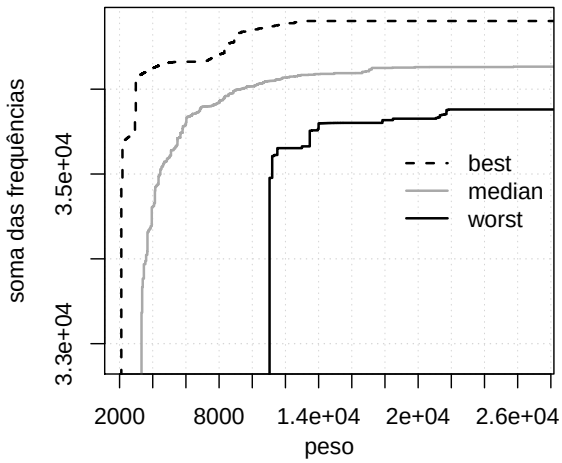


(e) $m = 16$

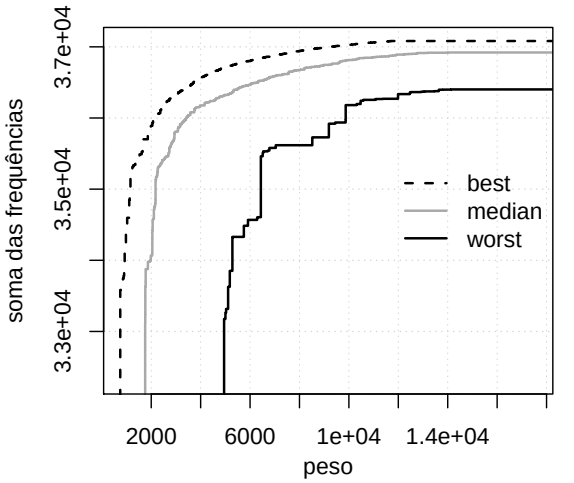
Figura 78 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 200 barras caso contínuo - caso II.



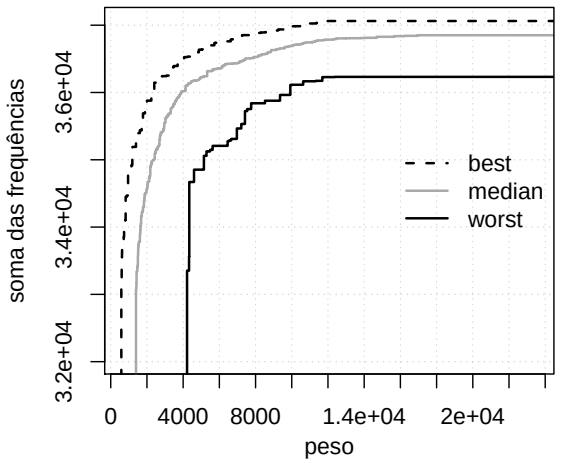
(a) sem r.c.



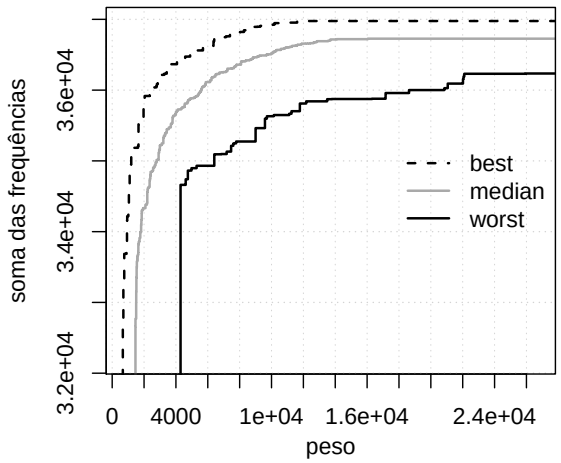
(b) $m = 2$



(c) $m = 4$



(d) $m = 8$



(e) $m = 16$

Figura 79 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 200 barras caso discreto - caso II.

6.3.3 **Caso III**

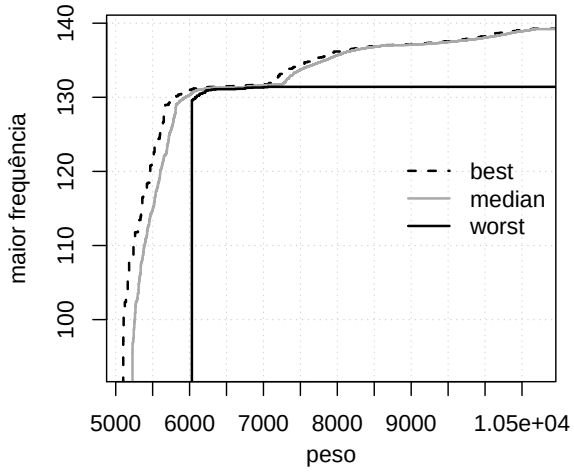
Por último, o caso III consiste em encontrar um conjunto de variáveis de projeto que correspondem às áreas da seções transversais das barras $(A_1, \ldots, A_{n_a})$ da estrutura que minimiza seu peso e que maximiza a maior frequência natural de vibração. O problema é formulado como

$$\begin{gathered} \min \quad W = \sum_{i=1}^{n_a} \rho A_i L_i \\ \text{e} \\ \max \quad \text{máximo } (\omega_j),\ j = 1, \ldots, n_f, \end{gathered} \tag{6.13}$$

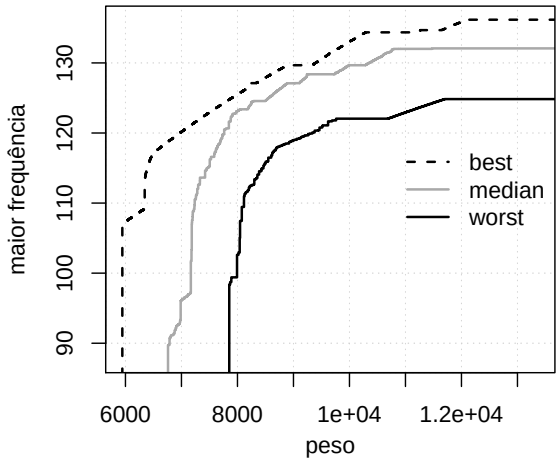
sujeito a

$$\begin{gathered} \frac{\sigma_{il}}{\bar{\sigma}} - 1 \leq 0, \quad i = 1, \ldots, n_a,\ l = 1, \ldots, n_l \\ \text{e} \\ \frac{u_{ql}}{\bar{u}} - 1 \leq 0, \quad q = 1, \ldots, n_d,\ l = 1, \ldots, n_l. \end{gathered} \tag{6.14}$$

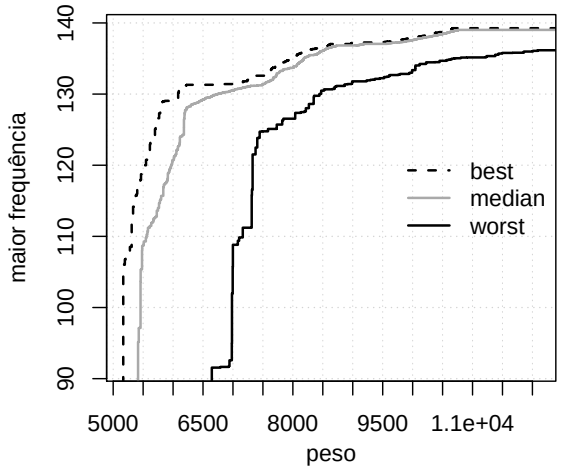
Para o último caso, as Figs. 80 a 86 apresentam as curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO.



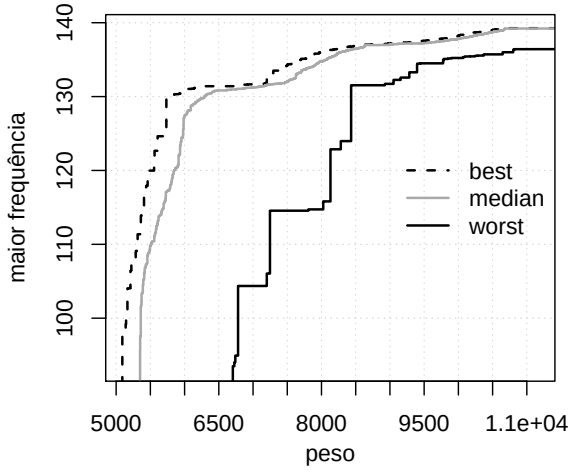
(a) sem r.c.



(b) $m = 2$

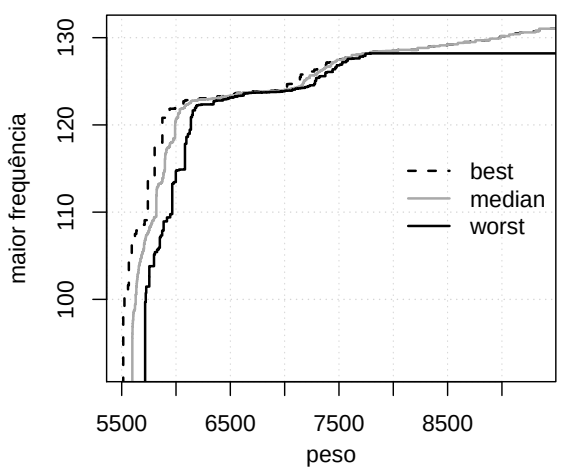


(c) $m = 4$

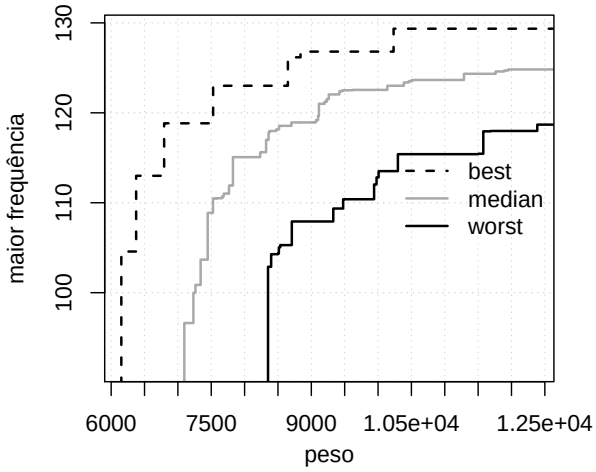


(d) $m = 8$

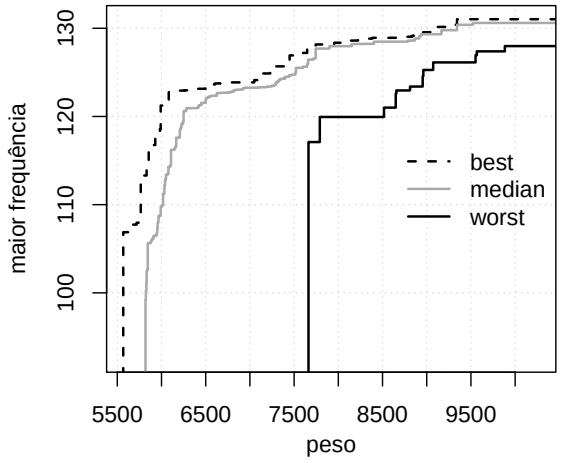
Figura 80 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 10 barras caso contínuo - caso III.



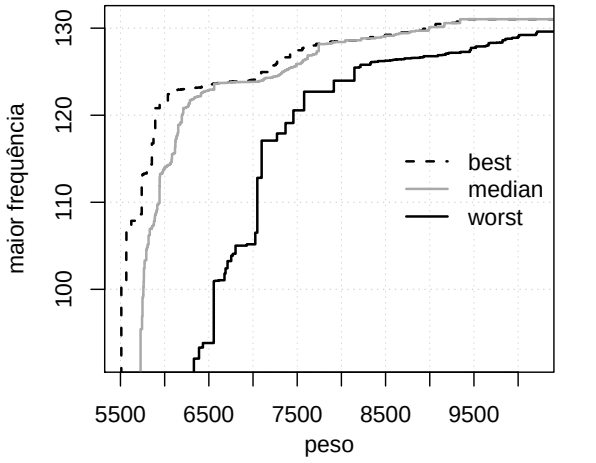
(a) sem r.c.



(b) $m = 2$

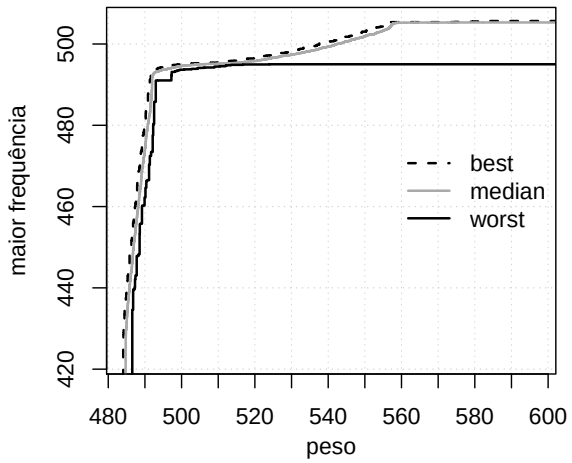


(c) $m = 4$

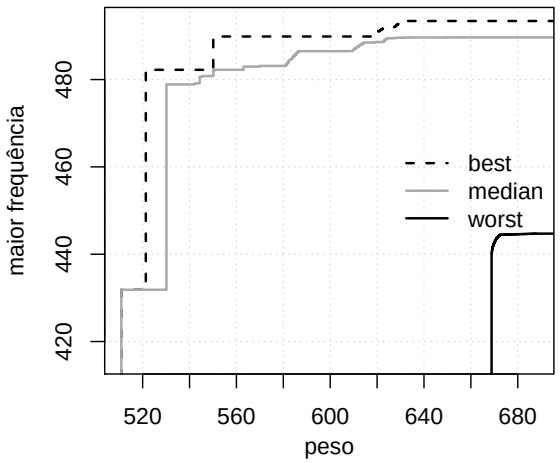


(d) $m = 8$

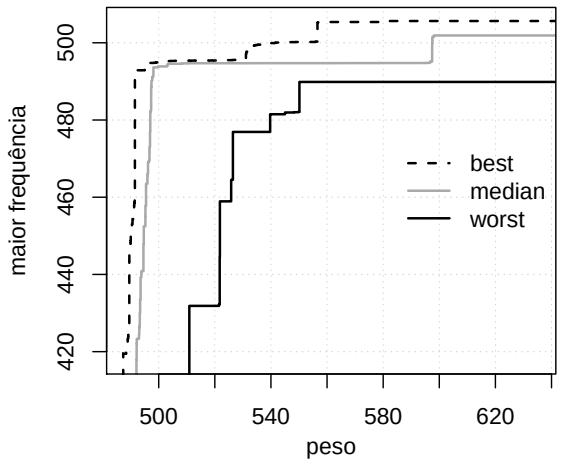
Figura 81 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 10 barras caso discreto - caso III.



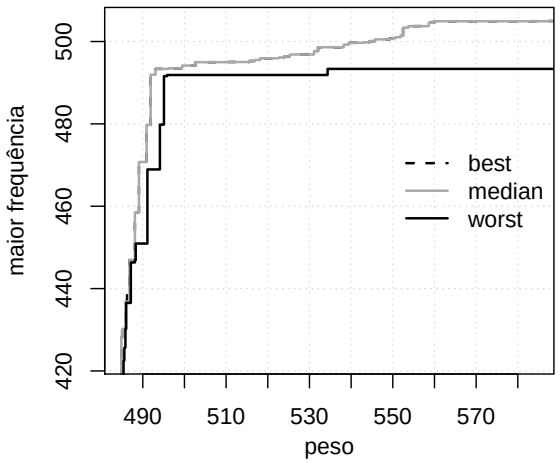
(a) sem r.c.



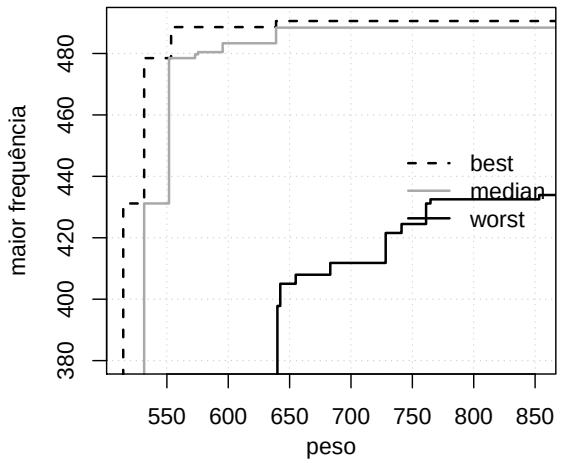
(b) $m = 2$



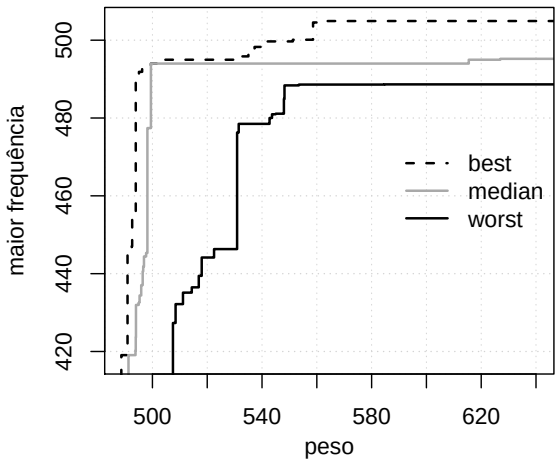
(c) $m = 4$



(d) sem r.c.

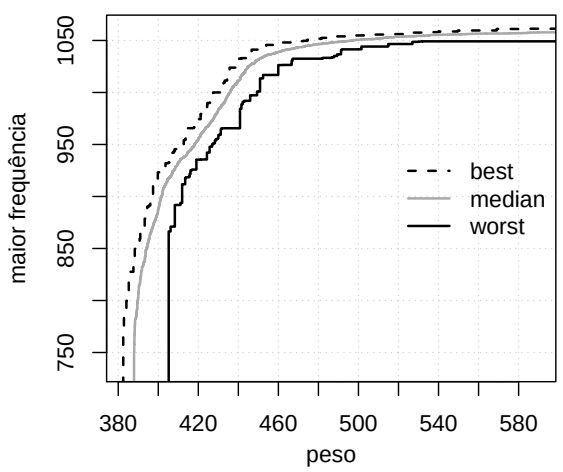


(e) $m = 2$

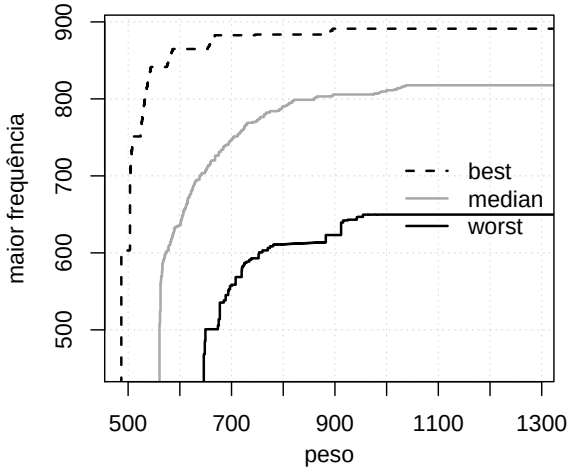


(f) $m = 4$

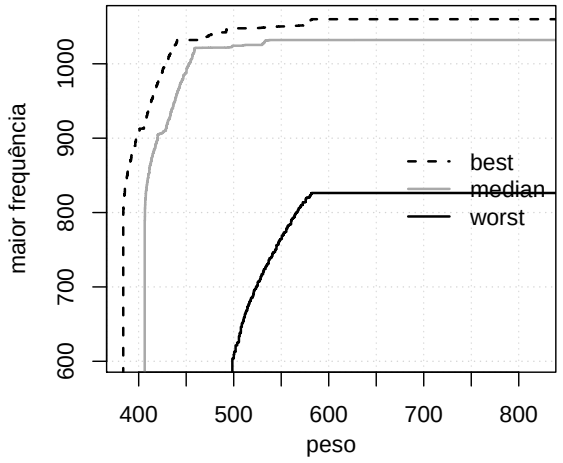
Figura 82 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para as treliças de 25 barras caso contínuo (figuras a, b, c) e caso discreto (figuras d, e, f) - caso III.



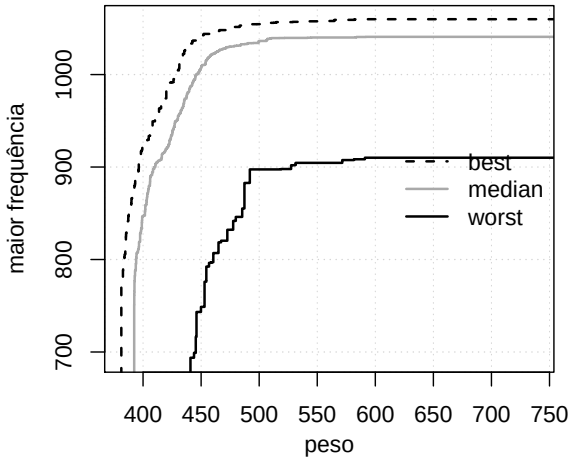
(a) sem r.c.



(b) $m = 2$

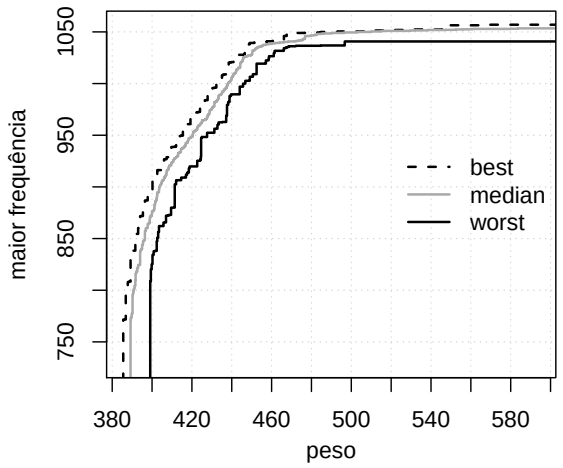


(c) $m = 4$

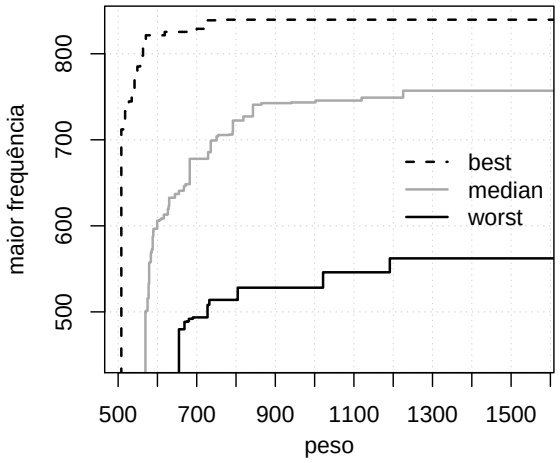


(d) $m = 8$

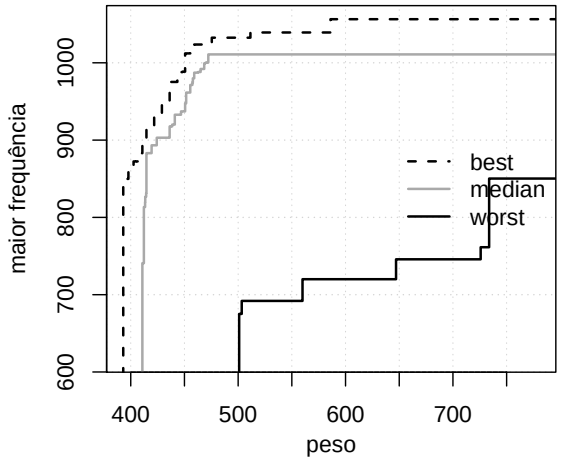
Figura 83 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 72 barras caso contínuo - caso III.



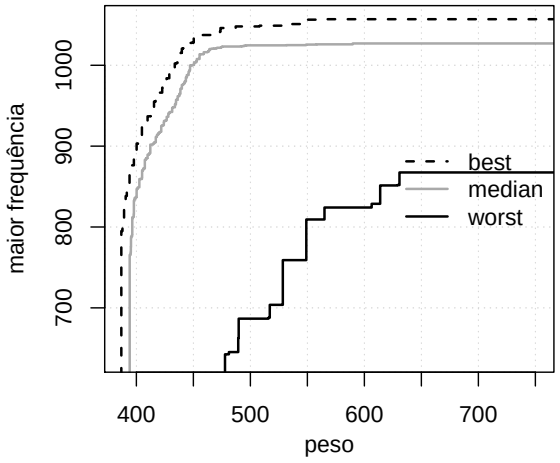
(a) sem r.c.



(b) $m = 2$

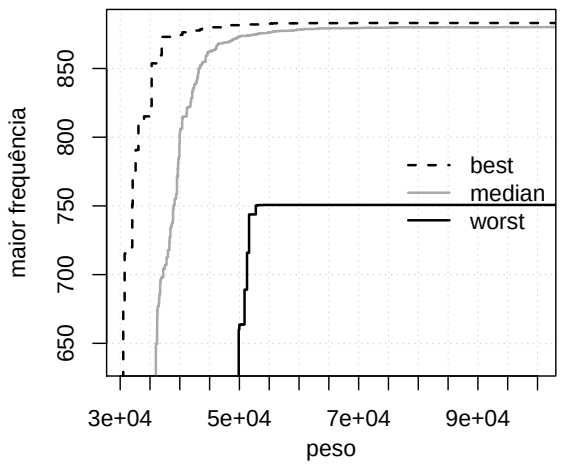


(c) $m = 4$

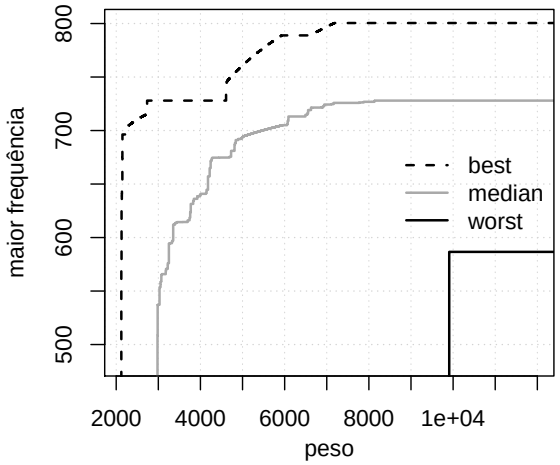


(d) $m = 8$

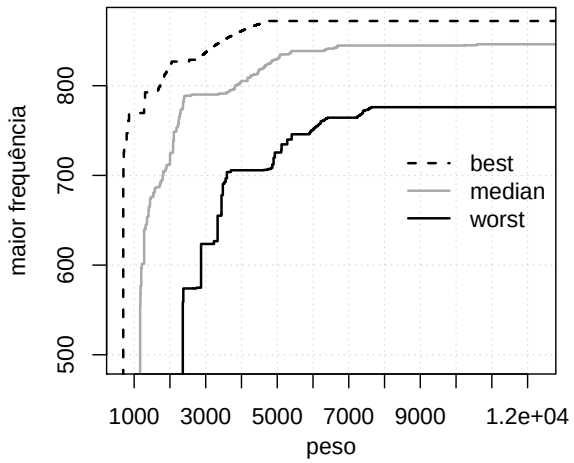
Figura 84 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 72 barras caso discreto - caso III.



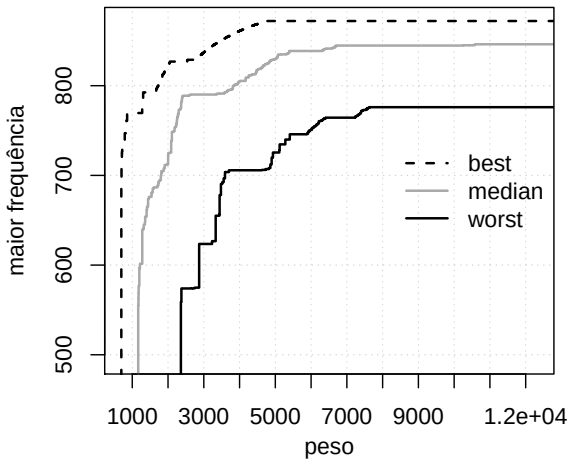
(a) sem r.c.



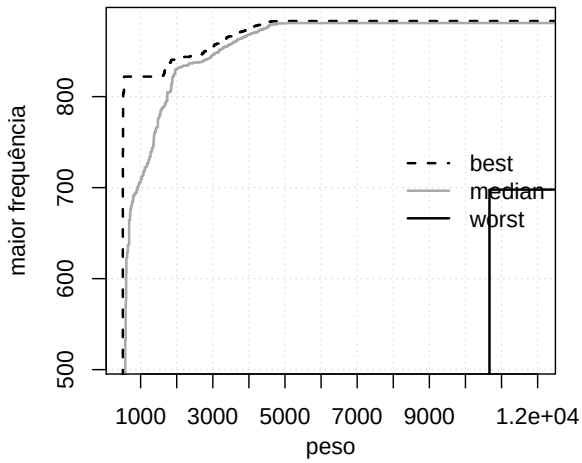
(b) $m = 2$



(c) $m = 4$

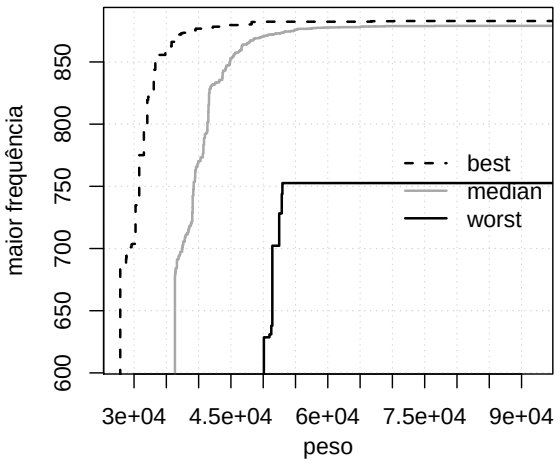


(d) $m = 8$

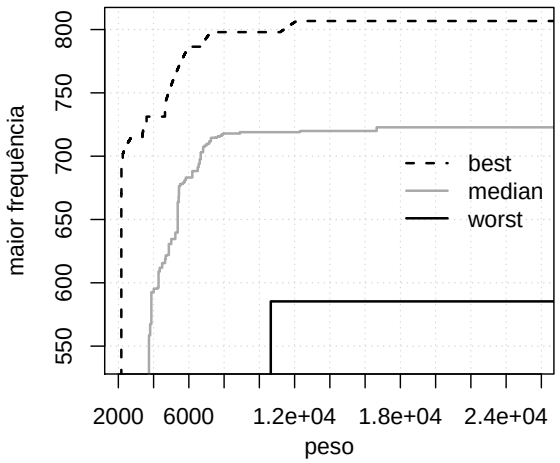


(e) $m = 16$

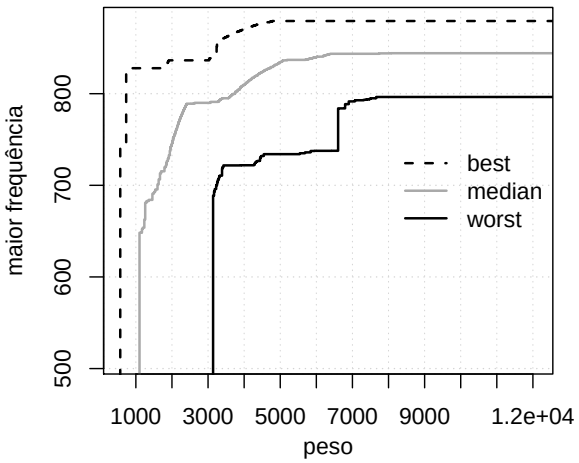
Figura 85 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 200 barras caso contínuo - caso III.



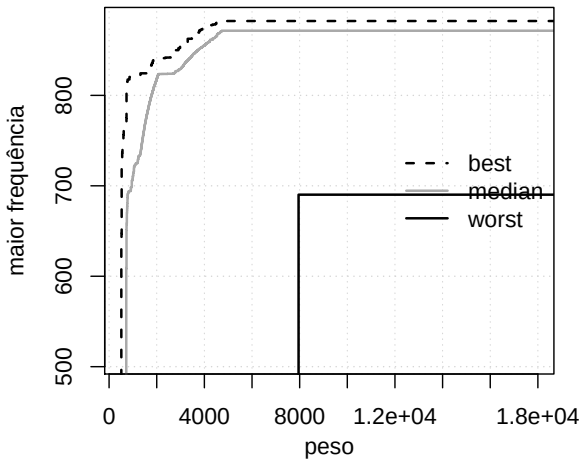
(a) sem r.c.



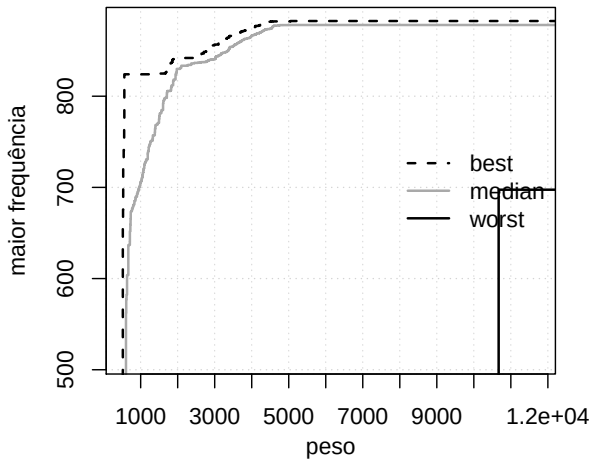
(b) $m = 2$



(c) $m = 4$



(d) $m = 8$



(e) $m = 16$

Figura 86 – Curvas EAF (*best*, *median* e *worst*) do algoritmo MOCRPSO para a treliça de 200 barras caso discreto - caso III.

6.3.4 **Análise dos resultados**

Não foram encontrados trabalhos na literatura que abordam a mesma metodologia apresentada nesse terceiro conjunto de experimentos, o que reforça o caráter inovador desta tese. Por conta disso, não foi possível propor comparações envolvendo outros algoritmos. E ainda, o número considerável de análises desses resultados não remetem a uma comparação caso a caso, o que seria bastante exaustivo e, talvez, pouco conclusivo. Desta forma, uma orientação para o uso desses resultados no que diz respeito à análise das fronteiras de Paretos, pode ser feita da seguinte maneira:

1. o tomador de decisões define qual o problema multiobjetivo que é do seu interesse;
2. define se analisará as soluções com ou sem cardinalidade (agrupamentos);
    2.1. se a decisão for pelo uso das soluções com o uso de cardinalidade, escolherá qual delas será analisada (qual o valor de m);
3. analisa a frente de Pareto referente a sua escolha.

Diante disso, um novo conjunto de experimentos fica disponível na literatura e pode ser utilizado em futuros estudos e/ou comparações.

Entretanto, uma análise envolvendo os resultados obtidos pelo MOCRPSO não pode ser desprezada. Os aspectos obtidos nas curvas EAF em todos os casos estão dentro da expectativa. As soluções com os maiores pesos apresentaram os maiores valores para a menor frequência, soma das frequências e também para a maior frequência. Um ponto interessante observado é que o aspecto assintótico das curvas, principalmente os das curvas das melhores soluções (*best*), teve início para valores de peso razoavelmente pequenos, indicando um intervalo de preferência interessante a ser investigado pelo tomador de decisão.

Neste caso, a expectativa natural do tomador de decisão é de ter configurações estruturais com valores de massa excessivos, o que pode não ser necessário para a obtenção das frequências dentro de determinados aspectos. A Fig 87 apresenta um exemplo de um resultado da treliça de 200 barras (caso discreto) para o caso I apontando um intervalo de preferência. A curva EAF mostra que a solução obtida no ponto A possui um valor de frequência natural igual ao da solução obtida no ponto B. Contudo, o valor do peso final da estrutura no ponto A é consideravelmente menor que o do ponto B. É importante salientar que os pontos A e B não representam necessariamente duas soluções. Eles identificam apenas uma possível região de interesse.

Esse aspecto faz com que os tomadores de decisão tenham a tarefa de identificar dentro de um conjunto de soluções uma ou mais soluções que melhor satisfazem suas preferências. Contudo, devido à natureza conflitante dos problemas de engenharia, o

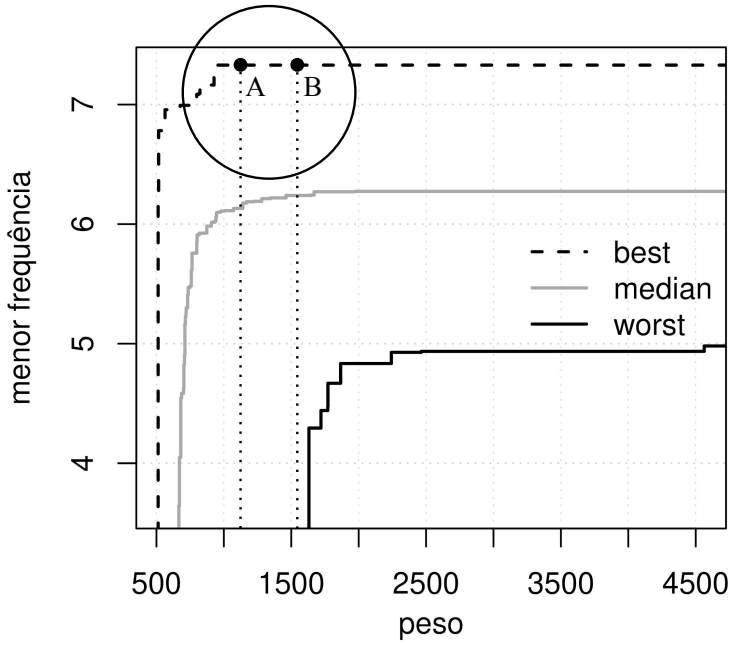


Figura 87 – Região de preferência do resultado da treliça de 200 barras (caso discreto) para o caso I.

número total de soluções pode ser muito grande. E ainda, as opções podem representar fracamente as preferências dos tomadores de decisão.

Algoritmos que utilizam informações sobre as preferências dos tomadores de decisão podem orientar a pesquisa por soluções na região de interesse da Frente de Pareto, evitando aquelas que não representam essas preferências. Isso pode provocar uma melhoria significativa na qualidade das opções de escolha do ponto de vista dos tomadores de decisão, representando suas preferências de um modo bem melhor [6].

Deb & Sundar [166] desenvolveram o algoritmo R-NSGA-II (*Reference Point based NSGA-II*), que funciona como o NSGA-II, porém as informações de preferência são adicionadas no momento da seleção dos indivíduos que terão preferência aqueles que estiverem mais próximos dos pontos de referência. Um estudo recente foi desenvolvido por Vargas [6] e incorpora informações de preferência ao algoritmo GDE3-APM. As informações de preferência são inseridas da mesma forma que é utilizado no R-NSGA-II, e o algoritmo é denominado como R-GDE3-APM.

Em uma análise subsequente em cada uma das estruturas do terceiro conjunto de experimentos definindo um intervalo de preferência, será possível resgatar todas as soluções a serem analisadas pelo tomador de decisão. Essa análise faz parte de um estudo posterior e pode ser indicada como trabalhos futuros desta tese.

Uma segunda análise foi proposta a fim de avaliar o efeito do uso das restrições de cardinalidade nos problemas analisados. As Figs. 88 a 93 apresentam os resultados comparativos das curvas EAF *best* de cada treliça para os casos I, II e III. Em cada gráfico

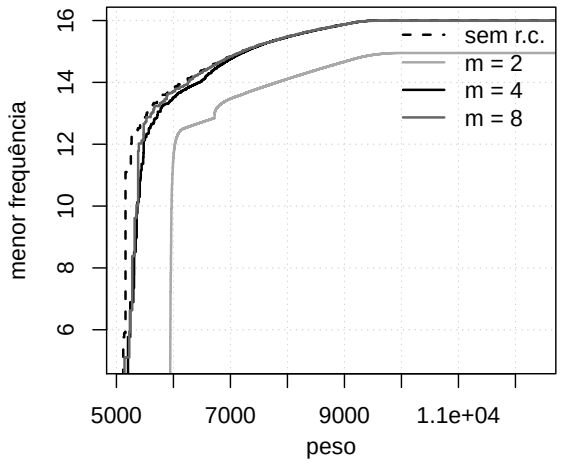
são apresentados também seus respectivos valores de hipervolume normalizados. A Tabela 20 apresenta um resumo da melhor configuração dos gráficos comparativos para cada problema baseado no seu valor de hipervolume.

Tabela 20 – Melhor configuração dos gráficos comparativos para as treliças de 10, 25, 72 e 200 barras nos casos I, II e III.

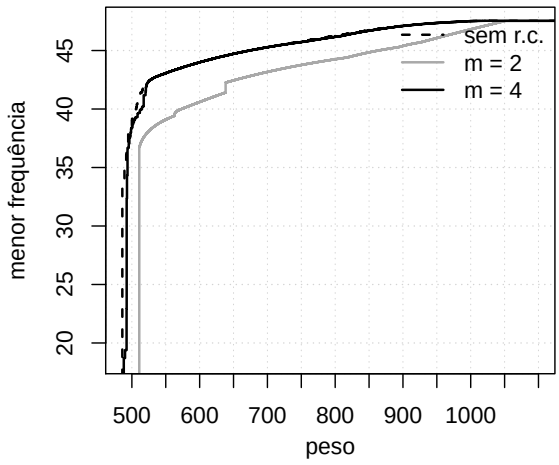
Problema	Caso I	Caso II	Caso III
Treliça de 10 barras (contínuo)	$m = 8$	sem r.c.	sem r.c.
Treliça de 10 barras (discreto)	$m = 2$	sem r.c.	sem r.c.
Treliça de 25 barras (contínuo)	sem r.c.	$m = 4$	$m = 4$
Treliça de 25 barras (discreto)	sem r.c.	$m = 4$	$m = 2$ e 4
Treliça de 72 barras (contínuo)	$m = 8$	$m = 4$	$m = 4$
Treliça de 72 barras (discreto)	$m = 2$, 4 e 8	$m = 8$	$m = 4$
Treliça de 200 barras (contínuo)	$m = 2$	$m = 2$ e 16	$m = 16$
Treliça de 200 barras (discreto)	$m = 16$	$m = 16$	$m = 16$

Para a treliça de 10 barras, o uso da restrição de cardinalidade não trouxe o benefício esperado. Para esse problema, é indicado não utilizar o agrupamento das barras, exceto para o caso I que apresentou melhores resultados para o uso de $m = 8$ e 2 (casos contínuo e discreto, respectivamente). Para o primeiro caso da treliça de 25 barras, o uso da restrição de cardinalidade não se mostrou satisfatório. Contudo, nos demais casos, melhores resultados foram encontrados quando $m = 4$. Em todos os casos da treliça de 72 barras, o uso da cardinalidade apontou melhores resultados comparado aos resultados sem restrição de cardinalidade. As cardinalidades 4 e 8 destacaram-se para este problema. Por fim, para a treliça de 200 barras, os melhores resultados encontrados foram para configurações com $m = 2$ e 16.

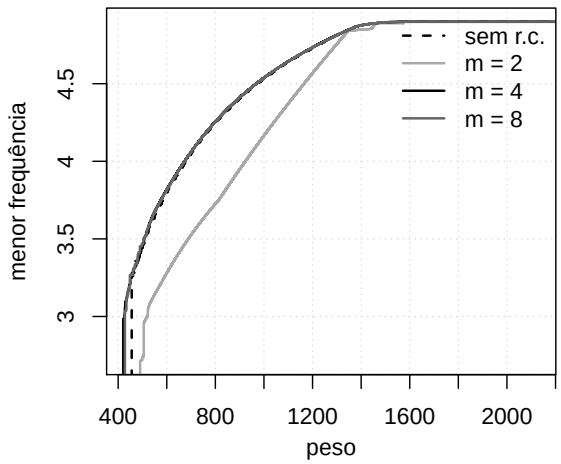
Em síntese, do ponto de vista da engenharia, o uso das restrições de cardinalidade nas treliças mostrou ser eficiente quando o número de barras é maior. Para as treliças de 25 e 72 barras, é indicado o uso de configurações com $m = 4$ e para a treliça de 200 barras, o agrupamento quando $m = 16$ é mais apropriado.



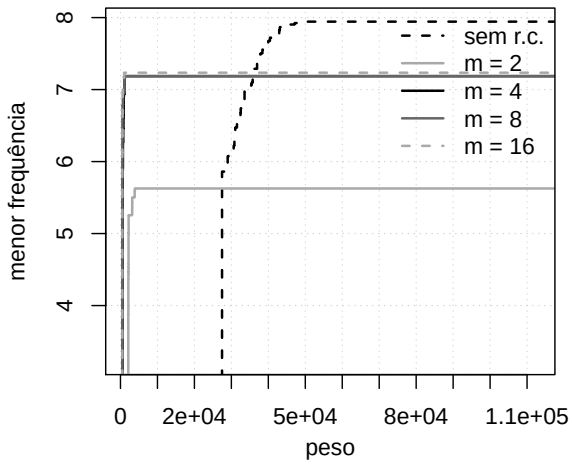
(a) $H_1 = 0.0177$, $H_2 = 0.0145$, $H_3 = 0.0161$ e $H_4 = 0.08294$



(b) $H_1 = 0.9516$, $H_2 = 0.9309$, e $H_3 = 0.9345$

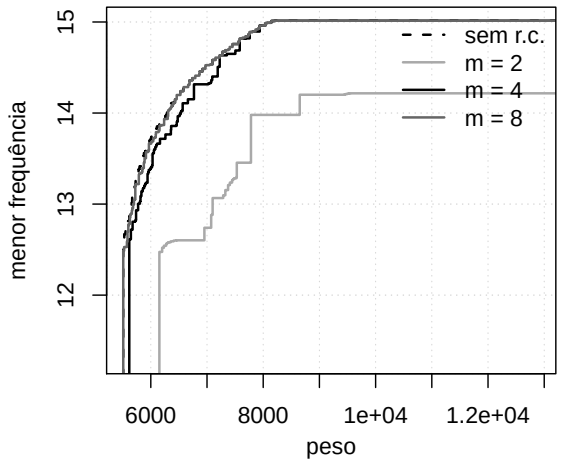


(c) $H_1 = 0.1566$, $H_2 = 0.3338$, $H_3 = 0.4207$ e $H_4 = 0.4806$

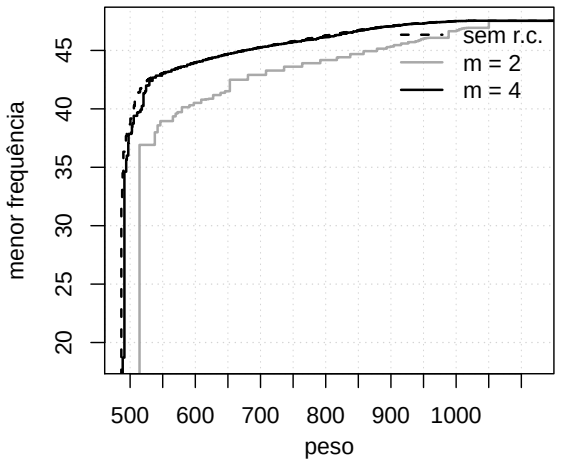


(d) $H_1 = 0.7517$, $H_2 = 0.9858$, $H_3 = 0.9792$, $H_4 = 0.9835$ e $H_5 = 0.8874$

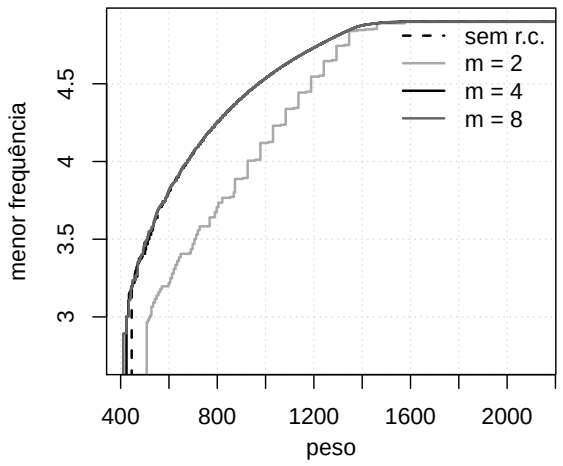
Figura 88 – Comparação entre as curvas EAF *best* com e sem restrição de cardinalidade caso contínuo para as treliças de 10, 25, 72 e 200 barras (figuras a, b, c, d, respectivamente). Valores de hipervolume normalizados: H_1 para sem r.c, H_2 para $m = 2$, H_3 para $m = 4$, H_4 para $m = 8$ e H_5 para $m = 16$, quando aplicáveis - caso 1.



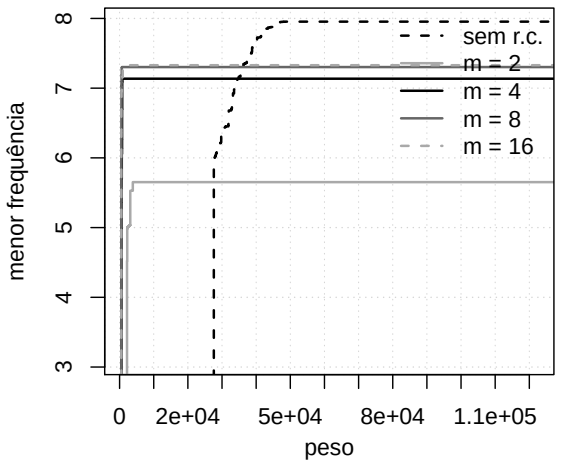
(a) $H_1 = 0.1845$, $H_2 = 0.4608$, $H_3 = 0.3886$ e $H_4 = 0.1971$



(b) $H_1 = 0.9992$, $H_2 = 0.9756$, e $H_3 = 0.9840$

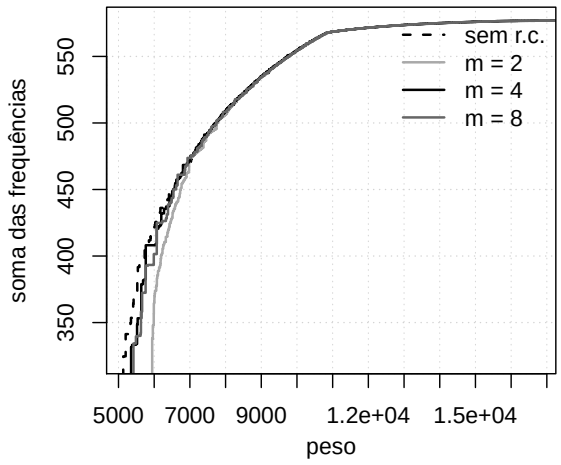


(c) $H_1 = 0.2351$, $H_2 = 0.5253$, $H_3 = 0.5253$ e $H_4 = 0.5253$

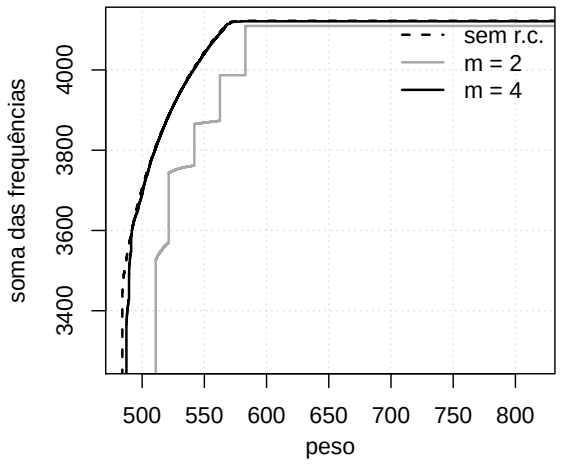


(d) $H_1 = 0.3688$, $H_2 = 0.4679$, $H_3 = 0.3688$, $H_4 = 0.4666$ e $H_5 = 0.4685$

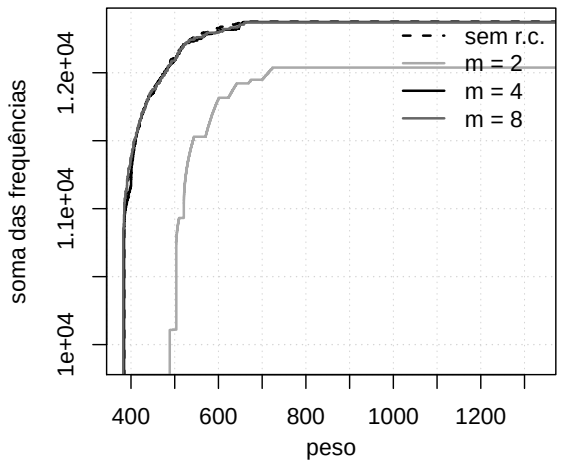
Figura 89 – Comparação entre as curvas EAF *best* com e sem restrição de cardinalidade caso discreto para as treliças de 10, 25, 72 e 200 barras (figuras a, b, c, d, respectivamente). Valores de hipervolume normalizados: H_1 para sem r.c, H_2 para $m = 2$, H_3 para $m = 4$, H_4 para $m = 8$ e H_5 para $m = 16$, quando aplicáveis - caso 1.



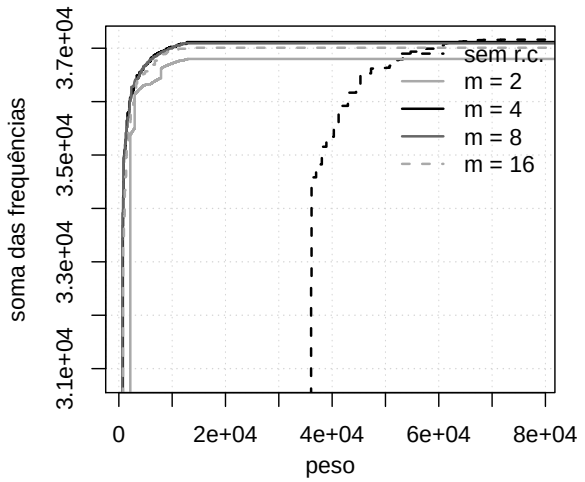
(a) $H_1 = 0.8050$, $H_2 = 0.5683$, $H_3 = 0.6028$ e $H_4 = 0.5683$



(b) $H_1 = 0.4429$, $H_2 = 0.6264$, e $H_3 = 0.6936$

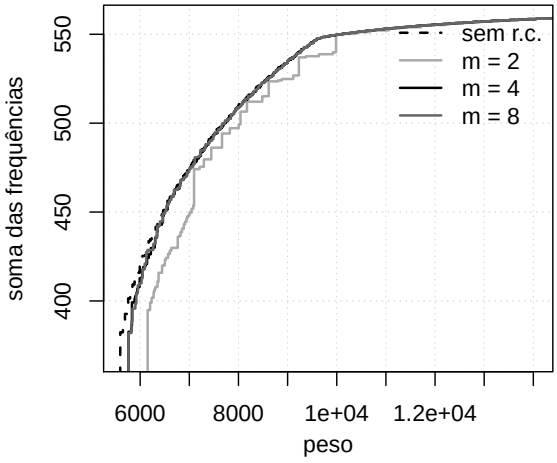


(c) $H_1 = 0.5353$, $H_2 = 0.5057$, $H_3 = 0.5681$ e $H_4 = 0.4658$

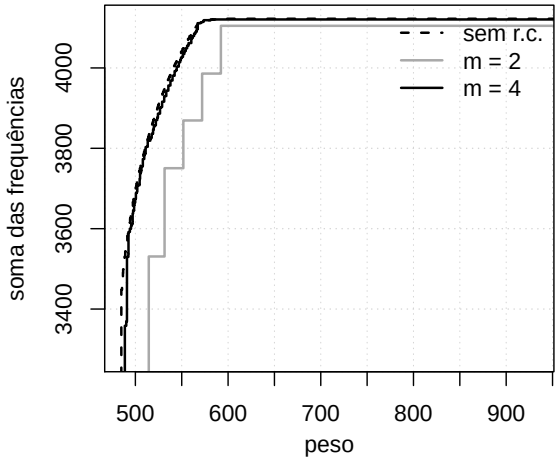


(d) $H_1 = 0.5565$, $H_2 = 0.8456$, $H_3 = 0.8179$, $H_4 = 0.8179$ e $H_5 = 0.8456$

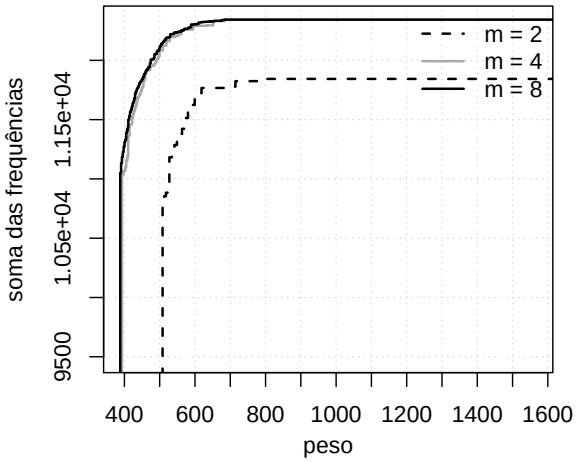
Figura 90 – Comparação entre as curvas EAF *best* com e sem restrição de cardinalidade caso contínuo para as treliças de 10, 25, 72 e 200 barras (figuras a, b, c, d, respectivamente). Valores de hipervolume normalizados: H_1 para sem r.c, H_2 para $m = 2$, H_3 para $m = 4$, H_4 para $m = 8$ e H_5 para $m = 16$, quando aplicáveis - caso 2.



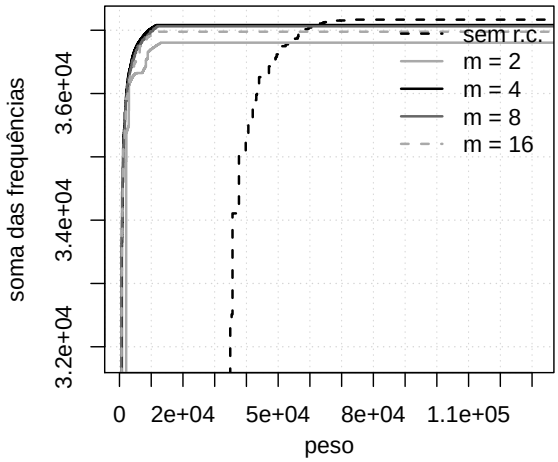
(a) $H_1 = 0.9049$, $H_2 = 0.4449$, $H_3 = 0.8028$ e $H_4 = 0.6661$



(b) $H_1 = 0.3543$, $H_2 = 0.7437$, e $H_3 = 0.9480$

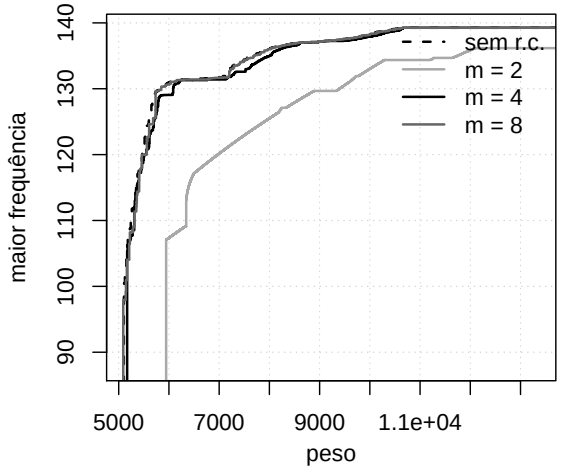


(c) $H_2 = 0.4523$, $H_3 = 0.4565$ e $H_4 = 0.5253$

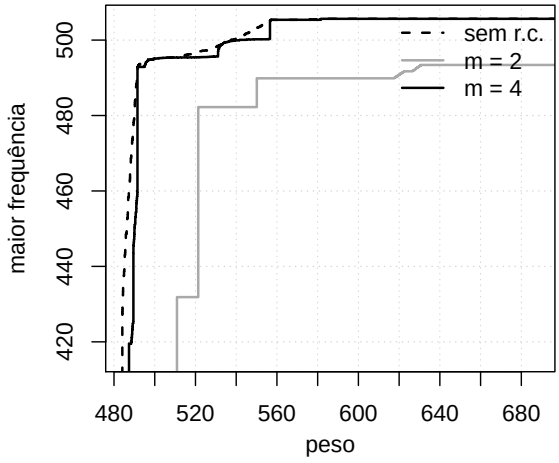


(d) $H_1 = 0.1056$, $H_2 = 0.7638$, $H_3 = 0.5757$, $H_4 = 0.7885$ e $H_5 = 0.7904$

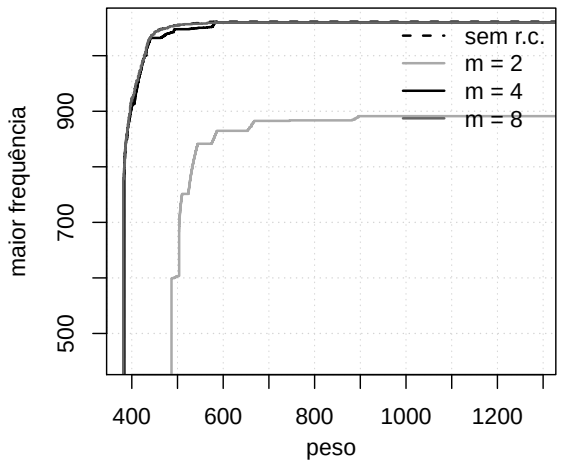
Figura 91 – Comparação entre as curvas EAF *best* com e sem restrição de cardinalidade caso discreto para as treliças de 10, 25, 72 e 200 barras (figuras a, b, c, d, respectivamente). Valores de hipervolume normalizados: H_1 para sem r.c, H_2 para $m = 2$, H_3 para $m = 4$, H_4 para $m = 8$ e H_5 para $m = 16$, quando aplicáveis - caso 2.



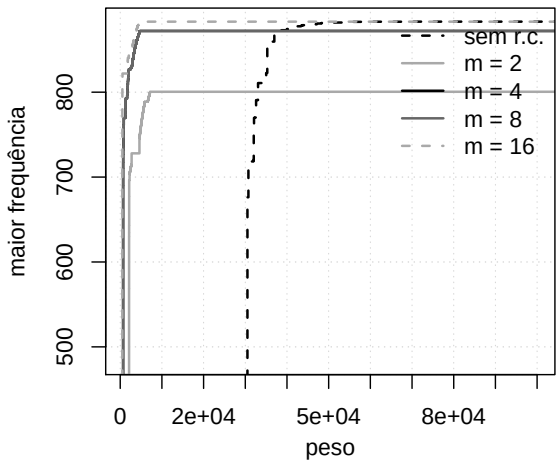
(a) $H_1 = 0.6967$, $H_2 = 0.1812$, $H_3 = 0.6618$ e $H_4 = 0.6556$



(b) $H_1 = 0.5991$, $H_2 = 0.7481$, e $H_3 = 0.8376$

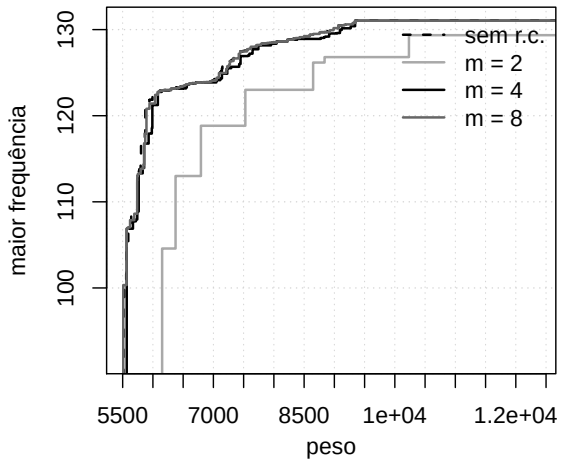


(c) $H_1 = 0.4000$, $H_2 = 0.4452$, $H_3 = 0.4912$ e $H_4 = 0.4608$

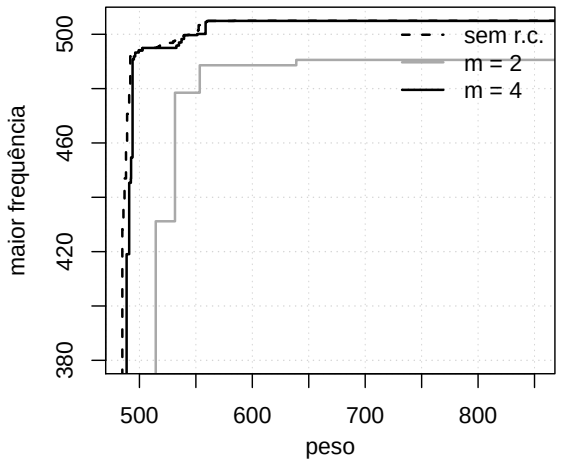


(d) $H_1 = 0.5607$, $H_2 = 0.9563$, $H_3 = 0.9859$, $H_4 = 0.8588$ e $H_5 = 0.9905$

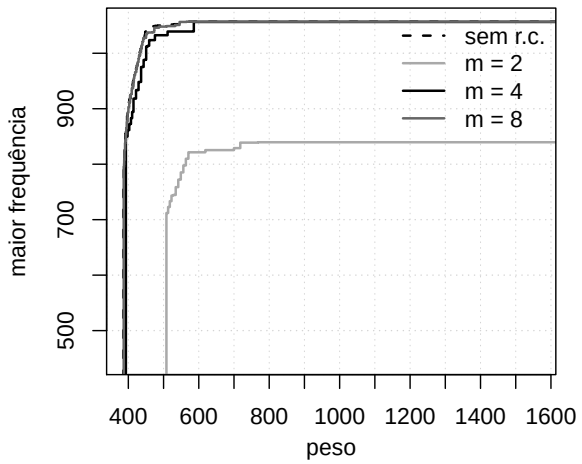
Figura 92 – Comparação entre as curvas EAF *best* com e sem restrição de cardinalidade caso contínuo para as treliças de 10, 25, 72 e 200 barras (figuras a, b, c, d, respectivamente). Valores de hipervolume normalizados: H_1 para sem r.c, H_2 para $m = 2$, H_3 para $m = 4$, H_4 para $m = 8$ e H_5 para $m = 16$, quando aplicáveis - caso 3.



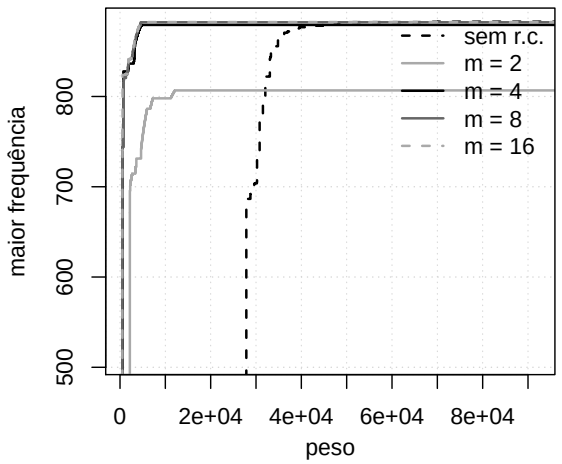
(a) $H_1 = 0.9467$, $H_2 = 0.6250$, $H_3 = 0.7147$ e $H_4 = 0.7916$



(b) $H_1 = 0.1680$, $H_2 = 0.2520$, e $H_3 = 0.2520$



(c) $H_1 = 0.3190$, $H_2 = 0.4383$, $H_3 = 0.4841$ e $H_4 = 0.4545$



(d) $H_1 = 0.0904$, $H_2 = 0.9471$, $H_3 = 0.7184$, $H_4 = 0.8134$ e $H_5 = 0.9875$

Figura 93 – Comparação entre as curvas EAF *best* com e sem restrição de cardinalidade caso discreto para as treliças de 10, 25, 72 e 200 barras (figuras a, b, c, d, respectivamente). Valores de hipervolume normalizados: H_1 para sem r.c, H_2 para $m = 2$, H_3 para $m = 4$, H_4 para $m = 8$ e H_5 para $m = 16$, quando aplicáveis - caso 3.

7 CONSIDERAÇÕES FINAIS

Na presente tese foi proposto um algoritmo de enxame de partículas multiobjetivo para a solução de problemas de otimização estrutural com restrições. O PSO padrão foi modificado através da introdução de um operador *craziness* no cálculo de sua velocidade, a fim de melhorar seu desempenho, denominado de CRPSO [25]. A versão multiobjetivo do CRPSO foi baseado no algoritmo desenvolvido em [5], e foi chamado aqui de MOCRPSO. As restrições presentes nos problemas foram tratadas através de um método de penalização, denominado APM, que tem um bom histórico de tratamento de restrições em problemas de otimização estrutural. Além disso, o MOCRPSO foi usado com uma codificação especial para as soluções candidatas considerando restrições de cardinalidade limitando o número máximo de áreas distintas da seção transversal para ser utilizado em uma solução otimizada.

7.1 CONCLUSÕES

Foram propostos três conjuntos de experimentos. Inicialmente, um primeiro conjunto contendo seis funções teste com dois e três objetivos foi utilizado para avaliar o desempenho do algoritmo proposto. Os resultados apresentados nesse primeiro conjunto foram comparados ao resultados obtidos pelo algoritmo NSGA-II, popular em estudos multiobjetivos. O MOCRPSO alcançou um desempenho superior em 4 das 6 funções teste analisadas, obtendo assim maiores valores de hipervolume normalizados.

Após a análise preliminar e os bons resultados obtidos pelo MOCRPSO, dois grupos de experimentos foram propostos envolvendo problemas de otimização estrutural amplamente discutidos na literatura. O segundo conjunto de experimentos refere-se à minimização do peso da estrutura e à minimização do máximo deslocamento, adicionando em sua formulação as restrições de cardinalidade. Para a análise, foram utilizadas as treliças de 10, 25, 60, 72 e 942 barras. Os resultados encontrados foram comparados com os resultados apresentados por dois pares de algoritmos multiobjetivo: GDE3 e GDE3-APM, para a comparação I e MOACS e MOAS, para a comparação II.

Para a comparação I, o MOCRPSO não obteve um desempenho superior comparado aos algoritmos GDE3 e GDE3-APM. Estes apresentaram maiores valores de hipervolumes que o MOCRPSO. Mesmo assim, o algoritmo proposto mostrou ser competitivo obtendo valores de hipervolumes próximos aos dos dois algoritmos. Na comparação II, onde as restrições de cardinalidade foram adicionadas na formulação do problema, os valores de hipervolumes apresentados pelo MOCRPSO foi competitivo comparados aos valores obtidos pelos algoritmos MOACS e MOAS. Uma característica observada, é que o MOCRPSO apresentou maiores valores de hipervolume em estruturas que possuem mais barras (treliças de 72 e 942 barras, por exemplo). Desta forma, os resultados obtidos no segundo conjunto

de experimentos comprovaram a eficiência e robustez do algoritmo proposto.

Para o terceiro conjunto de experimentos foram utilizadas as treliças de 10, 25, 72 e 200 barras e o problema envolve a minimização do peso da estrutura e a maximização das frequências naturais de vibração. O segundo objetivo foi dividido em 3 diferentes casos que abordam a maximização da menor frequência, a soma das frequências e a maior frequência, respectivamente. Esta proposta pode ser definida como um novo conjunto de problemas de otimização no que diz respeito à sua formulação, especificamente ao uso das frequências naturais de vibração como um dos objetivos, combinadas com as restrições de cardinalidade.

As curvas EAF para os casos I, II e III das 4 treliças analisadas com e sem restrições de cardinalidade foram apresentados no Capítulo 6. Uma avaliação envolvendo tais resultados foi apresentada e notou-se que, em alguns casos, foi observado um intervalo de preferência a ser investigado pelo tomador de decisão. Por conta disto, uma análise subsequente em cada uma das estruturas é indicada para estudos posteriores. E, ainda, além dos benefícios do uso das restrições de cardinalidade previamente discutidas nesta tese, os resultados e as análises apresentadas puderam reforçar tais benefícios, indicando melhores resultados para o peso e as frequências quando comparados aos resultados sem restrição de cardinalidade.

Finalmente, a análise exposta no Capítulo 6 envolvendo os resultados apresentados nos três conjuntos de experimentos permitiu concluir que o algoritmo MOCRPSO proposto apresentou bom desempenho, mostrando-se eficiente e robusto na resolução de problemas de otimização estrutural multiobjetivo. Entretanto, especificamente quando comparado a algoritmos multiobjetivos que possuem como base a Evolução Diferencial, cabe uma atenção e uma possível melhoria com o intuito de obter melhores resultados nessas comparações. Além disso, os experimentos envolvendo as frequências naturais de vibração da estrutura mostraram-se promissores e passíveis de futuras análises.

7.2 TRABALHOS FUTUROS

Baseado na proposta e análises apresentadas nesta tese, as sugestões para trabalhos futuros envolvem melhorias no algoritmo proposto, novos problemas e formulações para os problemas de otimização multiobjetivo. As sugestões são listadas a seguir:

- abordar problemas de otimização multiobjetivo considerando a possibilidade de mudanças de topologia da estrutura, ou seja, deixar ou retirar barras durante o processo evolutivo.
- utilizar informações de preferência do usuário *a priori* e interativa no mecanismo de busca em problemas de otimização multiobjetivo;

- uso de metamodelos para acelerar o processo de avaliação de uma solução candidata;
- realizar novos experimentos considerando o somatório dos deslocamentos como um dos objetivos;
- utilizar as restrições normalizadas referentes às tensões de flambagem máximas na formulação do problema multiobjetivo para o caso estático;
- realizar novos experimentos utilizando o somatório das frequências como um dos objetivos e, neste caso, por exemplo, somando-se um número menor de frequências de acordo com o interesse do projetista. As três primeiras frequências naturais, por exemplo.

REFERÊNCIAS

[1] *Tacoma Narrows Bridge*. Wikimedia, 2018.

[2] Takuya Kurihara and Kenya Jin'no. Analysis of convergence property of pso and its application to nonlinear blind source separation. In *2013 IEEE Congress on Evolutionary Computation (CEC)*, pages 976–981. IEEE, 2013.

[3] Regina M. Azuma. Otimização multiobjetivo em problema de estoque e roteamento gerenciados pelo fornecedor. Master's thesis, Programa de Pós-Graduação em Engenharia Elétrica, Universidade Estadual de Campinas, 2011.

[4] David E Golberg. Genetic algorithms in search, optimization, and machine learning. *Addion wesley*, 1989:102, 1989.

[5] Carlo R Raquel and Prospero C Naval Jr. An effective use of crowding distance in multiobjective particle swarm optimization. In *Proceedings of the 7th annual conference on Genetic and evolutionary computation*, pages 257–264. ACM, 2005.

[6] Dênis Emanuel da Costa Vargas. *Um algoritmo de evolução diferencial com penalização adaptativa para otimização estrutural multiobjetivo*. PhD thesis, Programa de Pós-graduação em Modelagem Computacional, Universidade Federal de Juiz de Fora, 2015.

[7] Manuel López-Ibánez, Luís Paquete, and Thomas Stützle. Exploratory analysis of stochastic local search algorithms in biobjective optimization. In *Experimental methods for the analysis of optimization algorithms*, pages 209–222. Springer, 2010.

[8] Lidiane Sartini de Oliveira and Sezimária FP Saramago. Multiobjective optimization techniques applied to engineering problems. *Journal of the brazilian society of mechanical sciences and engineering*, 32(1):94–105, 2010.

[9] H.M. Gomes. Truss optimization with dynamic constraints using a particle swarm algorithm. *Expert Systems with Applications*, 38(1):957–968, 2011.

[10] Miguel, L.F.F. e Miguel, L.F.F. Shape and size optimization of truss structures considering dynamic constraints through modern metaheuristic algorithms. *Expert Systems with Applications*, 39(10):9458 – 9467, 2012.

[11] Liu, X., Cheng, G., Yan, J. e Jiang, L. Singular optimum topology of skeletal structures with frequency constraints by AGGA. *Structural and Multidisciplinary Optimization*, 45(3):451–466, 2012.

[12] Carlos A Coello Coello. Constraint-handling using an evolutionary multiobjective optimization technique. *Civil Engineering Systems*, 17(4):319–346, 2000.

[13] José Elias Claudio Arroyo and Ana Amélia de Souza Pereira. A grasp heuristic for the multi-objective permutation flowshop scheduling problem. *The International Journal of Advanced Manufacturing Technology*, 55(5):741–753, 2011.

[14] J.S. Angelo, H.S. Bernardino, and H.J.C Barbosa. Ant colony approaches for multi-objective structural optimization problems with a cardinality constraint. *Advances in Engineering Software*, 80:101–115, 2015.

[15] DEC Vargas, ACC Lemonge, HJC Barbosa, and HS Bernardino. Um algoritmo baseado em evolução diferencial para problemas de otimização estrutural multiobjetivo com restrições. *Revista Internacional de Métodos Numéricos para Cálculo y Diseño en Ingeniería*, 32(2):91–99, 2016.

[16] Xin Li, Jingang Lai, and Ruoli Tang. A hybrid constraints handling strategy for multiconstrained multiobjective optimization problem of microgrid economical/environmental dispatch. *Complexity*, 2017, 2017.

[17] Guillermo Cabrera, Matthias Ehrgott, Andrew J Mason, Andrea Raith, et al. A matheuristic approach to solve the multiobjective beam angle optimization problem in intensity-modulated radiation therapy. *International Transactions in Operational Research*, 25(1):243–268, 2018.

[18] V Ho-Huu, T Nguyen-Thoi, T Truong-Khac, L Le-Anh, and T Vo-Duy. An improved differential evolution based on roulette wheel selection for shape and size optimization of truss structures with frequency constraints. *Neural computing and applications*, 29(1):167–185, 2018.

[19] Kalyanmoy Deb. *Multi-objective optimization using evolutionary algorithms*, volume 16. John Wiley & Sons, 2001.

[20] Agoston E Eiben, James E Smith, et al. *Introduction to evolutionary computing*, volume 53. Springer, 2003.

[21] Daniel Ashlock. *Evolutionary computation for modeling and optimization*. Springer Science & Business Media, 2006.

[22] David Corne, Marco Dorigo, Fred Glover, Dipankar Dasgupta, Pablo Moscato, Riccardo Poli, and Kenneth V Price. *New ideas in optimization*. McGraw-Hill Ltd., UK, 1999.

[23] Lawrence J Fogel. *Intelligence through simulated evolution: forty years of evolutionary programming*. John Wiley & Sons, Inc., 1999.

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

[25] Rajib Kar, Durbadal Mandal, Sangeeta Mondal, and Sakti P. Ghoshal. Craziness based particle swarm optimization algorithm for fir band stop filter design. *Swarm and Evolutionary Computation*, 2012.

[26] H.J.C. Barbosa and A.C.C. Lemonge. A genetic algorithm encoding for a class of cardinality constraints. In *Proceedings of the 7th annual GECCO*, pages 1193–1200. ACM, 2005.

[27] H.J.C. Barbosa and A.C.C. Lemonge. An adaptive penalty scheme in genetic algorithms for constrained optimiazation problems. In *GECCO*, volume 2, pages 287–294, 2002.

[28] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[29] Joseph B. Schwartz and Merck Sharp. Optimization techniques in product formulation. 32:287–301, 1981.

[30] Amir Hossein Gandomi, Xin-She Yang, Siamak Talatahari, and Amir Hossein Alavi. Metaheuristic algorithms in modeling and optimization. In *Metaheuristic applications in structures and infrastructures*, pages 1–24. Elsevier, 2013.

[31] Arzhang Alimoradi, Christopher M Foley, and Shahram Pezeshk. Benchmark problems in structural design and performance optimization: past, present and future–part i. In *19th ASCE Conf. Proc., State of the Art and Future Challenges in Structure. ASCE Publications*, 2010.

[32] Jian Hua Rong and Qing Quan Liang. A level set method for topology optimization of continuum structures with bounded design domains. *Computer Methods in Applied Mechanics and Engineering*, 197(17):1447–1465, 2008.

[33] M Shimoda, K Iwasa, and S Tsukada. Optimal shape design of shell structures. *Proceedings of IV European Conference on Computational Mechanics, ECCM 2010*, 2010.

[34] A.C.C. Lemonge, H.J.C. Barbosa, A.L.G.A. Coutinho, and C.C.H. Borges. Multiple cardinality constraints and automatic member grouping in the optimal design of steel framed structures. *Engineering Structures*, 33(2):433–444, 2011.

[35] Pedro Tiago de Freitas Mendes. *Análise dinâmica de uma estrutura: estudo numérico e experimental*. PhD thesis, 2012.

[36] P. Mendes. *Dinâmica de Estruturas*. Instituto Superior de Engenharia de Lisboa, 2012 (accessado em 20 de fevereiro de 2018). `http://pwp.net.ipl.pt/dec.isel/pmendes/publicacoes/Folhas_DE_versao_provisoria_08-06-2012.pdf`.

[37] Ray W Clough and Joseph Penzien. Dynamics of structures. 1993. *Applied Mechanics & Materials*, 1993.

[38] Ray W. Clough and Joseph Penzien. Dynamics of structures. Technical report, 1975.

[39] Tirupathi R. Chandrupatla, Ashok D. Belegundu, T. Ramesh, and Chaitali Ray. *Introduction to finite elements in engineering*. Prentice Hall Upper Saddle River, 1997.

[40] A Kaveh, L Jafari, and N Farhoudi. Truss optimization with natural frequency constraints using a dolphin echolocation algorithm. *Asian J Civil Eng*, 16:29–46, 2015.

[41] Shahin Jalili and Siamak Talatahari. Optimum design of truss structures under frequency constraints using hybrid css-mbls algorithm. *KSCE Journal of Civil Engineering*, 22(5):1840–1853, 2018.

[42] Guan-Chun Luh and Chung-Huei Chueh. Multi-objective optimal design of truss structure with immune algorithm. *Computers & structures*, 82(11):829–844, 2004.

[43] Ruiyi Su, Xu Wang, Liangjin Gui, and Zijie Fan. Multi-objective topology and sizing optimization of truss structures based on adaptive multi-island search strategy. *Structural and Multidisciplinary Optimization*, 43(2):275–286, 2011.

[44] A Kaveh and K Laknejadi. A hybrid multi-objective optimization and decision making procedure for optimal design of truss structures. *Iranian Journal of Science and Technology. Transactions of Civil Engineering*, 35(C2):137, 2011.

[45] JS Angelo, HS Bernardino, and HJC Barbosa. Multi-objective ant colony approaches for structural optimization problems. In *Proceedings of the eleventh international conference on computational structures technology, Civil-Comp Press, Stirlingshire (UK)*, 2012.

[46] Yan-fei Zhu and Xiong-min Tang. Overview of swarm intelligence. In *2010 International Conference on Computer Application and System Modeling (ICCASM)*, volume 9, pages V9–400. IEEE, 2010.

[47] Gerardo Beni. The concept of cellular robotic system. In *IEEE International Symposium on Intelligent Control, 1988. Proceedings*, pages 57–62. IEEE, 1988.

[48] Gerardo Beni and Jing Wang. Swarm intelligence in cellular robotic systems. In *Proceed. NATO Advanced Workshop on Robots and Biological Systems*. Springer, 1989.

[49] Alberto Colorni, Marco Dorigo, and Vittorio Maniezzo. Distributed optimization by ant colonies. In *Proceedings of the first European conference on artificial life*, volume 142, pages 134–142. Paris, France, 1991.

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

[51] Muzaffar M Eusuff and Kevin E Lansey. Optimization of water distribution network design using the shuffled frog leaping algorithm. *Journal of Water Resources planning and management*, 129(3):210–225, 2003.

[52] Kevin M Passino. Biomimicry of bacterial foraging for distributed optimization and control. *IEEE control systems*, 22(3):52–67, 2002.

[53] Konstantinos E. Parsopoulos and Michael N. Vrahatis. Recent approaches to global optimization problems through particle swarm optimization. *Natural computing*, 1(2-3):235–306, 2002.

[54] Saber M. Elsayed, Ruhul Sarker, and Efrén Mezura-Montes. Particle swarm optimizer for constrained optimization. In *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pages 2703–2711. IEEE, 2013.

[55] Mauro S. Innocente, Silvana M.B. Afonso, Johann Sienz, and Helen M. Davies. Particle swarm algorithm with adaptive constraint handling and integrated surrogate model for the management of petroleum fields. *Applied Soft Computing*, 2015.

[56] Amir H Gandomi and Ali R Kashani. Evolutionary bound constraint handling for particle swarm optimization. In *2016 4th International Symposium on Computational and Business Intelligence (ISCBI)*, pages 148–152. IEEE, 2016.

[57] Indah Soesanti and Ramadoni Syahputra. Batik production process optimization using particle swarm optimization method. *Journal of Theoretical and Applied Information Technology*, 86(2):272, 2016.

[58] Ran Cheng and Yaochu Jin. A social learning particle swarm optimization algorithm for scalable optimization. *Information Sciences*, 291:43–60, 2015.

[59] A. Kaveh and S. Talatahari. Particle swarm optimizer, ant colony strategy and harmony search scheme hybridized for optimization of truss structures. *Computers & Structures*, 87(5):267–283, 2009.

[60] A Kaveh and S Talatahari. Hybrid charged system search and particle swarm optimization for engineering design problems. *Engineering Computations*, 28(4):423–440, 2011.

[61] A. Kaveh and K. Laknejadi. A novel hybrid charge system search and particle swarm optimization method for multi-objective optimization. *Expert Systems with Applications*, 38(12):15475–15488, 2011.

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

[63] R.E. Perez and K. Behdinan. Particle swarm approach for structural design optimization. *Computers and Structures*, 85:1579–1588, 2007.

[64] Peter W. Jansen and Ruben E. Perez. Constrained structural design optimization via a parallel augmented lagrangian particle swarm optimization approach. *Computers & Structures*, 89(13):1352–1366, 2011.

[65] Guan-Chun Luh and Chun-Yi Lin. Optimal design of truss-structures using particle swarm optimization. *Computers & Structures*, 89(23):2221–2232, 2011.

[66] A Kaveh, R Sheikholeslami, S Talatahari, and M Keshvari-Ilkhichi. Chaotic swarming of particles: a new method for size optimization of truss structures. *Advances in Engineering Software*, 67:136–147, 2014.

[67] Ali Mortazavi and Vedat Toğan. Simultaneous size, shape, and topology optimization of truss structures using integrated particle swarm optimizer. *Structural and Multidisciplinary Optimization*, 54(4):715–736, 2016.

[68] Ali Mortazavi and Vedat Toğan. Sizing and layout design of truss structures under dynamic and static constraints with an integrated particle swarm optimization algorithm. *Applied Soft Computing*, 51:239–252, 2017.

[69] Zhao Xinchao. A perturbed particle swarm algorithm for numerical optimization. *Applied Soft Computing*, 10(1):119–124, 2010.

[70] Yuxin Zhao, Wei Zu, and Haitao Zeng. A modified particle swarm optimization via particle visual modeling analysis. *Computers & Mathematics with Applications*, 57(11):2022–2029, 2009.

[71] Yu Wang, Bin Li, Thomas Weise, Jianyu Wang, Bo Yuan, and Qiongjie Tian. Self-adaptive learning based particle swarm optimization. *Information Sciences*, 181(20):4515–4538, 2011.

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

[73] Matheus Rosendo. Um algoritmo de otimização por nuvem de partículas para resolução de problemas combinatórios. Master's thesis, Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, Universidade Federal do Paraná, 2010.

[74] James Kennedy. Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance. In *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, volume 3. IEEE, 1999.

[75] Carmelo J.A. Bastos-Filho, Marcel P. Caraciolo, Péricles B.C. Miranda, and Danilo F. Carvalho. Multi-ring particle swarm optimization. In *Neural Networks, 2008. SBRN'08. 10th Brazilian Symposium on*, pages 111–116. IEEE, 2008.

[76] Rafal Kicinger, Tomasz Arciszewski, and Kenneth De Jong. Evolutionary computation and structural design: A survey of the state-of-the-art. *Computers & Structures*, 83(23):1943–1978, 2005.

[77] Carlos A. Coello Coello. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, 191(11-12):1245 – 1287, 2002.

[78] Mezura-Montes Efren. Constraint-handling in evolutionary optimization. *Pland: Springer*, 2009.

[79] Carlos A. Coello Coello and A. Carlos. A survey of constraint handling techniques used with evolutionary algorithms. *Lania-RI-99-04, Laboratorio Nacional de Informática Avanzada*, 1999.

[80] Richard Courant. Variational methods for the solution of problems of equilibrium and vibrations. *Bull. Amer. Math. Soc*, 49(1):23, 1943.

[81] Charles W Carroll. The created response surface technique for optimizing nonlinear, restrained systems. *Operations Research*, 9(2):169–184, 1961.

[82] Anthony V Fiacco and Garth P McCormick. Extensions of sumt for nonlinear programming: equality constraints and extrapolation. *Management Science*, 12(11):816–828, 1966.

[83] Jon T. Richardson, Mark R. Palmer, Gunar E. Liepins, and Mike Hilliard. Some guidelines for genetic algorithms with penalty functions. In *Proceedings of the third international conference on Genetic algorithms*, pages 191–197. Morgan Kaufmann Publishers Inc., 1989.

[84] Efrén Mezura-Montes and Carlos A. Coello Coello. Constraint-handling in nature-inspired numerical optimization: past, present and future. *Swarm and Evolutionary Computation*, 1(4):173–194, 2011.

[85] H.J.C. Barbosa, A.C.C. Lemonge, and H.S. Bernardino. A critical review of adaptive penalty techniques in evolutionary computation. In *Evolutionary Constrained Optimization*, pages 1–27. Springer, 2015.

[86] H.J.C. Barbosa and A.C.C. Lemonge. An adaptive penalty scheme in genetic algorithms for constrained optimiazation problems. In *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference*, pages 287–294, New York, 9-13 July 2002. Morgan Kaufmann Publishers.

[87] H.J.C. Barbosa, A.C.C. Lemonge, and C.C.H. Borges. A genetic algorithm encoding for cardinality constraints and automatic variable linking in structural optimization. *Engineering Structures*, 30:3708–3723, 2008.

[88] Barbosa, H.J.C., Lemonge, A.C.C. An adaptive penalty scheme in genetic algorithms for constrained optimization problems. *GECCO 2002: proceedings of the genetic and evolutionary computation conference. New York: Morgan Kaufmann Publishers*, pages 287–294, 2002.

[89] A.C.C. Lemonge, H.J.C. Barbosa, A.L.G.A. Coutinho, and C.C.H. Borges. Multiple cardinality constraints and automatic member grouping in the optimal design of steel framed structures. *Engineering Structures*, 33:433–444, 2011.

[90] Lingyun Wei, Tianbing Tang, Xianghong Xie, and Wenjie Shen. Truss optimization on shape and sizing with frequency constraints based on parallel genetic algorithm. *Structural and Multidisciplinary Optimization*, 43(5):665–682, 2011.

[91] Xiaofeng Liu, Gengdong Cheng, Jun Yan, and Lei Jiang. Singular optimum topology of skeletal structures with frequency constraints by AGGA. *Structural and Multidisciplinary Optimization*, 45(3):451–466, 2012.

[92] M. Kripka, G.F. Medeiros, and A.C.C. Lemonge. Structural optimization of reinforced concrete building grillages considering cardinality constraints. In *10th World congress on structural and multidisciplinary optimization*, pages 01–06, 2013.

[93] A. Kaveh and A. Zolghadr. Democratic PSO for truss layout and size optimization with frequency constraints. *Computers & Structures*, 130:10–21, 2014.

[94] A. Kaveh and S.M. Javadi. Shape and size optimization of trusses with multiple frequency constraints using harmony search and ray optimizer for enhancing the particle swarm optimization algorithm. *Acta Mechanica*, 225(6):1595–1605, 2014.

[95] Ghanshyam G. Tejani, Vimal J. Savsani, and Vivek K. Patel. Modified sub-population teaching-learning-based optimization for design of truss structures with natural frequency constraints. *Mechanics Based Design of Structures and Machines*, (just-accepted), 2016.

[96] M. Farshchin, C.V. Camp, and M. Maniat. Multi-class teaching–learning-based optimization for truss design with frequency constraints. *Engineering Structures*, 106:355–369, 2016.

[97] Ghanshyam G. Tejani, Vimal J Savsani, and Vivek K. Patel. Adaptive symbiotic organisms search (SOS) algorithm for structural design optimization. *Journal of Computational Design and Engineering*, 2016.

[98] Can B Kalayci, Okkes Ertenlice, Hasan Akyer, and Hakan Aygoren. An artificial bee colony algorithm with feasibility enforcement and infeasibility toleration procedures for cardinality constrained portfolio optimization. *Expert Systems with Applications*, 85:61–75, 2017.

[99] A Kaveh and A Zolghadr. Truss shape and size optimization with frequency constraints using tug of war optimization. 2017.

[100] José Elias Claudio Arroyo. Heurísticas e metaheurísticas para otimização combinatória multiobjetivo. 2002.

[101] Carlos A. Coello Coello. A comprehensive survey of evolutionary-based multiobjective optimization techniques. *Knowledge and Information systems*, 1(3):269–308, 1999.

[102] Carlos A. Coello Coello. Evolutionary multi-objective optimization: a historical view of the field. *Computational Intelligence Magazine, IEEE*, 1(1):28–36, 2006.

[103] Lin Feng, Zhizhong Mao, Ping Yuan, and Bi Zhang. Multi-objective particle swarm optimization with preference information and its application in electric arc furnace steelmaking process. *Structural and Multidisciplinary Optimization*, 52(5):1013–1022, 2015.

[104] Licheng Jiao, Juanjuan Luo, Ronghua Shang, and Fang Liu. A modified objective function method with feasible-guiding strategy to solve constrained multi-objective optimization problems. *Applied Soft Computing*, 14:363–380, 2014.

[105] V. Pareto. Cours d'économie politique: vol. i and ii,. *CDE Politique - Rouge, Lausanne*, pages 1776–1960, 1896.

[106] Aimin Zhou, Bo-Yang Qu, Hui Li, Shi-Zheng Zhao, Ponnuthurai Nagaratnam Suganthan, and Qingfu Zhang. Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation*, 1(1):32–49, 2011.

[107] Haifeng Ling, Yihong Xiao, Xianzhong Zhou, and Xunlin Jiang. An improved pso algorithm for constrained multiobjective optimization problems. In *2011 International Conference on Computer Science and Service System (CSSS)*, pages 3859–3863, June 2011.

[108] Hemlata S. Urade and Rahila Patel. Dynamic particle swarm optimization to solve multi-objective optimization problem. *Procedia Technology*, 6:283–290, 2012.

[109] N.C. Sahoo, S. Ganguly, and D. Das. Multi-objective planning of electrical distribution systems incorporating sectionalizing switches and tie-lines using particle swarm optimization. *Swarm and Evolutionary Computation*, 3:15–32, 2012.

[110] K.R. Harrison, A.P. Engelbrecht, and B.M. Ombuki-Berman. A scalability study of multi-objective particle swarm optimizers. In *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pages 189–197, June 2013.

[111] J.D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Genetic Algorithms and their Applications: Proc. of the First Int. Conf. on Genetic Algorithms*, pages 93–100. Lawrence Erlbaum, 1985.

[112] M Rabiee, M Zandieh, and P Ramezani. Bi-objective partial flexible job shop scheduling problem: Nsga-ii, nrga, moga and paes approaches. *International Journal of Production Research*, 50(24):7327–7342, 2012.

[113] Abolfazl Khalkhali, Sharif Khakshournia, and Nader Nariman-Zadeh. A hybrid method of fem, modified nsgaii and topsis for structural optimization of sandwich panels with corrugated core. *Journal of Sandwich Structures and Materials*, page 1099636214531516, 2014.

[114] Seyed Hamid Reza Pasandideh, Seyed Taghi Akhavan Niaki, and Kobra Asadi. Bi-objective optimization of a multi-product multi-period three-echelon supply chain problem under uncertain environments: NSGA-II and NRGA. *Information Sciences*, 292:57–74, 2015.

[115] Rajesh Arora, SC Kaushik, Raj Kumar, and Ranjana Arora. Multi-objective thermo-economic optimization of solar parabolic dish stirling heat engine with regenerative losses using nsga-ii and decision making. *International Journal of Electrical Power & Energy Systems*, 74:25–35, 2016.

[116] Feng Xue. *Multi-objective differential evolution: theory and applications*. PhD thesis, Rensselaer Polytechnic Institute, 2004.

[117] Eckart Zitzler and Lothar Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, 1999.

[118] Tea Robič and Bogdan Filipič. Demo: Differential evolution for multiobjective optimization. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 520–533. Springer, 2005.

[119] Saku Kukkonen and Jouni Lampinen. GDE3: The third evolution step of generalized differential evolution. In *2005 IEEE Congress on Evolutionary Computation*, volume 1, pages 443–450. IEEE, 2005.

[120] Jouni Lampinen. De's selection rule for multiobjective optimization. *Lappeenranta University of Technology, Department of Information Technology, Tech. Rep*, pages 03–04, 2001.

[121] Saku Kukkonen and Jouni Lampinen. An extension of generalized differential evolution for multi-objective optimization with constraints. In *International Conference on Parallel Problem Solving from Nature*, pages 752–761. Springer, 2004.

[122] Dênis E.C. Vargas, Afonso C.C. Lemonge, Helio J.C. Barbosa, and Heder S. Bernardino. Análise do desempenho de algoritmos baseados em evolução diferencial acoplados a uma técnica de penalização adaptativa em problemas de otimização estrutural multiobjetivo com restrições. *XI Simpósio de Mecânica Computacional - SIMMEC e II Encontro Mineiro de Modelagem Computacional - EMMCOMP*, 2014.

[123] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. Spea2: Improving the strength pareto evolutionary algorithm. *TIK-report*, 103, 2001.

[124] AM Adham, N Mohd-Ghazali, and R Ahmad. Performance optimization of a microchannel heat sink using the improved strength pareto evolutionary algorithm (spea2). *Journal of engineering thermophysics*, 24(1):86–100, 2015.

[125] Bhargav Gadhvi, Vimal Savsani, and Vivek Patel. Multi-objective optimization of vehicle passive suspension system using nsga-ii, spea2 and pesa-ii. *Procedia Technology*, 23:361–368, 2016.

[126] Fei He, Kang Shen, Li Guan, and Mingming Jiang. Research on energy-saving scheduling of a forging stock charging furnace based on an improved spea2 algorithm. *Sustainability*, 9(11):2154, 2017.

[127] Rachid Ellaia, Abderrahmane Habbal, and Emmanuel Pagnacco. A new accelerated multi-objective particle swarm algorithm. applications to truss topology optimization. In *10th World Congress on Structural and Multidisciplinary Optimization*, 2013.

[128] Hamid R Tizhoosh. Opposition-based reinforcement learning. *JACIII*, 10(4):578–585, 2006.

[129] CA Coello Coello and N Cruz Cortés. An approach to solve multiobjective optimization problems based on an artificial immune system. In *first international Conference on artificial immune systems (ICARIS'2002)*, pages 212–221, 2002.

[130] Carlos M Fonseca, Luís Paquete, and Manuel López-Ibánez. An improved dimension-sweep algorithm for the hypervolume indicator. In *IEEE Congress on Evolutionary Computation, 2006 - CEC 2006*, pages 1157–1163. IEEE, 2006.

[131] Carlos M Fonseca and Peter J Fleming. On the performance assessment and comparison of stochastic multiobjective optimizers. In *International Conference on Parallel Problem Solving from Nature*, pages 584–593. Springer, 1996.

[132] Carlos A Coello Coello, Gary B Lamont, David A Van Veldhuizen, et al. *Evolutionary algorithms for solving multi-objective problems*, volume 5. Springer, 2007.

[133] Elizabeth D. Dolan and Jorge J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91:201–213, 2002.

[134] Francilene Barbosa Santos Silva. Algoritmos genéticos para otimização de estruturas reticuladas baseadas em modelos adaptativos e lagrangeano aumentado. Master's thesis, Programa de Pós-Graduação em Modelagem Computacional, Universidade Federal de Juiz de Fora, 2011.

[135] Helio J. C. Barbosa, Heder S. Bernardino, and André M. S. Barreto. Using performance profiles to analyze the results of the 2006 cec constrained optimization competition. In *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pages 1–8. IEEE, 2010.

[136] André Barreto, Heder S Bernardino, and Helio JC Barbosa. Probabilistic performance profiles for the experimental evaluation of stochastic algorithms. In *Proceedings of the 12th annual GECCO*, pages 751–758. ACM, 2010.

[137] Heder S. Bernardino, Helio J. C. Barbosa, and Leonardo G. Fonseca. Surrogate-assisted clonal selection algorithms for expensive optimization problems. *Evolutionary Intelligence*, 4(2):81–97, 2011.

[138] A.H. Gandomi, X-S Yang, S. Talatahari, and A.H. Alavi. Firefly algorithm with chaos. *Communications in Nonlinear Science and Numerical Simulation*, 18(1):89–98, 2013.

[139] DT Pham, Afshin Ghanbarzadeh, Ebubekir Koc, Sameh Otri, S Rahim, and Muhamed Zaidi. The bees algorithm. technical note. *Manufacturing Engineering Centre, Cardiff University, UK*, pages 1–57, 2005.

[140] Xin-She Yang and Suash Deb. Multiobjective cuckoo search for design optimization. *Computers & Operations Research*, 40(6):1616–1624, 2013.

[141] Gustavo R. Zavala, Antonio J. Nebro, Francisco Luna, and Carlos A. Coello Coello. A survey of multi-objective metaheuristics applied to structural optimization. *Structural and Multidisciplinary Optimization*, 49(4):537–558, 2014.

[142] V. Yepes, T. García-Segura, and J.M. Moreno-Jiménez. A cognitive approach for the multi-objective optimization of rc structural problems. *Archives of Civil and Mechanical Engineering*, 15(4):1024–1036, 2015.

[143] Shouyong Jiang and Shengxiang Yang. An improved multiobjective optimization evolutionary algorithm based on decomposition for complex pareto fronts. *IEEE Transactions on Cybernetics*, 46(2):421–437, 2016.

[144] K Muralitharan, Rathinasamy Sakthivel, and Yan Shi. Multiobjective optimization technique for demand side management with load balancing approach in smart grid. *Neurocomputing*, 177:110–119, 2016.

[145] Ming-Der Yang, Min-Der Lin, Yu-Hao Lin, and Kang-Ting Tsai. Multiobjective optimization design of green building envelope material using a non-dominated sorting genetic algorithm. *Applied Thermal Engineering*, 111:1255–1264, 2017.

[146] Sanaz Mostaghim and Jürgen Teich. Strategies for finding good local guides in multi-objective particle swarm optimization (MOPSO). In *Proceedings of the 2003 IEEE Swarm Intelligence Symposium, 2003. SIS'03*, pages 26–33. IEEE, 2003.

[147] Carlos A. Coello Coello, Gregorio T. Pulido, and M. Salazar Lechuga. Handling multiple objectives with particle swarm optimization. *Evolutionary Computation, IEEE Transactions on*, 8(3):256–279, 2004.

[148] Julio E. Alvarez-Benitez, Richard M. Everson, and Jonathan E Fieldsend. A mopso algorithm based exclusively on pareto dominance concepts. In *Evolutionary Multi-Criterion Optimization*, pages 459–473. Springer, 2005.

[149] Margarita Reyes-Sierra and Carlos A. Coello Coello. Multi-objective particle swarm optimizers: A survey of the state-of-the-art. *International journal of computational intelligence research*, 2(3):287–308, 2006.

[150] Seyed Saeed Hosseini, Sajad Ahmad Hamidi, Motahar Mansuri, and Ali Ghoddosian. Multi objective particle swarm optimization (mopso) for size and shape optimization of 2d truss structures. *Periodica Polytechnica. Civil Engineering*, 59(1):9, 2015.

[151] Wang Hu and Gary G Yen. Adaptive multiobjective particle swarm optimization based on parallel cell coordinate system. *IEEE Transactions on Evolutionary Computation*, 19(1):1–18, 2015.

[152] Kanchana Sethanan and Woraya Neungmatcha. Multi-objective particle swarm optimization for mechanical harvester route planning of sugarcane field operations. *European Journal of Operational Research*, 252(3):969–984, 2016.

[153] Farshad Rezaei, Hamid R Safavi, Ali Mirchi, and Kaveh Madani. f-mopso: An alternative multi-objective pso algorithm for conjunctive water use management. *Journal of Hydro-environment Research*, 14:1–18, 2017.

[154] Honggui Han, Wei Lu, and Junfei Qiao. An adaptive multiobjective particle swarm optimization based on multiple adaptive methods. *IEEE Transactions on cybernetics*, 47(9):2754–2767, 2017.

[155] Yong Zhang, Dun-wei Gong, and Jian Cheng. Multi-objective particle swarm optimization approach for cost-based feature selection in classification. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 14(1):64–75, 2017.

[156] Anqi Pan, Lei Wang, Weian Guo, and Qidi Wu. A diversity enhanced multiobjective particle swarm optimization. *Information Sciences*, 436:441–465, 2018.

[157] A. A. Martins and A. A. Oluyinka. An adaptive velocity particle swarm optimization for high-dimensional function optimization. In *2013 IEEE Congress on Evolutionary Computation (CEC)*, pages 2352–2359. IEEE, 2013.

[158] Michael Pluhacek, Roman Senkerik, and Ivan Zelinka. Investigation on the performance of a new multiple choice strategy for pso algorithm in the task of large scale optimization problems. In *2013 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2013.

[159] José PG Carvalho, Afonso CC Lemonge, Érica CR Carvalho, Patrícia H Hallak, and Heder S Bernardino. Truss optimization with multiple frequency constraints and automatic member grouping. *Structural and Multidisciplinary Optimization*, 57(2):547–577, 2018.

[160] H.J.C. Barbosa and A.C.C. Lemonge. A genetic algorithm encoding for a class of cardinality constraints. In H.-G. Beyer et al, editor, *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-2005*, pages 1193–1200, New York, 2005. ACM Press.

[161] T.J.R. Hughes. *The finite element method: linear static and dynamic finite element analysis*. Courier Corporation, 2012.

[162] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary computation*, 8(2):173–195, 2000.

[163] Carlos M Fonseca, Andreia P Guerreiro, Manuel López-Ibáñez, and Luís Paquete. On the computation of the empirical attainment function. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 106–120. Springer, 2011.

[164] Eduardo Krempser, Douglas A Augusto, and Helio JC Barbosa. Improved surrogate model assisted differential evolution with an infill criterion. In *10th World Congress on Structural and Multidisciplinary Optimization, Orlando, Florida, USA*, 2013.

[165] João Gabriel Rocha Silva, Iago Augusto Carvalho, Michelli Marlane Silva Loureiro, Vinícus da Fonseca Vieira, and Carolina Ribeiro Xavier. Developing tasty calorie restricted diets using a differential evolution algorithm. In *International Conference on Computational Science and Its Applications*, pages 171–186. Springer, 2016.

[166] Kalyanmoy Deb and J Sundar. Reference point based multi-objective optimization using evolutionary algorithms. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 635–642. ACM, 2006.