OLACIR RODRIGUES CASTRO JUNIOR

ALGORITMOS DE NUVEM DE PARTÍCULAS E A OTIMIZAÇÃO COM MUITOS OBJETIVOS

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre. Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, Universidade Federal do Paraná.
Orientadora: Profa. Dra. Aurora Trinidad Ramirez Pozo

CURITIBA

2013

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Problemas de otimização multiobjetivo (MOPs) são problemas que possuem mais de uma função objetivo a ser minimizada ou maximizada. Entre as abordagens mais utilizadas atualmente para resolvê-los destaca-se o uso de metaheurísticas populacionais. Esta popularidade se deve principalmente à natureza destas de lidar simultaneamente com diversas soluções (população) em uma única execução.

Um algoritmo muito utilizado para lidar com MOPs é chamado otimização por nuvem de partículas multiobjetivo (MOPSO), esta é uma abordagem derivada da otimização por nuvem de partículas (PSO), que é uma metaheurística inspirada no comportamento de conjuntos de aves.

Devido ao bom desempenho apresentado pelos MOPSOs ao resolver MOPs, esta abordagem vem sendo estendida para a resolução de problemas de otimização com muitos objetivos (MaOPs). Estes problemas são caracterizados por apresentarem mais de três funções objetivo e uma alta complexidade causada principalmente porque a proporção de soluções não dominadas em uma população aumenta rapidamente com o número de objetivos, o que diminui a pressão de seleção em direção à fronteira de Pareto. Além disso, o número de pontos necessários para representar a fronteira aumenta exponencialmente de acordo com o número de objetivos dificultando a obtenção de soluções diversas o suficiente para cobri-la totalmente.

Este trabalho apresenta algumas técnicas aplicadas para melhorar o desempenho do MOPSO ao resolver MaOPs e torná-lo menos sensível ao aumento no número de objetivos. Primeiramente estudaram-se duas técnicas de controle da área de dominância das soluções para aumentar a pressão de seleção, normalmente reduzida pelo aumento no número de objetivos.

Outra técnica estudada foi a alteração do método de seleção de líderes do MOPSO com a realização de um estudo empírico usando seis métodos e os melhores foram destacados. Foi estudada também a influência sofrida por esses métodos devido à alteração na técnica

de controle da área de dominância, e as melhores combinações foram identificadas através de estudos empíricos.

Por último um novo MOPSO é proposto usando o conceito de pontos de referência distribuindo melhor as soluções obtidas e com isso melhorando a convergência à fronteira real. Estudos empíricos também foram realizados para comparar a nova abordagem à abordagem clássica.

A partir dos trabalhos realizados aqui três artigos foram publicados, sendo o primeiro um estudo sobre os métodos de seleção de líderes, o segundo propondo um novo MOPSO que usa uma técnica de controle da área de dominância, e o terceiro que avalia a influência das técnicas de controle da área de dominância no desempenho dos métodos de seleção de líder e identifica as melhores combinações entre técnica de controle da área de dominância e método de seleção de líder.

Em geral todos os estudos realizados apresentaram melhorias de desempenho em relação ao algoritmo original utilizado, especialmente no contexto de muitos objetivos.

ABSTRACT

Multiobjective optimization problems (MOPs) are problems that have more than one objective function to be minimized or maximized. Among the approaches currently used to solve them we highlight the use of populational metaheuristics. This popularity is mainly due to its nature of dealing simultaneously with many solutions (population) in a single run.

An algorithm frequently used to deal with MOPs is called multiobjective particle swarm optimization (MOPSO), this approach is derived from the particle swarm optimization (PSO), which is a metaheuristic inspired by the behavior of bird flocks.

Due to the good performance presented by MOPSOs to solve MOPs, this approach has been extended for solving many objective optimization problems (MaOPs). These problems are characterized by presenting more than three objective functions and a high complexity caused mainly because the proportion of non-dominated solutions in a population increases rapidly with the number of objectives, which reduces the selection pressure toward the Pareto frontier, moreover, the number of points to represent the frontier increases exponentially with the number of objectives becoming harder to obtain solutions diverse enough to cover it entirely.

This work presents some techniques applied to improve the performance of MOPSO to solve MaOPs and make it less sensitive to the increase in the number of objectives. Firstly were studied the use of two techniques to control the dominance area of solutions to increase the selection pressure, typically reduced by the increase in the number of objectives.

The impact of the leader selection method was also studied. Six methods were investigated experimentally and the best were highlighted. It was also studied the influence suffered by these methods due to the change in the control of the domination area of solutions, and the best combinations were identified through empirical studies.

Finally a new MOPSO is proposed using the concept of reference points to better

spread the solutions and thereby improve the convergence to the true frontier. Empirical studies were also conducted to compare the new approach with the classical one.

From the work done here three articles were published, the first being a study on methods of leader selection, the second proposes a new MOPSO that uses a method to control the dominance area of solutions, and the third that assesses the influence of techniques to control the dominance area of solutions in the performance of leader selection methods and identifies the best combination between the technique to control the dominance area of solutions and the leader selection method.

In general all studies showed performance improvements compared to the original algorithm used, especially in the context of many objectives.

CAPÍTULO 1

INTRODUÇÃO

1.1 Introdução

Vários problemas reais envolvem a otimização simultânea de dois ou mais objetivos, que estão usualmente em conflito [42, 6]. Estes problemas são chamados de problemas de otimização multiobjetivo (*Multiobjective Optimization Problems (MOPs)*). Na otimização mono-objetivo, a solução ótima de um dado conjunto é claramente determinada, entretanto na otimização multiobjetivo com a ausência de informações de preferência não há uma maneira única ou direta de determinar se uma solução é melhor que outra. Comumente se adota o conceito de dominância de Pareto, usando esta relação normalmente não é possível encontrar uma única solução ótima, mas um conjunto destas, que representam diferentes relações entre os objetivos. Este conjunto é chamado de conjunto ótimo de Pareto, e sua imagem no espaço de objetivos é chamada de fronteira de Pareto [27]. Normalmente ao otimizar um MOP, entre as diversas soluções encontradas preferem-se as que apresentem um bom compromisso entre os objetivos [9].

Apesar de terem sido desenvolvidas diversas técnicas na Pesquisa Operacional e outras disciplinas, a complexidade de se otimizar MOPs ainda desafia os pesquisadores. Entre as abordagens mais utilizadas atualmente, destacam-se os algoritmos evolucionários multiobjetivo (*Multiobjective Evolutionary Algorithms (MOEAs)*).

Esta popularidade se deve principalmente à natureza dos MOEAs de lidar simultaneamente com diversas soluções possíveis (população), o que permite a geração de várias soluções potenciais em uma única execução. Além disso, os MOEAs são menos suscetíveis à forma ou continuidade da fronteira de Pareto [9].

Uma metaheurística promissora para a otimização de MOPs é a otimização por nuvem de partículas multiobjetivo (*Multi-Objective Particle Swarm Optimization (MOPSO)*) [10]. Esta abordagem partilha muitas características com os MOEAs, entre elas o conceito de

população. O MOPSO é derivado da otimização por nuvem de partículas (*Particle Swarm Optimization (PSO)*) [25], que consiste em uma metaheurística populacional inspirada no comportamento de conjuntos de aves, criada para otimizar funções não lineares contínuas. Cada possível solução, chamada partícula, usa regras locais simples para guiar suas ações, e através de interações com o grupo, toda a população, chamada de nuvem ou enxame (*swarm*) atinge seus objetivos. O conjunto de possíveis soluções se move através espaço de busca em um procedimento cooperativo.

Estes movimentos são executados por um operador que controla a velocidade das partículas para uma determinada direção. Este operador é guiado por três componentes, um local, um social e o inercial. O componente local de uma partícula contém a melhor solução encontrada pela própria partícula durante toda a busca, o componente social representa a melhor solução encontrada por seus vizinhos, ou por todo o enxame, dependendo da topologia de vizinhança empregada, e o inercial define a influência da velocidade anterior da partícula sobre a velocidade atual.

Para expandir o PSO de modo que seja possível a otimização de MOPs, com isso criando o MOPSO, normalmente duas alterações são efetuadas: a incorporação de um mecanismo de seleção baseado na otimalidade de Pareto, e a adoção de um mecanismo de preservação de diversidade que evita a convergência para uma única solução. O propósito é descobrir soluções que não são dominadas por nenhuma outra no espaço de objetivos. Em muitos problemas, a busca pelo conjunto Pareto ótimo é NP-difícil [9], então estes algoritmos focam em encontrar um conjunto de aproximação (PF_{known}), o mais próximo possível da fronteira de Pareto real (PF_{true}).

MOPSOs têm apresentado bons resultados ao resolver MOPs [12, 30], este bom desempenho vem incentivando diversos pesquisadores a aprimorar esta abordagem para lidar com problemas de otimização com muitos objetivos (*Many-Objective Optimization Problems (MaOPs)*) [23], que consistem em problemas de otimização com mais de três objetivos.

A otimização de MaOPs é uma tarefa importante porque algumas aplicações do mundo real necessitam considerar e otimizar várias funções objetivo [41, 22, 20], entretanto,

devido à falta de algoritmos adequados, estes problemas normalmente são reduzidos para dois ou três objetivos e resolvidos [14].

No artigo [22], é explorado o problema de projetos de formas de onda para um radar Doppler pulsado (*Pulsed Doppler Radar*) típico em muitos sistemas de radar de aeronaves de combate. Este sistema deve medir a distância e a velocidade dos alvos, infelizmente, com as grandes distâncias e velocidades envolvidas, não é possível medir simultaneamente a distância e a velocidade dos alvos corretamente usando um formato de onda simples. Por isso vários formatos de onda simples são transmitidos, cada um subitamente diferente do último, então os dados são combinados para aumentar a precisão. O problema é como escolher o conjunto de formatos de onda simples.

Este problema envolve a otimização de nove objetivos, sendo eles:

1. Faixa de distância média do alvo antes que o conjunto de ondas não seja mais decodificável (em metros),
2. Faixa de velocidade média do alvo antes que o conjunto de ondas não seja mais decodificável (em ms-1),
3. Faixa de distância média do alvo antes que o conjunto de ondas tenha regiões cegas (em metros),
4. Faixa de velocidade média do alvo antes que o conjunto de ondas tenha regiões cegas (em ms-1),
5. Faixa de distância mínima do alvo antes que o conjunto de ondas não seja mais decodificável (em metros),
6. Faixa de velocidade mínima do alvo antes que o conjunto de ondas não seja mais decodificável (em ms-1),
7. Faixa de distância mínima do alvo antes que o conjunto de ondas tenha regiões cegas (em metros),

8. Faixa de velocidade mínima do alvo antes que o conjunto de ondas tenha regiões cegas (em ms-1),

9. Tempo necessário para transmitir todo o formato de onda (em milissegundos).

Neste problema os primeiros oito objetivos devem ser maximizados, enquanto o nono deve ser minimizado.

Na resolução de problemas com muitos objetivos como este, os algoritmos de otimização baseados em dominância de Pareto não apresentam bom desempenho, pois geralmente deterioram sua capacidade de busca de acordo com o aumento no número de objetivos. Isto ocorre principalmente porque a proporção de soluções não dominadas em uma população aumenta rapidamente com o número de objetivos, o que diminui a pressão de seleção em direção à fronteira de Pareto. Além disso, o número de pontos para representar a fronteira aumenta exponencialmente de acordo com o número de objetivos, dificultando a obtenção de soluções diversas o suficiente para cobrir toda a fronteira. Apesar de ser possível utilizar uma grande população com o poder computacional disponível atualmente, certamente é difícil para um tomador de decisões considerar um número muito grande de soluções [23, 14]. O objetivo da otimização com muitos objetivos é encontrar técnicas para superar estas limitações.

Este trabalho apresenta algumas técnicas empregadas atualmente para melhorar o desempenho do MOPSO para muitos objetivos, além de discutir os principais conceitos relacionados ao tema. Seu objetivo é aplicar técnicas voltadas a muitos objetivos que apresentaram bons resultados na literatura, além de desenvolver ou aprimorar técnicas já existentes para melhorar o desempenho do MOPSO e torná-lo menos sensível ao aumento no número de objetivos.

1.2 Objetivos

Neste trabalho optou-se por estudar a área de otimização multiobjetivo através da metaheurística nuvem de partículas multiobjetivo, especialmente por sua velocidade de convergência e capacidade de gerar diversas soluções ao mesmo tempo, constituindo, a cada

iteração uma fronteira aproximada viável. Portanto, o objetivo geral deste trabalho é aplicar técnicas a esta metaheurística para melhorar seu desempenho ao otimizar muitos objetivos. Para atingir este fim, alguns objetivos específicos podem ser destacados:

- Ganhar conhecimento em otimização de muitos objetivos, especialmente usando MOPSOs.
- Comparar e avaliar técnicas exclusivas do MOPSO para o aumento do seu desempenho.
- Aplicar, aperfeiçoar ou desenvolver técnicas para melhorar o desempenho dos MOPSOs para a otimização com muitos objetivos.

Para ganhar conhecimento na área, uma série de estudos sobre a bibliografia relacionada à otimização multiobjetivo foi efetuada, e posteriormente estudos empíricos foram conduzidos para comprovar os resultados teóricos. A comparação e avaliação de técnicas para o aumento do desempenho dos MOPSOs foi feita usando diversos métodos de escolha de líder para encontrar os métodos que obtêm os melhores desempenhos. Por fim, técnicas que obtiveram bons resultados na literatura para outros tipos de algoritmos bio-inspirados foram adaptadas e aplicadas ao MOPSO, como os métodos de controle da área de dominância CDAS [35] e S-CDAS [36], e uma abordagem usando pontos de referência [14].

Todos os estudos empíricos realizados utilizaram métricas de desempenho e funções de *benchmark* conhecidas no contexto de otimização com muitos objetivos.

1.3 Contribuições

Neste trabalho foram realizados estudos em conjunto com o aluno de doutorado André Britto de Carvalho, também orientando da Profa. Dra. Aurora Trinidad Ramirez Pozo. Estes trabalhos deram origem a três artigos: No primeiro [7] foi desenvolvido um estudo empírico para determinar quais métodos de líder apresentam o melhor desempenho para muitos objetivos entre diversos métodos de escolha de líder propostos na literatura e um

proposto neste trabalho. No segundo artigo [8] foi desenvolvido um novo MOPSO baseado em um estudo anterior [12] que usava a técnica CDAS [35] de controle da área de dominância aplicada ao MOPSO. No novo trabalho, foi criado um MOPSO que utiliza um aprimoramento da técnica CDAS chamado S-CDAS [36], além de terem sido realizados estudos empíricos para comparar ambos. O terceiro artigo [24] avalia a influência das técnicas de controle da área de dominância estudadas, no desempenho dos métodos de seleção de líder, além de determinar quais as melhores combinações de técnicas de controle da área de dominância e métodos de escolha de líder. Nesta dissertação também é apresentado um estudo não publicado no qual um novo MOPSO que utiliza pontos de referência como critério adicional para o arquivamento é proposto e seu desempenho é comparado através de estudos empíricos com outro MOPSO.

Portanto, as contribuições derivadas deste trabalho de pesquisa podem ser sumarizadas assim:

- A determinação, via análise empírica dos melhores métodos de escolha de líderes no MOPSO, na qual foram comparados os métodos Distância de agrupamento [13], Soma ponderada [4], NWSum [31], Sigma [29], Aleatório [10] e o novo método Oposto.

- A determinação, via análise empírica da melhor técnica de controle da área de dominância em situações de muitos e poucos objetivos no MOPSO, na qual foram comparadas as técnicas CDAS [35] e S-CDAS [36].

- Um MOPSO baseado em pontos de referência para a otimização com muitos objetivos [14].

1.4 Organização

O restante deste trabalho está organizado da seguinte forma: No Capítulo 2, são apresentados alguns trabalhos relacionados, incluindo trabalhos sobre a otimização com muitos objetivos e sobre MOPSOs para otimização multiobjetivo e com muitos objetivos. Os

principais conceitos relacionados à metaheurística nuvem de partículas mono e multiobjetivo, além dos problemas de teste e indicadores de desempenho utilizados neste trabalho são apresentados no Capítulo 3. Um estudo sobre as técnicas de controle da área de dominância CDAS e S-CDAS é apresentado no Capítulo 4. O Capítulo 5, apresenta uma revisão contendo diferentes métodos de escolha de líder populares na literatura, além de um novo método proposto, bem como uma análise empírica comparando-os. Um novo MOPSO voltado para muitos objetivos e baseado no conceito de pontos de referência é apresentado no Capítulo 6, e finalmente as conclusões deste trabalho são apresentadas no Capítulo 7.

CAPÍTULO 2

TRABALHOS RELACIONADOS

A resolução de MaOPs apresenta alguns problemas específicos a serem resolvidos, neste capítulo são apresentados alguns trabalhos abordando estes problemas e algumas soluções viáveis. Nos trabalhos de Jaimes e Coello [27] e Ishibuchi *et al.* [23] os principais problemas são descritos. Entre eles estão a deterioração da capacidade de busca devido ao rápido aumento da proporção de soluções não dominadas na população de acordo com o número de objetivos, o aumento exponencial no número de soluções requeridas para aproximar toda a fronteira de Pareto, além da dificuldade de visualização das soluções obtidas.

As soluções mais comuns destes problemas também são indicadas, sendo elas: a redução do número de objetivos do problema, que identifica os objetivos menos conflitantes e os remove. A adoção de relações de preferência para induzir um melhor ordenamento do espaço de objetivos. Além do uso de modificação da relação de dominância, reduzindo o número de soluções não dominadas.

No artigo de Ishibuchi *et al.* [23], além de uma discussão sobre os problemas relacionados aos MaOPs, são realizados experimentos utilizando o algoritmo NSGA-II na otimização de problemas entre dois e oito objetivos para demonstrar seu efeito na prática, além disso, as principais técnicas para melhorar a escalabilidade dos MOEAs são explicadas.

No trabalho de López e Coello [27], além da discussão sobre os problemas de escalabilidade, duas contribuições são apresentadas: primeiramente são propostos dois métodos para a incorporação de um algoritmo de redução do número de objetivos em um otimizador, esta incorporação serve para reduzir os objetivos durante o processo de busca. O algoritmo incorporado encontra um subconjunto dos objetivos que produz o mínimo de erro possível. Um dos métodos de incorporação reduz o número de objetivos sucessivamente durante a maior parte da busca, e somente ao se aproximar do fim da busca todos

os objetivos são reintegrados. O outro método alterna durante a busca o processo de redução com o de integração dos objetivos. A segunda contribuição é um estudo comparativo preliminar de relações de preferência propostas para lidar com muitos objetivos, os métodos comparados foram o *average ranking (AR)*, *maximum ranking (MR)*, *favour relation*, *preference order relation (POR)* e *Controlling Dominance Area of Solutions (CDAS)*. Ambas as contribuições amenizaram os problemas causados pelo aumento no número de objetivos.

Em geral os algoritmos de otimização multiobjetivo encontrados na literatura obtêm bons resultados ao resolver MOPs, entretanto apresentam resultados ruins quando utilizados no contexto de muitos objetivos. Existem poucos trabalhos na literatura que exploram a otimização de muitos objetivos. López e Coello utilizaram em [28] diferentes relações de preferência para aumentar a convergência na busca, entretanto estas preferências diminuem a extensão de cobertura da fronteira obtida pelas soluções, resultando em uma relação inversa entre convergência e porção da fronteira coberta.

Sato *et al.* [35] observou que para problemas com muitos objetivos utilizar apenas a relação de dominância para ranquear soluções não é mais efetivo, portanto este trabalho propõe um método chamado controle da área de dominância das soluções (*Controlling Dominance Area of Solutions (CDAS)*). Este método utiliza uma variável S, que contrai ou expande a área de dominância das soluções aumentando a convergência ou a diversidade da busca de acordo com o valor desta variável.

Uma continuação deste trabalho é apresentada por Sato *et al.* em [36], no qual é apresentado um método chamado auto controle da área de dominância das soluções (*Self-Controlling Dominance Area of Solutions (S-CDAS)*). Nesta abordagem, o algoritmo controla automaticamente o grau de expansão ou contração da área de dominância para cada solução sem a necessidade de parâmetros adicionais. Estes dois trabalhos inspiraram as contribuições desenvolvidas no Capítulo 4, e parte das contribuições desenvolvidas no Capítulo 5.

Um novo NSGA-II aprimorado para trabalhar com muitos objetivos, chamado de M-NSGA-II é apresentado por Deb e Jain em [14]. Neste novo algoritmo pontos de

referência bem distribuídos são gerados em um hiperplano, e os membros da população são associados a estes pontos, com isto, soluções são encontradas próximas de cada ponto de referência, obtendo uma maior cobertura da fronteira real, com pontos bem espaçados sobre esta fronteira. O novo algoritmo foi testado em dois problemas reais, um deles relacionado à resistência ao choque no projeto de veículos, com três objetivos, e um segundo relacionado ao projeto de cabines para veículos, com nove objetivos. O Capítulo 6 detalha o novo algoritmo e apresenta uma adaptação destas ideias para o MOPSO.

Adra e Fleming [2] discutem algumas características da otimização com muitos objetivos. Além disso, propõem dois mecanismos de gerenciamento de diversidade para melhorar o desempenho na otimização de MaOPs usando o algoritmo NSGA-II. O primeiro mecanismo ativa ou desativa a promoção de diversidade de acordo com a distribuição das soluções em uma população. Esta promoção de diversidade consiste em dois operadores genéticos: seleção para variação e seleção para sobrevivência. O outro mecanismo proposto adapta o intervalo de mutação para cada variável de decisão de acordo com condições locais de distribuição ou agrupamento. Estas técnicas têm o objetivo de balancear o conflito entre convergência e a diversidade da busca. Nos experimentos realizados, a primeira técnica apresentou bons resultados, enquanto a segunda não apresentou melhora de desempenho no algoritmo.

Um estudo envolvendo seis MOPSOs é apresentado por Durillo *et al.* em [19] no qual análises empíricas são realizadas para compará-los e identificar qual deles apresenta o melhor desempenho usando três famílias de problemas de teste. Ao observar os resultados dos experimentos verificou-se que as partículas apresentavam movimentos erráticos devido ao excesso de velocidade levando ao desenvolvimento do algoritmo SMPSO, que se destacou por obter resultados promissores em problemas nos quais outros MOPSOs apresentaram desempenhos inadequados. Em [30], Nebro *et al.* compara o SMPSO com cinco algoritmos multiobjetivo importantes, e este apresenta o melhor desempenho entre eles em problemas bi-objetivo. Devido ao seu bom desempenho [18, 17] o SMPSO é utilizado como algoritmo base nos experimentos realizados neste trabalho.

A abordagem MOPSO exige algumas características específicas para lidar adequa-

damente com problemas multiobjetivo especialmente devido à ausência de uma forma direta de ranquear as soluções. Uma dessas características é a seleção do líder, que não é uma tarefa trivial e pode ser feita utilizando diversos métodos diferentes. Alguns desses métodos são revistos por Padhye *et al.* em [31]. Neste trabalho as características de convergência, diversidade e tempos computacionais dos mesmos são consideradas. A comparação também inclui dois métodos propostos por eles, chamados NWSum, que é uma variação do WSum proposto por Branke e Mostaghim em [4], e o *Indicator Based*, que usa a métrica hipervolume [46] proposta por Zitzler e Thiele como um indicador para a seleção de líder. Os métodos propostos apresentaram bons resultados, indicando que a escolha pode modificar o desempenho de um MOPSO. Este trabalho motivou o desenvolvimento dos estudos apresentados no Capítulo 5.

Zhang *et al.* propõem em [44] um MOPSO chamado mPSO-DHA, que utiliza um novo sistema de arquivamento baseado em hipercubos que consiste em uma modificação do sistema de arquivamento em hipercubos proposto por Coello e Lechuga em [10], neste novo sistema os limites do espaço de objetivos são modificados dinamicamente durante o processo de otimização. Um processo de mutação também é empregado para ajudar as partículas a fugir de ótimos locais, além de outros mecanismos. O mPSO-DHA superou outros MOPSOs nos experimentos realizados, apresentando bons resultados.

Na literatura existem alguns trabalhos que lidam com MOPSO para MaOPs. Entre eles destaca-se o trabalho desenvolvido por Carvalho e Pozo em [12], que explora a influência de diferentes relações de dominância entre as soluções do repositório usando dois algoritmos MOPSO. Com isto o algoritmo CDAS-SMPSO é proposto neste trabalho como forma de obter um melhor desempenho ao otimizar MaOPs. Outra contribuição deste trabalho é a identificação dos melhores valores do parâmetro S os problemas DTLZ2, DTLZ4 e DTLZ7.

Outro trabalho importante explorando MOPSOs para muitos objetivos é apresentado por Carvalho e Pozo em [5], nele, sete métodos de arquivamento encontrados na literatura são revisados, além disso, quatro novos métodos são propostos e um método aleatório também é usado. Estudos empíricos são realizados, e as características de cada método

são analisadas. Nos resultados obtidos, os métodos de arquivamento que não limitam o tamanho do repositório obtêm melhores resultados. Entretanto, dado o alto custo computacional de se manter um arquivo com muitas soluções, o método de arquivamento com limite de soluções que apresentou a maior convergência foi o Ideal, proposto neste trabalho, entretanto ele limita a busca somente à região próxima ao joelho da fronteira.

Hughes [22] apresenta um problema real envolvendo a otimização de muitos objetivos que pode ser usado como problema de teste para avaliar otimizadores. Neste trabalho, ele explora o problema de desenvolver projetos de formas de onda (*waveforms*) para um radar Doppler pulsado (*Pulsed Doppler Radar*) típico em muitos sistemas de radar de aeronaves de combate. Este sistema deve medir a distância e a velocidade dos alvos, infelizmente, com as grandes distâncias (aproximadamente 185 Km) e velocidades (eventualmente mach 5) envolvidas, não é possível medir simultaneamente a distância e a velocidade dos alvos corretamente usando um formato de onda simples.

Para permitir medir a distância e a velocidade simultaneamente de forma correta, vários formatos de onda simples são transmitidos, cada um subitamente diferente do último, então os dados são combinados para aumentar a precisão. O problema é como escolher o conjunto de formatos de onda simples.

Este problema envolve a otimização de nove objetivos, sendo eles:

1. Faixa de distância média do alvo antes que o conjunto de ondas não seja mais decodificável (em metros),

2. Faixa de velocidade média do alvo antes que o conjunto de ondas não seja mais decodificável (em ms-1),

3. Faixa de distância média do alvo antes que o conjunto de ondas tenha regiões cegas (em metros),

4. Faixa de velocidade média do alvo antes que o conjunto de ondas tenha regiões cegas (em ms-1),

5. Faixa de distância mínima do alvo antes que o conjunto de ondas não seja mais decodificável (em metros),

6. Faixa de velocidade mínima do alvo antes que o conjunto de ondas não seja mais decodificável (em ms-1),

7. Faixa de distância mínima do alvo antes que o conjunto de ondas tenha regiões cegas (em metros),

8. Faixa de velocidade mínima do alvo antes que o conjunto de ondas tenha regiões cegas (em ms-1),

9. Tempo necessário para transmitir todo o formato de onda (em milissegundos).

Neste problema os primeiros oito objetivos devem ser maximizados, enquanto o nono deve ser minimizado.

Para otimizar estes objetivos foram utilizados três algoritmos: NSGA-II [13], MSOPS [21] além de um terceiro algoritmo derivado do MSOPS chamado de CAAOS. Nos experimentos conduzidos os três obtiveram desempenho similar apesar dos testes e algoritmos completamente independentes, e foi hipoteticamente assumido que os conjuntos de soluções não dominadas obtidos estão localizados bem próximos à fronteira de Pareto ótima.

A partir dos trabalhos analisados, verificou-se que existem poucos estudos envolvendo MOPSOs para a otimização de problemas com muitos objetivos, e que esta metaheurística tem apresentado bons resultados nos trabalhos estudados, portanto o objetivo desta dissertação é comparar as principais técnicas disponíveis na literatura para lidar com MaOPs e aplicar as que obtiverem melhores resultados no MOPSO para criar um novo algoritmo menos sensível ao aumento no número de objetivos.

CAPÍTULO 3

OTIMIZAÇÃO POR NUVEM DE PARTÍCULAS

Otimização por nuvem de partículas, do inglês *Particle Swarm Optimization* (PSO), proposta em [25] é uma metaheurística simples inspirada no comportamento social de grupos de pássaros procurando alimento, projetada para a otimização de funções não lineares.

O PSO utiliza um conjunto de soluções possíveis que se movem através do espaço de busca até que uma solução ótima ou um critério de parada seja atingido. Neste caso, cada solução $\vec{x}$ é representada por uma partícula. E uma nuvem ou enxame *(swarm)* representa um conjunto de partículas (população). A responsabilidade por mover o enxame para a região ótima fica a cargo das equações de posição e velocidade, que são compostas de três elementos: a inércia de velocidade, o componente cognitivo ($p\vec{best}$), e o componente social ($g\vec{best}$) [25].

O componente cognitivo ou ótimo pessoal ($p\vec{best}$) representa a melhor solução encontrada pela própria partícula. Já o líder social, também chamado de ótimo global ($g\vec{best}$), representa a melhor solução encontrada por todo o enxame ou um subconjunto deste, conforme a topologia de vizinhança empregada. Estes componentes são escolhidos de acordo com uma medida de desempenho, similar ao *fitness* usado em outras abordagens [9].

A posição de cada partícula é alterada de acordo com sua própria experiência e a de seus vizinhos. $\vec{x}(t)$ denota a posição de uma partícula $\vec{p}$ no momento t. A posição de $\vec{p}$ é atualizada adicionando-se a velocidade $\vec{v}(t)$ à sua posição atual, como mostrado na Equação 3.1 [34].

$$\vec{x} = \vec{x} + \vec{v} \tag{3.1}$$

O vetor de velocidade reflete as informações trocadas cooperativamente, e em geral é

definido como na Equação 3.2.

$$\vec{v}(t) = \omega\vec{v}(t-1) + C_1 r_1(\vec{x}_{pbest} - \vec{x}(t)) + C_2 r_2(\vec{x}_{gbest} - \vec{x}(t)) \tag{3.2}$$

Na qual $r_1, r_2 \in [0, 1]$ são valores aleatórios e ω representa o peso de inércia empregado para controlar o impacto da velocidade anterior na atual. Os fatores de aprendizado C_1 e C_2 definem a atração da partícula ao seu próprio sucesso, e a atração da partícula em direção ao sucesso dos seus vizinhos, respectivamente [34].

As partículas são influenciadas pelo sucesso de outras partículas conectadas a ela. É a topologia de vizinhança que define a estrutura social do enxame, ou seja, a forma como as partículas se conectam entre si. As principais estruturas de vizinhança usadas no PSO são [34]:

- **Grafo vazio:** Nesta topologia as partículas estão isoladas e comparam sua posição atual somente com suas próprias melhores posições já encontradas.

- **Melhor local:** Nesta topologia cada partícula é afetada pelo melhor desempenho de seus vizinhos imediatos, além de seu próprio melhor desempenho.

- **Grafo totalmente conectado:** Esta topologia é o contrário da grafo vazio, nela todos os membros do enxame estão conectados. Cada partícula usa sua melhor posição já encontrada além da posição da melhor partícula do enxame. O algoritmo SMPSO utiliza esta topologia por padrão em sua implementação.

- **Estrela:** Uma partícula é conectada a todas as outras, e elas são conectadas somente a uma (chamada partícula focal). Nesta topologia as partículas são isoladas umas das outras, e toda a informação tem de passar através da partícula focal, esta compara os desempenhos de todo o enxame e ajusta sua trajetória em direção à melhor partícula. Este desempenho é comunicado ao restante do enxame eventualmente.

- **Árvore:** Nesta topologia todas as partículas são arranjadas em uma árvore, e cada nó contém uma partícula. Uma partícula é influenciada pela sua própria melhor posição já encontrada, além da melhor posição da partícula diretamente sobre si na árvore (nó pai). Se uma partícula em um nó filho encontrar uma posição melhor que todas as já encontradas por seu nó pai, estas partículas trocam de posição. Assim esta topologia oferece uma vizinhança dinâmica.

O Algoritmo 1 mostra como um PSO básico funciona. Primeiramente o enxame é inicializado, tanto as posições quanto as velocidades. O $\vec{pbest}$ de cada partícula é inicializado, e o $\vec{gbest}$ é determinado. Então por um dado número de iterações ($gmax$) cada partícula se movimenta no espaço de busca atualizando sua posição e velocidade através das Equações 3.1 e 3.2, além de avaliar a solução atual e atualizar o $\vec{pbest}$. Ao final da iteração o $\vec{gbest}$ é atualizado, e ao final da execução o melhor $\vec{gbest}$ já encontrado é retornado.

Algoritmo 1: Pseudocódigo do PSO

```
início
    Inicialize nuvem
    Determine o gbest
    g = 0
    enquanto g < gmax faça
        para Cada partícula faça
            Atualize posição e velocidade
            Avalie
            Atualize o pbest
        fim para
        Atualize o gbest
        g++
    fim enqto
    retorne gbest
fim
```

A simplicidade e velocidade de convergência do PSO o fazem um candidato altamente viável a ser usado não só para problemas com uma única função objetivo, mas também para a otimização de problemas multiobjetivo [9].

3.1 Otimização multiobjetivo

Um problema multiobjetivo (*Multi-Objective Problem (MOP)*) envolve a satisfação simultânea de duas ou mais funções objetivo, normalmente em conflito. Um problema de minimização multiobjetivo pode ser expresso como em (3.3).

$$Minimizar \vec{f}(\vec{x}) = (f_1(\vec{x}), f_2(\vec{x})..., f_m(\vec{x})) \tag{3.3}$$

sujeito a $\vec{x} \in \Omega$, no qual: $\vec{x} = (x_1, ..., x_k)$ é um vetor de soluções possíveis, Ω é a região de soluções possíveis delimitada pelas restrições do problema, e m é o número de objetivos. Neste caso, o propósito é otimizar m funções objetivo simultaneamente, com a finalidade de encontrar soluções que apresentem um bom compromisso entre os objetivos. Para isto é usado o conceito de otimalidade de Pareto [32], que é apresentado a seguir conforme [9]:

A dominância de Pareto define que um vetor $\vec{u} = (u_1, ..., u_k)$, com $\vec{u} = \vec{f}(\vec{x})$ e $\vec{x} \in \Omega$ domina outro vetor $\vec{v} = (v_1, ..., v_k)$, com $\vec{v} = \vec{f}(\vec{y})$ e $\vec{y} \in \Omega$ denotada por $\vec{u} \prec \vec{v}$, se e somente se $\vec{u}$ for parcialmente menor que $\vec{v}$, por exemplo, $\forall i \in \{1, ..., k\}, u_i \leq v_i \wedge \exists i \in \{1, ..., k\} : u_i < v_i$. Ou seja, $\vec{u}$ domina $\vec{v}$ se houver pelo menos um elemento em $\vec{u}$ menor que o mesmo elemento em $\vec{v}$ e todos os elementos de $\vec{v}$ forem menores ou iguais os elementos de $\vec{u}$.

Um vetor de objetivos $\vec{u}$ pode ser dito não dominado com respeito ao seu espaço de objetivos O, se não existir nenhum vetor de objetivos $\vec{v}$ que o domine em O.

Uma solução $\vec{x} \in \Omega$ pode ser considerada Pareto ótima com respeito a Ω se não houver nenhum $\vec{y} \in \Omega$, tal que $\vec{v} = \vec{f}(\vec{y})$ domine $\vec{u} = \vec{f}(\vec{x})$.

O conjunto ótimo de Pareto P^* é definido por: $P^* = \{\vec{x} \in \Omega | \vec{x}$ é Pareto ótima $\}$. A fronteira de Pareto é definida por: $FP = \{\vec{u} = \vec{f}(\vec{x}) | \vec{x} \in P^*\}$.

A Figura 3.1 apresenta um exemplo de fronteira de Pareto para um problema de minimização com dois objetivos. Nesta figura, os pontos pretos representam as soluções não dominadas que compõem a fronteira de Pareto e os pontos cinza representam soluções dominadas. As linhas tracejadas representam o limite que define se uma solução domina

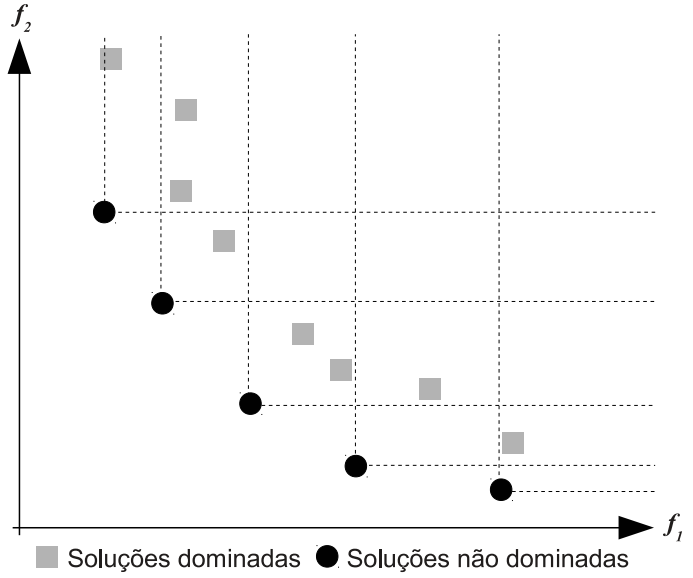


Figura 3.1: Exemplo de fronteira de Pareto

a outra. As soluções que estão dentro da área definida pelas linhas possuem piores valores nos objetivos, ou seja, são dominadas.

Apesar da maioria dos estudos em MOPs serem focados em problemas com dois ou três objetivos, problemas de otimização práticos envolvem um grande número destes critérios [41, 20]. Portanto, esforços de pesquisa foram orientados para investigar a escalabilidade destes algoritmos com respeito ao número de objetivos [23, 12]. MOPs com mais de três objetivos são chamados de problemas de otimização com muitos objetivos (*Many-Objective Optimization Problems (MaOPs)*), e a área que estuda soluções para os problemas causados pelo aumento no número de objetivos é chamada de otimização com muitos objetivos (*Many-Objective Optimization*).

Diversos estudos provaram que MOEAs apresentam baixo desempenho em MaOPs [23, 12, 27]. Os principais problemas encontrados, conforme [23] são:

1. Deterioração da habilidade de busca em algoritmos baseados em dominância de Pareto. Quando o número de objetivos aumenta, muitas soluções tornam-se não dominadas, o que enfraquece severamente a pressão de seleção em direção à fronteira de Pareto, prejudicando muito a capacidade de convergência desses algoritmos.

2. Aumento exponencial no número de soluções requeridas para aproximar a fronteira

de Pareto inteira. O objetivo dos MOEAs é encontrar um conjunto de soluções não dominadas que melhor se aproxime da fronteira de Pareto. Uma vez que esta fronteira é uma hiper-superfície no espaço de objetivos, o número de soluções requeridas para esta aproximação aumenta exponencialmente com o número de objetivos.

3. Dificuldade de visualização das soluções. Normalmente se assume que a escolha de uma solução final de um conjunto de soluções não dominadas obtido é feita por um tomador de decisões baseada em sua preferência. Com o aumento do número de objetivos, a visualização das soluções obtidas torna-se muito difícil. O que significa que a escolha de uma solução final é dificultada.

Para evitar estes problemas, atualmente a comunidade de pesquisa os tem abordado usando principalmente três estratégias: adaptação de relações de preferência [35, 36] para controlar a relação de dominância e induzir um ranqueamento apropriado das soluções, redução de dimensionalidade [26], cuja ideia geral é identificar o maior número de objetivos não conflitantes e descartá-los, e as estratégias de decomposição [45], que usam métodos de decomposição estudados pela comunidade de programação matemática em algoritmos evolutivos para otimização multiobjetivo.

3.2 Otimização por nuvem de partículas multiobjetivo

A extensão do PSO para problemas multiobjetivo é chamada de *(Multi-Objective Particle Swarm Optimization (MOPSO))*, nela algumas características são modificadas em relação ao PSO original.

As duas principais abordagens introduzidas no MOPSO são o conceito de otimalidade de Pareto, que determina a relação de dominância entre as soluções, e a introdução de um arquivo externo ou repositório que armazena as melhores soluções não dominadas encontradas durante a busca. Neste arquivo entram somente soluções não dominadas com respeito ao conteúdo deste. Caso uma solução nova domine soluções já presentes no arquivo, as soluções antigas que forem dominadas são excluídas. Este arquivo é usado tanto como fonte para a seleção de líderes, quanto para ter seu conteúdo reportado ao

final da execução do algoritmo [34].

Esta abordagem apresenta o problema de aumentar o tamanho do arquivo muito rapidamente, especialmente porque este necessita ser atualizado a cada iteração. A complexidade de se atualizar este arquivo é de $O(mN^2)$, com N sendo o tamanho do enxame e m é o número de objetivos. Por isso o arquivo necessita ser podado, o que faz necessária uma estratégia para decidir quais soluções não dominadas devem ser descartadas quando o arquivo estiver cheio [34].

Algoritmo 2: Pseudocódigo do MOPSO

```
início
    Inicialize nuvem
    Inicialize os líderes em um arquivo externo
    Avaliação dos líderes
    g = 0
    enquanto g < gmax faça
        para Cada partícula faça
            Selecione o líder
            Atualize posição e velocidade
            Mutação
            Avalie
            Atualize o pbest
        fim para
        Atualize os líderes no arquivo externo
        Avaliação dos líderes
        g++
    fim enqto
    retorne as soluções no arquivo externo
fim
```

O Algoritmo 2 mostra o funcionamento geral do MOPSO. Primeiramente o enxame é inicializado, então um conjunto de líderes também é inicializado com as partículas não dominadas do enxame. Em seguida um critério de avaliação dos líderes é calculado para a seleção destes pelas partículas na próxima etapa, exemplos de critérios de avaliação são a distância de agrupamento ou o vetor Sigma, detalhados no Capítulo 5. A cada iteração do algoritmo todas as partículas têm um líder selecionado e a posição e velocidade atualizadas. Caso a melhor posição da partícula e a atual não se dominem entre si (incomparáveis), a nova solução substitui a antiga. Alguns MOPSOs aplicam algum tipo de operador de mutação ou turbulência após a atualização da posição. Cada partícula é

avaliada e seus correspondentes $p\vec{best}$ são atualizados. Depois desta atualização, o arquivo externo também é atualizado, e finalmente a medida de avaliação dos líderes é calculada novamente. Este processo é repetido por um número predefinido de iterações ($gmax$), e ao final as soluções contidas no arquivo externo são retornadas como a fronteira aproximada.

3.2.1 SMPSO

O SMPSO *(Speed-constrained Multi-objective Particle Swarm Optimization)* foi apresentado em [30], e consiste em um MOPSO que usa um mecanismo para limitar a velocidade das partículas. Em certas condições especiais a velocidade das partículas em um MOPSO pode se tornar muito alta, gerando movimentos erráticos em direção aos limites das posições das partículas, este comportamento pode ser prevenido usando um mecanismo de limitação de velocidade.

Para controlar a velocidade das partículas, ao invés de se utilizar parâmetros máximos e mínimos para limitar a quantidade de alteração da velocidade, é adotado um coeficiente de constrição calculado conforme a Equação 3.4 [30].

$$\chi = \frac{2}{2 - \varphi - \sqrt{\varphi^2 - 4\varphi}} \tag{3.4}$$

Na qual[1]:

$$\varphi = \begin{cases} C_1 + C_2 & \text{se} \quad C_1 + C_2 > 4 \\ 1 & \text{se} \quad C_1 + C_2 \leq 4 \end{cases} \tag{3.5}$$

Portanto a nova velocidade é definida conforme a Equação 3.6

$$\vec{v} = \chi\vec{v} \tag{3.6}$$

[1]Esta é a fórmula do cálculo de φ apresentada em [30], entretanto durante a implementação, caso $\varphi = 1$ então $\sqrt{1^2 - 4 \times 1} = \sqrt{-3}$, neste caso esta raiz negativa é substituída por 0.

Além disso, é criado um mecanismo no qual a velocidade acumulada por cada variável j em cada partícula é podada por meio da Equação 3.7.

$$v_{i,j}(t) = \begin{cases} delta_j & \text{se} \quad v_{i,j}(t) > delta_j \\ -delta_j & \text{se} \quad v_{i,j}(t) \leq -delta_j \\ v_{i,j}(t) & \text{nos outros casos} \end{cases} \tag{3.7}$$

Na qual:

$$delta_j = \frac{(limite\_superior_j - limite\_inferior_j)}{2} \tag{3.8}$$

Resumindo o procedimento, a velocidade das partículas é calculada de acordo com a Equação 3.2, esta velocidade é então multiplicada pelo fator de constrição (Equação 3.4), e o valor resultante é restrito usando a Equação 3.7, resultando na velocidade final a ser usada no cálculo de posição da partícula [30].

Tanto o mecanismo de poda do arquivo externo quanto o de seleção de líderes originais do SMPSO utilizam a distância de agrupamento. No mecanismo de poda, a solução com a menor distância de agrupamento é removida do arquivo, pois representa um candidato a líder presente em uma região muito populosa. No método de seleção de líderes, dois líderes são selecionados aleatoriamente e o que tiver maior distância de agrupamento é escolhido, pois representa uma solução em uma região menos populosa do repositório.

Os parâmetros básicos utilizados neste trabalho estão disponíveis em [30]. Com o coeficiente de inércia ω variando aleatoriamente entre [0, 0.8] e as constantes C_1 e C_2 variando no intervalo [1.5, 2.5]. Uma mutação polinomial é aplicada com probabilidade $p_{mut} = 1/m$, com m representando o número de variáveis de decisão do problema. Cada

variável da velocidade é limitada no intervalo [-5, +5]. Entretanto neste trabalho o número de iterações, população e tamanho do repositório foram alterados conforme o conjunto de experimentos efetuado, portanto estes parâmetros estão disponíveis em cada seção de experimentos.

3.3 Problemas de *benchmark*

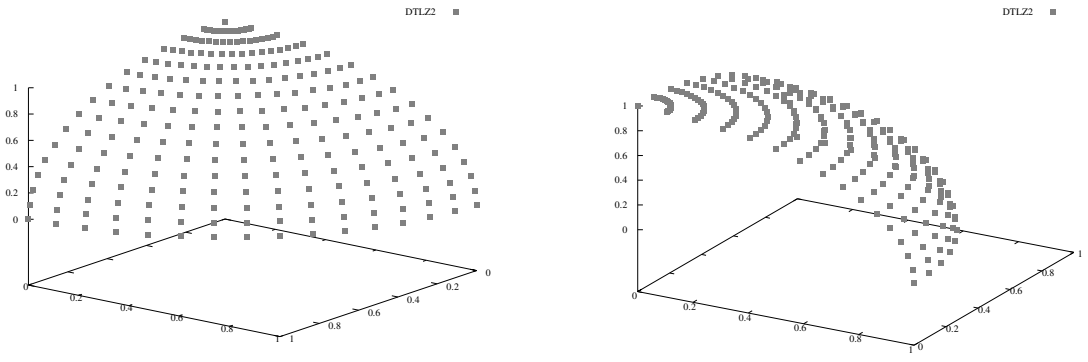
Em otimização multiobjetivo existem alguns problemas de teste bastante empregados na literatura, entre eles destaca-se a família de problemas ZDT [47]. Embora sejam muito usados para testar otimizadores, os problemas desta família não são escaláveis quanto ao número de objetivos, o que impossibilita sua utilização com mais de dois objetivos.

Os problemas utilizados neste trabalho são parte da conhecida família DTLZ de MOPs [15]. A família DTLZ é composta por uma série de problemas de referência comumente usados na avaliação de otimizadores multiobjetivo, especialmente no contexto de muitos objetivos. Em todos os problemas da classe DTLZ deve-se **minimizar** as funções objetivo. Neste trabalho três problemas da família DTLZ são usados: DTLZ2, DTLZ4 e DTLZ6.

O DTLZ2 pode ser usado para investigar a habilidade de um algoritmo de manter seu desempenho à medida que o número de objetivos aumenta. Este problema apresenta uma fronteira de Pareto ótima na forma esférica, e pode ser descrito conforme a Equação 3.9.

$$
\begin{aligned}
&Minimize \quad f_1(\vec{x}) = (1+g(x_m))cos(x_1\pi/2) \quad ... \quad cos(x_{m-2}\pi/2)cos(x_{m-1}\pi/2),\\
&Minimize \quad f_2(\vec{x}) = (1+g(x_m))cos(x_1\pi/2) \quad ... \quad cos(x_{m-2}\pi/2)sen(x_{m-1}\pi/2),\\
&Minimize \quad f_3(\vec{x}) = (1+g(x_m))cos(x_1\pi/2) \quad ... \quad sen(x_{m-1}\pi/2),\\
&\quad \vdots \qquad\qquad\qquad \vdots \\
&Minimize \quad f_m(\vec{x}) = (1+g(x_m))sen(x_1\pi/2),\\
&\quad com \qquad g(x_m) = \textstyle\sum_{x_i \in X_m}(x_i - 0,5)^2\\
&\qquad\quad 0 \leq x_i \leq 1, \text{para } i = 1, 2, ..., n,
\end{aligned}
\tag{3.9}
$$

Na Equação 3.9, m representa o número de objetivos, as soluções Pareto ótimas correspondem a x_i=0,5 para todos os $x_i \in x_m$ e todos os valores de função objetivo devem satisfazer $\sum_{i=1}^{m}(f_i)^2 = 1$. O número total de variáveis de decisão é $n = m + k - 1$, é recomendado usar $k = |x_m| = 10$. Uma representação gráfica da fronteira real tridimensional para o problema DTLZ2 é apresentada na Figura 3.2.



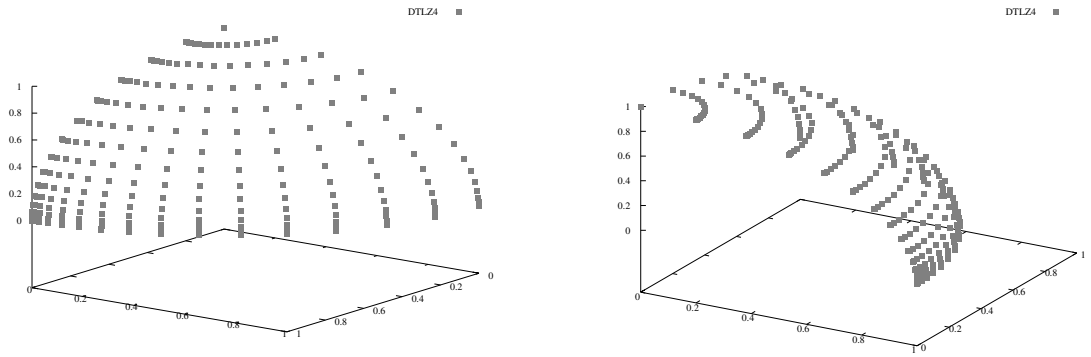
(a) Visão frontal (b) Visão lateral

Figura 3.2: Fronteira real tridimensional para o problema DTLZ2

Conforme [15], o problema DTLZ4 é uma modificação do DTLZ2 que foca em investigar a habilidade de um algoritmo de manter uma boa distribuição de soluções. Nele uma meta-variável diferente é usada $\vec{x} \rightarrow \vec{x}^{\alpha}$, esta modificação permite que um denso conjunto de soluções exista próximo ao plano $F_m - f_1$.

$$
\begin{aligned}
Minimize \quad & f_1(\vec{x}) = (1 + g(x_m))cos(x_1^{\alpha}\pi/2) \quad ... \quad cos(x_{m-2}^{\alpha}\pi/2)cos(x_{m-1}^{\alpha}\pi/2), \\
Minimize \quad & f_2(\vec{x}) = (1 + g(x_m))cos(x_1^{\alpha}\pi/2) \quad ... \quad cos(x_{m-2}^{\alpha}\pi/2)sen(x_{m-1}^{\alpha}\pi/2), \\
Minimize \quad & f_3(\vec{x}) = (1 + g(x_m))cos(x_1^{\alpha}\pi/2) \quad ... \quad sen(x_{m-1}^{\alpha}\pi/2), \\
\vdots \quad & \qquad\qquad\qquad \vdots \\
Minimize \quad & f_m(\vec{x}) = (1 + g(x_m))sen(x_1^{\alpha}\pi/2), \\
com \quad & g(x_m) = \textstyle\sum_{x_i \in X_m}(x_i - 0,5)^2. \\
& 0 \leq x_i \leq 1, \text{para } i = 1, 2, ..., n,
\end{aligned}
\tag{3.10}
$$

Na Equação 3.10, o parâmetro α tem um valor sugerido de 100. m representa o número de objetivos e todas as variáveis x_1 a x_{m-1} variam entre 0 e 1. O parâmetro $k = 10$ também é sugerido e o número de variáveis de decisão é $n = m + k - 1$. Uma representação gráfica da fronteira real tridimensional para o problema DTLZ4 é apresentada na Figura 3.3.



(a) Visão frontal (b) Visão lateral

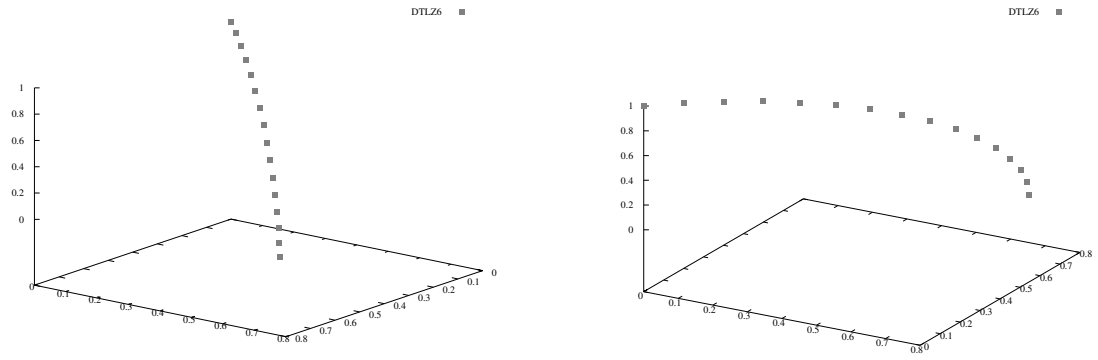
Figura 3.3: Fronteira real tridimensional para o problema DTLZ4

O problema DTLZ6 também é derivado do problema DTLZ2, entretanto, sua dificuldade está na modificação da função g com $\sum_{i=1}^{k} z_i^{0.1}$.

$$
\begin{aligned}
Minimize \quad & f_1(\vec{x}) = (1 + g(x_m))cos(\theta_1\pi/2) \quad ... \quad cos(\theta_{m-2}\pi/2)cos(\theta_{m-1}\pi/2), \\
Minimize \quad & f_2(\vec{x}) = (1 + g(x_m))cos(\theta_1\pi/2) \quad ... \quad cos(\theta_{m-2}\pi/2)sen(\theta_{m-1}\pi/2), \\
Minimize \quad & f_3(\vec{x}) = (1 + g(x_m))cos(\theta_1\pi/2) \quad ... \quad sen(\theta_{m-1}\pi/2), \\
& \vdots \qquad\qquad\qquad\qquad \vdots \\
Minimize \quad & f_m(\vec{x}) = (1 + g(x_m))sen(\theta_1\pi/2), \\
Com \quad & \theta_i = \tfrac{\pi}{4(1+g(x_m))}(1 + 2g(x_m)x_i) \qquad \text{para } i = 2, 3, ..., (m-1), \\
& g(x_m) = \textstyle\sum_{x_i \in X_m} x_i^{0,1}, \\
& 0 \leq x_i \leq 1, \text{para } i = 1, 2, ..., n,
\end{aligned}
\tag{3.11}
$$

Na Equação 3.11, a fronteira ótima corresponde a $x_i = 0$ para todo $x_i \in x_m$. O

tamanho do vetor x_m é 10 e o número total de variáveis é $n = m + k - 1$. Este é um problema que apresenta uma dificuldade para convergir, esta falta de convergência faz com que os algoritmos encontrem uma superfície dominada como fronteira aproximada, enquanto a fronteira real é uma curva em qualquer número de objetivos [9]. Uma representação gráfica da fronteira real tridimensional para o problema DTLZ6 é apresentada na Figura 3.4.



(a) Visão frontal (b) Visão lateral

Figura 3.4: Fronteira real tridimensional para o problema DTLZ6

3.4 Medidas de desempenho

De acordo com [1], os três principais requisitos para um otimizador multiobjetivo são:

- **Convergência**: O conjunto de aproximação obtido por um MOP (PF_{known}) deve ser o mais próximo possível da fronteira real (PF_{true}).

- **Diversidade**: Devido à não existência de uma única solução ideal em otimização multiobjetivo com objetivos conflitantes, e devido ao fato de que a superfície da fronteira real pode, potencialmente apresentar um número infinito de soluções, o conjunto de soluções obtido deve estar bem distribuído, e cobrindo uniformemente grandes áreas de fronteira de Pareto. Esta diversidade normalmente é preferida no espaço de objetivos para apresentar ao tomador de decisões, e pode ser baseada em prioridades nos objetivos, ou regiões de interesse, entretanto a diversidade não é restrita ao espaço de objetivos, podendo ser um requisito no espaço de decisões em

algumas aplicações. Neste trabalho a diversidade é considerada apenas no espaço de objetivos.

- **Pertinência**: O tomador de decisões muitas vezes está interessado em sub-regiões do espaço de busca, as quais tornam o processo de decisão e busca mais prático e eficiente. Portanto a convergência e a diversidade das soluções é particularmente requerida nessas regiões de interesse.

Diversos indicadores de qualidade podem ser usados na otimização multiobjetivo, entre eles alguns podem ser destacados.

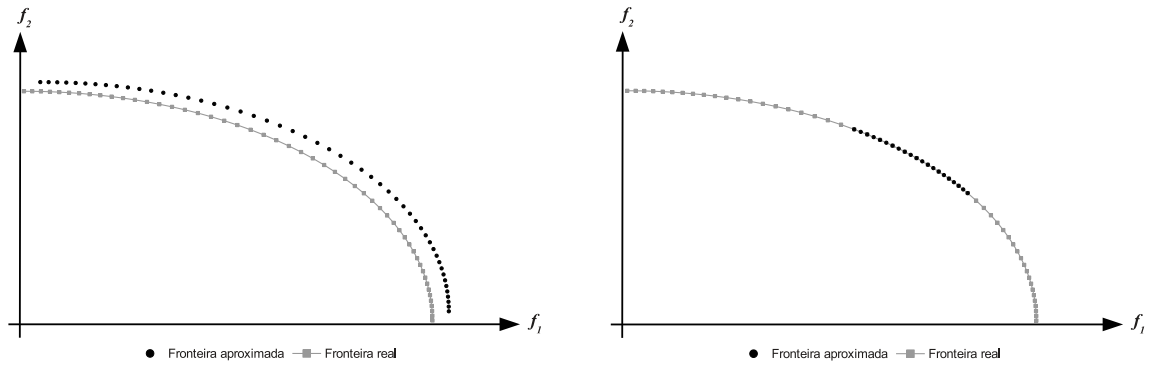
A Distância Geracional *(Generational Distance (GD))* [40] é uma métrica que estima, em média a distância entre os elementos de (PF_{known}) e os de (PF_{true}). A Equação 3.12 define como é feito o cálculo da GD [39, 9].

$$GD \triangleq \frac{(\sum_{i=1}^{n} d_i^p)^{1/p}}{n} \tag{3.12}$$

Na qual n representa o número de vetores em PF_{known}, $p = 2$, e d_i é a distância Euclidiana no espaço de objetivos entre cada membro i de PF_{known} e seu membro mais próximo em PF_{true}. Um valor de zero indica $PF_{known} = PF_{true}$ [39].

A Distância Geracional Invertida *(Inverted Generational Distance (IGD))* mede a menor distância entre cada ponto em PF_{true} e os pontos em PF_{known}. A IGD permite observar se PF_{known} é bem diversificado e se converge para PF_{true}. Seu cálculo é efetuado basicamente como o do GD, entretanto os papeis de PF_{true} e PF_{known} são invertidos na definição do GD [28, 9].

A Figura 3.5 apresenta dois exemplos de aproximação da fronteira de Pareto. Os pontos pretos representam PF_{known}, enquanto os pontos ligados pela linha cinza representam PF_{true} para um problema de minimização biobjetivo. A aproximação disponível na Figura 3.5(a) visivelmente apresenta boa diversidade, entretanto não mostra boa convergência. Seus valores de GD e IGD são 0,00704 e 0,00712 respectivamente. A Figura 3.5(b) mostra um conjunto de soluções pertencentes a uma região da fronteira real, portanto possui convergência ótima, porém não apresenta boa diversidade, seus valores



(a) Fronteira afastada e distribuída (b) Fronteira próxima e agrupada

Figura 3.5: Exemplos de fronteiras aproximadas.

de GD e IGD são 0 e 0,04666 respectivamente.

É muito difícil definir valores de referência considerados bons para estas métricas, uma vez que a qualidade das fronteiras geradas dependem fortemente de diversos fatores, como complexidade do problema, número de objetivos, poder computacional empregado na geração das fronteiras (número de iterações, população, etc.) e a eficiência dos algoritmos em questão, o que normalmente se deseja comparar. Valores de GD obtidos por Tan e Teo [38] estão disponíveis na Tabela 3.1 como exemplos de valores que podem ser encontrados ao se otimizar os problemas estudados neste trabalho.

Tabela 3.1: Valores de GD obtidos em [38]

Probblema	**Objetivos**	**Exemplos de valores de GD**
DTLZ2	**3**	0,001886
	4	0,006066
	5	0,012837
DTLZ4	**3**	0,001758
	4	0,005852
	5	0,015299
DTLZ6	**3**	0,013555
	4	0,065610
	5	0,291700

É importante fazer uma análise conjunta dos indicadores GD e IGD pois o GD mede somente a convergência. Portanto como exemplificado pela Figura 3.5(b) caso as soluções contidas em PF_{known} estejam agrupadas em uma pequena região do espaço de objetivos, porém próximas a PF_{true}, o valor de GD resultante deste conjunto será baixo, o que não significa necessariamente que este conjunto seja bom.

Já o indicador IGD apesar de também medir convergência, pode levar a conclusões erradas sobre esta, uma vez que uma fronteira bem distribuída, porém mais distante de PF_{true}, como a da Figura 3.5(a) pode resultar em melhor valor de IGD que uma fronteira mal distribuída com elementos pertencentes a PF_{true}, como a da Figura 3.5(b). Entretanto o IGD é uma importante métrica de diversidade, pois uma fronteira mal distribuída normalmente resultará em piores valores de IGD.

Portanto através de uma análise conjunta destes indicadores é possível inferir a qualidade de um conjunto de soluções tanto em termos de convergência quanto de diversidade.

O espaçamento médio (*Spacing*) descreve numericamente a distribuição dos vetores através de PF_{known}. Seu valor representa a variância das distâncias de vetores vizinhos em PF_{known}. A Equação 3.13 é usada para o cálculo desta métrica [39, 9].

$$S \triangleq \sqrt{\frac{1}{n-1}\sum_{i+1}^{n}(\overline{d} - d_i)^2}, \tag{3.13}$$

Na qual $d_i = \min_j(|f_1^i(\vec{x}) - f_1^j(\vec{x})| + |f_2^i(\vec{x}) - f_2^j(\vec{x})|)$, i, $j = (1, ..., n)$, $\overline{d}$ é a média de todos os d_i, e n é o número de vetores em PF_{known}. Um valor de zero indica que todos os membros de PF_{known} são equidistantes [39].

O hipervolume [46] é uma métrica amplamente usada para comparar o desempenho de algoritmos de otimização multiobjetivo. Este indicador mede o quanto do espaço de objetivos é dominado por um conjunto de soluções coletivamente. O hipervolume de um conjunto é medido com relação a um ponto de referência, normalmente o pior ponto possível no espaço de objetivos [43], sendo definido matematicamente conforme a Equação 3.14 [9].

$$HV \triangleq \left\{ \bigcup_i vol_i | vec_i \in PF_{known} \right\} \tag{3.14}$$

Na qual vec_i é um vetor não dominado em PF_{known}, e vol_i é o volume entre o ponto de referência e o vetor vec_i.

O cálculo do hipervolume exato é uma tarefa que apresenta um custo computacional muito grande, mesmo os algoritmos mais eficientes atualmente possuem tempos de execução exponenciais conforme o número de objetivos, o que restringe o uso do hipervolume exato a problemas com menos de dez objetivos [3].

Embora existam métodos capazes de calcular o hipervolume aproximado mais rapidamente, por exemplo usando o método de Monte Carlo [3], uma margem de erro é introduzida. Para este tipo de aproximação é crucial estabelecer pequenos espaços para amostragem, entretanto a normalização das fronteiras obtidas não é uma opção viável, uma vez que dado seu formato, ou pontos extremos, esta pode passar a se localizar em regiões diferentes do hipercubo normalizado, aumentando muito a margem de erro. Além disso, conforme apresentado em [28, 12], fronteiras com melhores valores de hipervolume nem sempre apresentam melhor desempenho.

Para detectar se existem diferenças significativas entre os resultados obtidos, é utilizado o teste estatístico de Friedman. O teste de Friedman é um teste estatístico não paramétrico que pode ser usado para detectar diferenças entre resultados de experimentos usando diferentes algoritmos [16].

Neste trabalho a implementação do teste de Friedman e do teste *post-hoc* utilizadas estão disponíveis no ambiente R para computação estatística [33].

Cada uma das trinta execuções independentes de cada algoritmo comparado resulta em um valor para cada métrica utilizada. Estes trinta valores de cada métrica são introduzidos no teste de Friedman, e o algoritmo que apresenta a melhor média das trinta execuções é considerado o melhor algoritmo, em seguida, todos os algoritmos cujos resultados não apresentam diferença estatística em relação ao melhor são considerados os melhores algoritmos e incluídos nas tabelas do teste de Friedman.

CAPÍTULO 4

SMPSO COM CONTROLE AUTOMÁTICO DA ÁREA DE DOMINÂNCIA

Este capítulo apresenta um novo algoritmo chamado SMPSO com controle automático da área de dominância *(Self-Controlling Dominance SMPSO (SCDAS-SMPSO))*, baseado no CDAS-SMPSO proposto em [12]. O CDAS-SMPSO aplica a técnica de controle da área de dominância CDAS [35] ao algoritmo SMPSO [30]. Neste novo algoritmo, a técnica de controle da área de dominância é substituída pela mais recente S-CDAS [36]. Parte do trabalho apresentado neste capítulo foi publicado no artigo [8].

A Seção 4.1 apresenta as duas técnicas de controle da área de dominância das soluções usadas neste capítulo. A Seção 4.2 traz um estudo comparativo entre os resultados obtidos usando ambas as técnicas de controle da área de dominância, além de compará-las com o algoritmo SMPSO original, e a Seção 4.3 traz uma análise dos resultados apresentados.

4.1 Técnicas de controle da área de dominância aplicadas ao MOPSO

A primeira aplicação de uma técnica de controle da área de dominância no MOPSO para muitos objetivos foi feita em [12] dando origem ao algoritmo CDAS-SMPSO, no qual o nível de contração ou expansão da área de dominância das soluções do repositório é modificado usando um parâmetro S definido pelo usuário. Com esta alteração, soluções originalmente não dominadas podem passar a ser dominadas por outras. A Equação 4.1 é responsável pelo cálculo da nova área de dominância.

$$f'_i(\vec{x}) = \frac{r \cdot sen(\omega_i - S\pi)}{sen(S\pi)} \tag{4.1}$$

Na qual $\vec{x}$ é uma solução no espaço de busca, $\vec{f}(\vec{x})$ é seu vetor objetivo e r é a norma de $\vec{f}(\vec{x})$. ω_i é o ângulo de inclinação entre $f_i(\vec{x})$ e $f'_i(\vec{x})$. Na Equação 4.1 é feita a operação trigonométrica que translada cada valor objetivo $f_i(\vec{x})$ em um novo valor objetivo $f'_i(\vec{x})$.

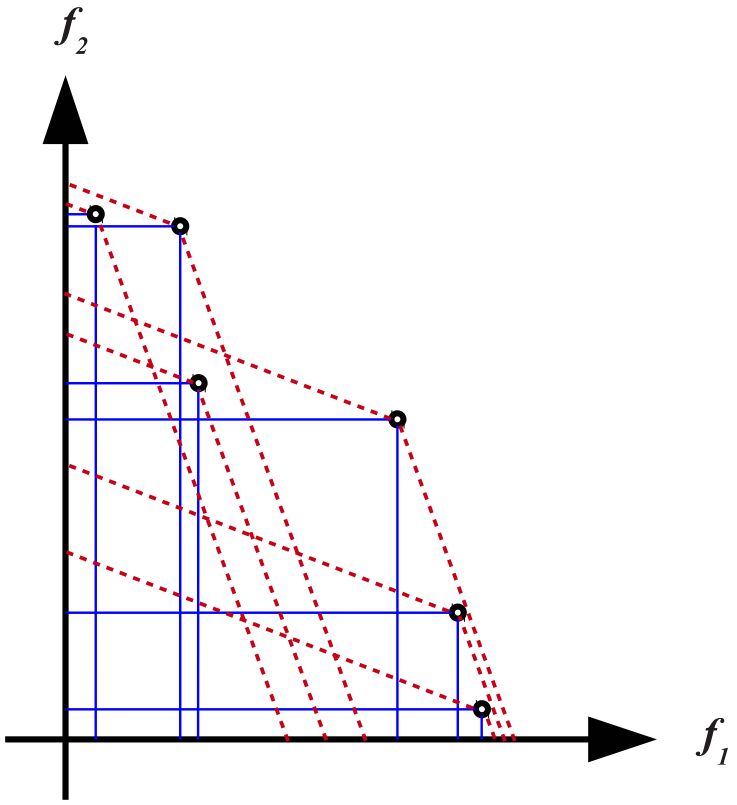


Figura 4.1: Representação do método CDAS

A Figura 4.1 mostra o funcionamento do método CDAS para um problema de maximização, no qual as linhas azuis representam a área de dominância original das soluções, e as linhas vermelhas pontilhadas mostram a área de dominância modificada pelo CDAS. Todas as soluções dentro da área de dominância formada abaixo e à esquerda de uma solução são dominadas por ela. Nesta figura é possível notar que com a dominância tradicional todas as seis soluções são não dominadas entre si, entretanto ao alterar a área de dominância com o CDAS, apenas duas delas se tornam não dominadas.

O CDAS-SMPSO[1] introduz o método CDAS durante o processo de atualização do

[1]Para o correto funcionamento do método CDAS, todos os objetivos do problema devem estar na mesma escala.

arquivo externo no algoritmo SMPSO, neste passo, caso $S = 0.5$, então $f_i'(\vec{x}) = f_i(\vec{x})$, e a relação de dominância se mantém inalterada. Se a área de dominância é alterada com um valor $S < 0.5$, então $f_i'(\vec{x}) > f_i(\vec{x})$, e menos soluções se tornam não dominadas, portanto o algoritmo tende a convergir para uma área menor do espaço de busca. Se $S > 0.5$, então $f_i'(\vec{x}) < f_i(\vec{x})$, e a relação de dominância é relaxada, portanto soluções que eram dominadas antes, se tornam não dominadas e influenciam as outras partículas do enxame [12].

No algoritmo SCDAS-SMPSO a nova técnica de controle da área de dominância é incorporada ao SMPSO, sua principal vantagem é o fato de ela controlar automaticamente a área de dominância para cada solução com base nos valores das soluções extremas da fronteira a cada iteração, sem depender de parâmetros externos.

Para o cálculo da nova área de dominância, primeiramente a origem $\vec{O} = (O_1, ..., O_m)$ é movida no espaço objetivo para $O_i = f_i^{min} - \delta$, com $i = (1, ..., m)$, f_i^{min} é o menor valor da i-ésima função objetivo da fronteira de Pareto, e δ é uma constante de valor muito baixo.

Em seguida cria-se um conjunto de vetores de referência $\vec{\mathcal{L}} = \{\vec{p}_1, ..., \vec{p}_m\}$, no qual $\vec{p}_i = (O_1, O_2, ..., f_i^{max} - \delta, ..., O_m)$, e f_i^{max} denota o maior valor da i-ésima função objetivo que é derivada da solução extrema E_i que faz parte da fronteira de Pareto.

Após este passo, para todas as soluções da fronteira calcula-se $\varphi(\vec{x}) = (\varphi_1(\vec{x}), ..., \varphi_m(\vec{x}))$ através da Equação 4.2.

$$\varphi_i(\vec{x}) = sen^{-1}\left\{\frac{r(\vec{x}) \cdot sen(\omega_i(\vec{x}))}{l_i(\vec{x})}\right\} \qquad (i = 1, ..., m) \tag{4.2}$$

Na qual $l_i(\vec{x})$ é a distância Euclidiana entre a solução $\vec{x}$ e o vetor de referência $\vec{p}_i$. Este cálculo determina o equivalente ao valor $(S.\pi)$ do CDAS, entretanto aqui é feito através da Equação 4.2 para cada solução da fronteira.

Em seguida os valores objetivo de todas as outras soluções $\vec{y}$ da fronteira são modificados de acordo com a Equação 4.3.

$$f_i'(\vec{y}) = \frac{r(\vec{y}) \cdot sen(\omega_i(\vec{y}) + \varphi_i(\vec{x}))}{sen(\varphi_i(\vec{x}))} \qquad (i = 1, ..., m) \tag{4.3}$$

A Equação 4.3 executa a operação trigonométrica que translada cada valor objetivo $f_i(\vec{x})$ em um novo valor objetivo $f_i'(\vec{x})$. Após isto, devem ser checadas todas as relações de dominância entre a solução $\vec{x}$ e as outras soluções $\vec{y}$ para determinar o novo conjunto de soluções não dominadas.

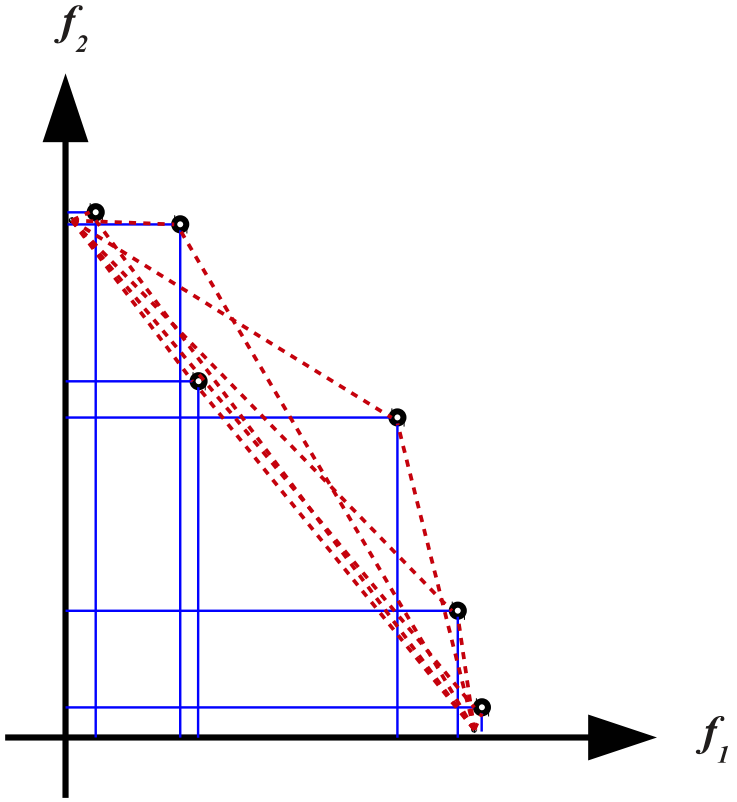


Figura 4.2: Representação do método S-CDAS

Esta alteração na área de dominância é representada pela Figura 4.2 para um problema de maximização, no qual as linhas azuis representam a área de dominância original das soluções, e as linhas vermelhas pontilhadas mostram a área de dominância modificada pelo CDAS. Todas as soluções dentro da área de dominância formada abaixo e à esquerda de uma solução são dominadas por ela. A partir desta figura é possível perceber que apesar de o método S-CDAS eliminar algumas soluções, aumentando a pressão de seleção, ele

consegue uma melhor diversidade, mantendo as soluções extremas.

Originalmente a técnica S-CDAS foi usada para reclassificar as soluções em fronteiras de diferentes níveis no algoritmo NSGA-II, entretanto, na metaheurística MOPSO, a fronteira não é dividida em níveis, portanto no SCDAS-SMPSO essa técnica foi usada para reduzir o número de soluções não dominadas e assim aumentar a pressão de seleção. Assim como o CDAS, o S-CDAS também necessita que os objetivos do problema estejam na mesma escala.

4.2 Experimentos

Esta seção descreve os experimentos realizados para comparar as técnicas de controle da área de dominância CDAS e S-CDAS, além de comparar ambas com o algoritmo SMPSO original. Estes experimentos foram realizados para verificar os impactos no desempenho do MOPSO conforme o número de objetivos aumenta, observando qual algoritmo obtém a melhor convergência em relação à fronteira real, e a melhor diversidade de soluções cobrindo esta.

Nestes experimentos foram usados os problemas DTLZ2 e o DTLZ6. O DLTZ2 foi usado por ser um dos problemas mais simples da família DTLZ, portanto pertinente para avaliar a escalabilidade do algoritmo. O problema DTLZ6 foi usado por ser um problema que apresenta dificuldade na convergência por apresentar várias fronteiras sub-ótimas, resultando em grande chance de o otimizador ficar preso em regiões sub-ótimas.

O número de objetivos foi variado entre dois e trinta. O número de iterações foi fixado em 100 e tanto a população quanto o tamanho do repositório foram mantidos em 250. Os resultados estão apresentados através de gráficos, neles, uma curva representa o algoritmo SMPSO usando o método S-CDAS, outra representa o SMPSO sem alteração na área de dominância ($S = 0.5$), e as outras cinco curvas representam os diferentes valores do parâmetro S para o SMPSO usando o método CDAS. Cada ponto representa a média dos valores obtidos com o indicador de qualidade em cada uma das trinta execuções efetuadas para cada número de objetivos.

Observe que nestes experimentos os valores de GD e IGD chegam à ordem de 10^{-5} o

que pode ser considerado desprezível mesmo para o pior algoritmo. Para decidir se existem ou não diferenças estatisticamente significantes entre os algoritmos os testes estatísticos foram aplicados e seus resultados encontram-se nas Tabelas 4.1, 4.2 e 4.3.

Primeiramente é analisada a convergência das diferentes abordagens de modificação de relação de dominância. Para analisar a convergência a métrica mais relevante é a GD.

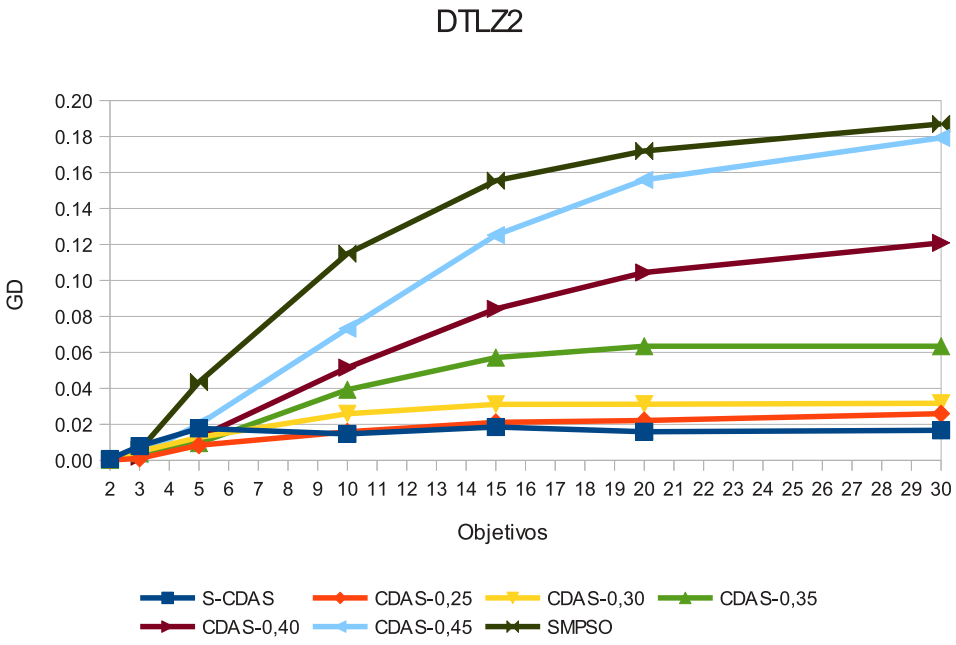


Figura 4.3: Valores de GD ao otimizar o problema DTLZ2

Os gráficos das Figuras 4.3 e 4.4 representam as médias dos valores de GD para os problemas DTLZ2 e DTLZ6 respectivamente. Para o problema DTLZ2 não é possível notar diferenças visíveis para dois e três objetivos, já para o problema DTLZ6, o CDAS com valores S= 0.25, 0.3 e 0.35 não obtém bons resultados para dois objetivos, para três objetivos somente o CDAS com $S = 0.25$ apresenta resultados visivelmente piores por obter convergência prematura para uma fronteira sub-ótima. Para dez ou mais objetivos, no problema DTLZ2 a técnica S-CDAS, e a CDAS com valores $S = 0.25$ e $S = 0.3$ apresentam os melhores desempenhos. Enquanto no problema DTLZ6 os melhores resultados para cinco ou mais objetivos são obtidos com a técnica CDAS com valores $S = 0.3$ e $S = 0.35$. Estes resultados sugerem que algoritmos usando alguma técnica de controle da área de dominância apresentam convergência superior se comparados com o SMPSO

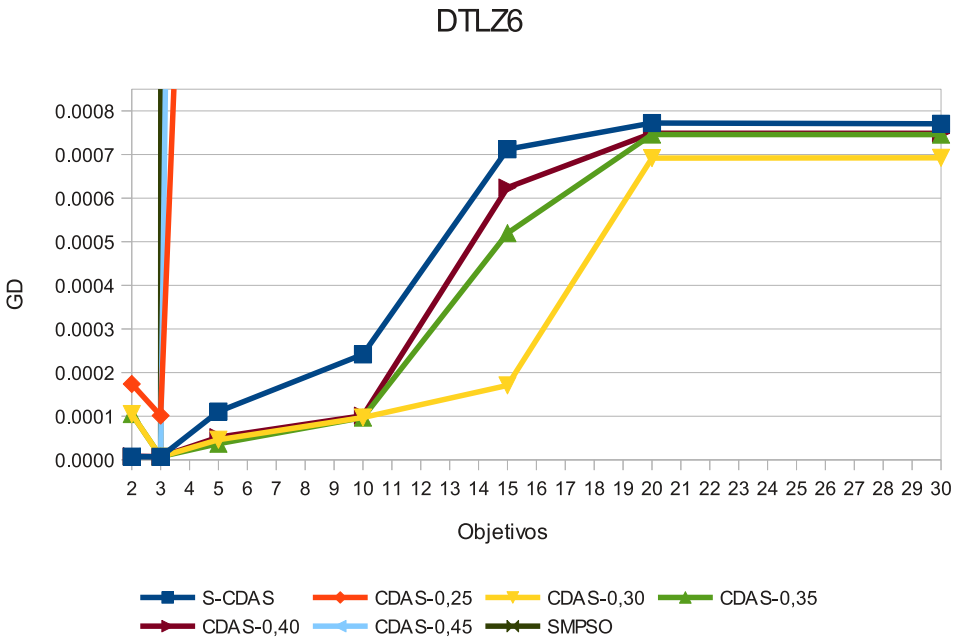


Figura 4.4: Valores de GD ao otimizar o problema DTLZ6

original, especialmente em muitos objetivos.

Tabela 4.1: Melhor configuração de controle da área de dominância de acordo com o GD para os problemas DTLZ2 e DTLZ4 de acordo com o teste de Friedman.

Prob	**Obj**	**Melhores esquemas de controle da área de dominância**
		GD
DTLZ2	**2**	CDAS-0.25, CDAS-0.3, CDAS-0.35
	3	CDAS-0.25, CDAS-0.4
	5	CDAS-0.25, CDAS-0.3, CDAS-0.35, CDAS-0.4
	10	S-CDAS, CDAS-0.25, CDAS-0.3
	15	S-CDAS, CDAS-0.25, CDAS-0.3
	20	S-CDAS, CDAS-0.25, CDAS-0.3
	30	S-CDAS, CDAS-0.25, CDAS-0.3
DTLZ6	**2**	S-CDAS, CDAS-0.45, SMPSO
	3	S-CDAS, CDAS-0.3, CDAS-0.35, CDAS-0.4, CDAS-0.45, SMPSO
	5	CDAS-0.3, CDAS-0.35, CDAS-0.4
	10	CDAS-0.3, CDAS-0.35, CDAS-0.4
	15	CDAS-0.3, CDAS-0.35
	20	CDAS-0.3, CDAS-0.35
	30	CDAS-0.3, CDAS-0.35

Nos testes estatísticos do GD, disponíveis na Tabela 4.1, para poucos objetivos, todas as técnicas de controle da área de dominância apresentaram bons resultados em alguma combinação de problemas e objetivos, indicando não possuírem diferenças significativas, e o melhor desempenho em ambos os problemas incluem a técnica CDAS com parâmetro

$S = 0.3$.

Os conjuntos de aproximação obtidos com CDAS com baixos valores de S, especialmente 0.25 apresentam poucas soluções, este comportamento normalmente gera bons resultados, mas restritos a uma área muito pequena do espaço de objetivos de acordo com [12].

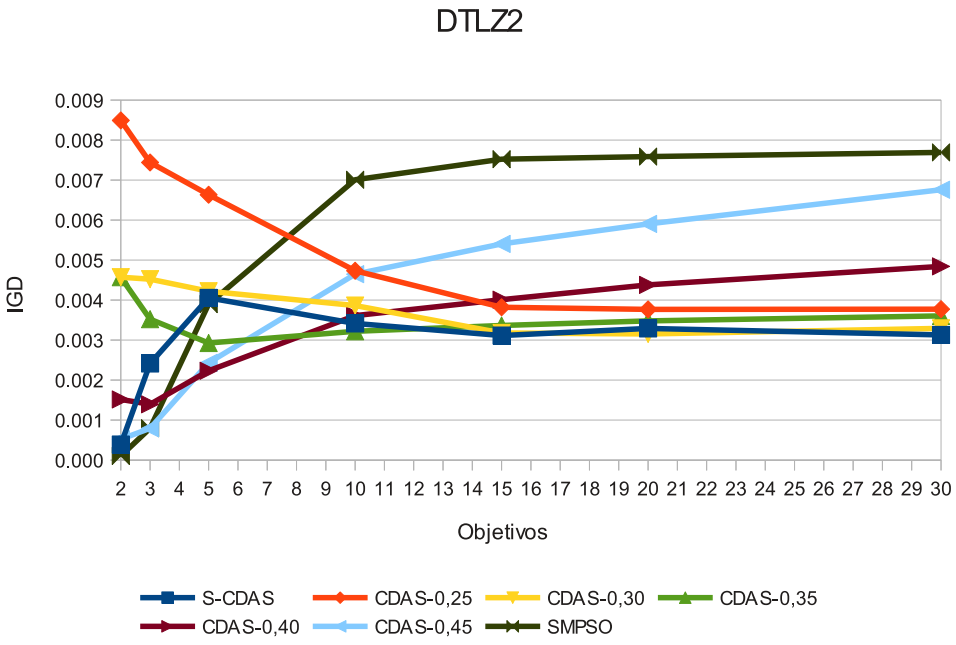


Figura 4.5: Valores de IGD ao otimizar o problema DTLZ2

Analisando a diversidade das abordagens, as métricas IGD e Spacing são as mais significativas. Para o IGD, apenas o S-CDAS apresentou o melhor desempenho em ambos os problemas, como pode ser visto nas Figuras 4.5 e 4.6, que contém os gráficos de IGD para os problemas DTLZ2 e DTLZ6 respectivamente, e na Tabela 4.2, contendo os testes estatísticos.

Estes resultados indicam que a técnica S-CDAS alcança uma convergência similar à CDAS no problema DTLZ2, entretanto sua diversidade é superior em ambos os problemas. Estes bons valores de IGD indicam que o algoritmo não gera soluções muito aglomeradas em uma pequena área do espaço de objetivos, como a CDAS com pequenos valores de S.

Na métrica Spacing, cujos resultados são mostrados nas Figuras 4.7 e 4.8 para os problemas DTLZ2 e DTLZ6 respectivamente, e na Tabela 4.3, são obtidos melhores de-

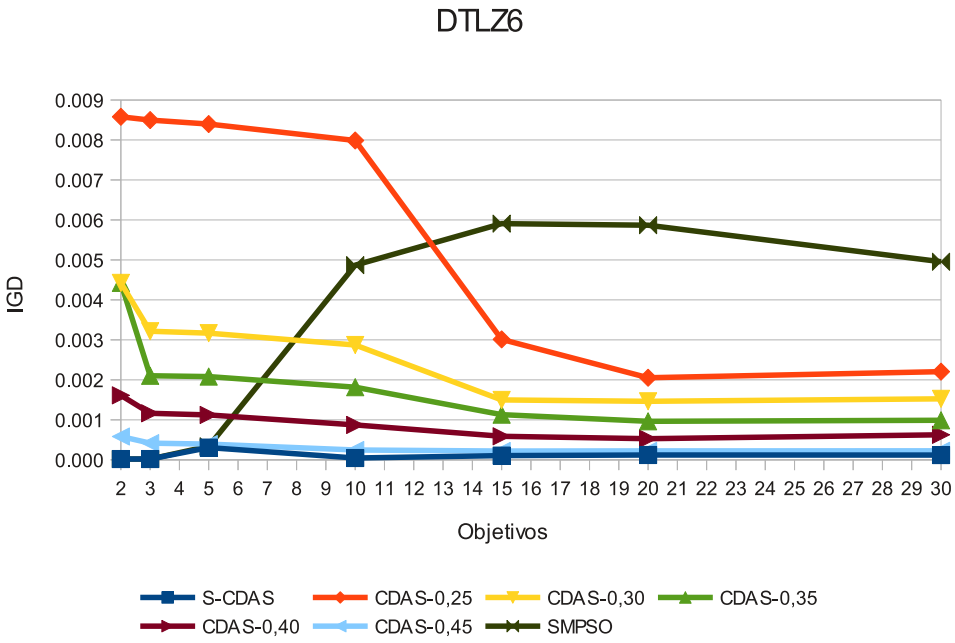


Figura 4.6: Valores de IGD ao otimizar o problema DTLZ6

Tabela 4.2: Melhor configuração de controle da área de dominância de acordo com o IGD para os problemas DTLZ2 e DTLZ4 de acordo com o teste de Friedman.

Prob	**Obj**	**Melhores esquemas de controle da área de dominância**
		IGD
DTLZ2	**2**	S-CDAS, SMPSO
	3	S-CDAS, CDAS-0.45, SMPSO
	5	CDAS-0.35, CDAS-0.4, CDAS-0.45
	10	S-CDAS, CDAS-0.35, CDAS-0.4
	15	S-CDAS, CDAS-0.25, CDAS-0.3, CDAS-0.35
	20	S-CDAS, CDAS-0.25, CDAS-0.3, CDAS-0.35
	30	S-CDAS, CDAS-0.25, CDAS-0.3, CDAS-0.35
DTLZ6	**2**	S-CDAS, CDAS-0.45, SMPSO
	3	S-CDAS, CDAS-0.45, SMPSO
	5	S-CDAS, CDAS-0.45, SMPSO
	10	S-CDAS, CDAS-0.45
	15	S-CDAS, CDAS-0.45
	20	S-CDAS, CDAS-0.45
	30	S-CDAS, CDAS-0.45

sempenhos usando o CDAS com pequenos valores de S, especialmente 0.25, 0.3 e 0.35 para ambos os problemas, enquanto o S-CDAS não obtém bons resultados para o problema DTLZ4 com menos de dez objetivos, nem para o problema DTLZ6. Este comportamento ocorre principalmente devido à característica do CDAS com baixos valores de S de agrupar as soluções em pequenas áreas do espaço de objetivos [12], isto faz a distância média

entre as soluções muito pequena e parecida.

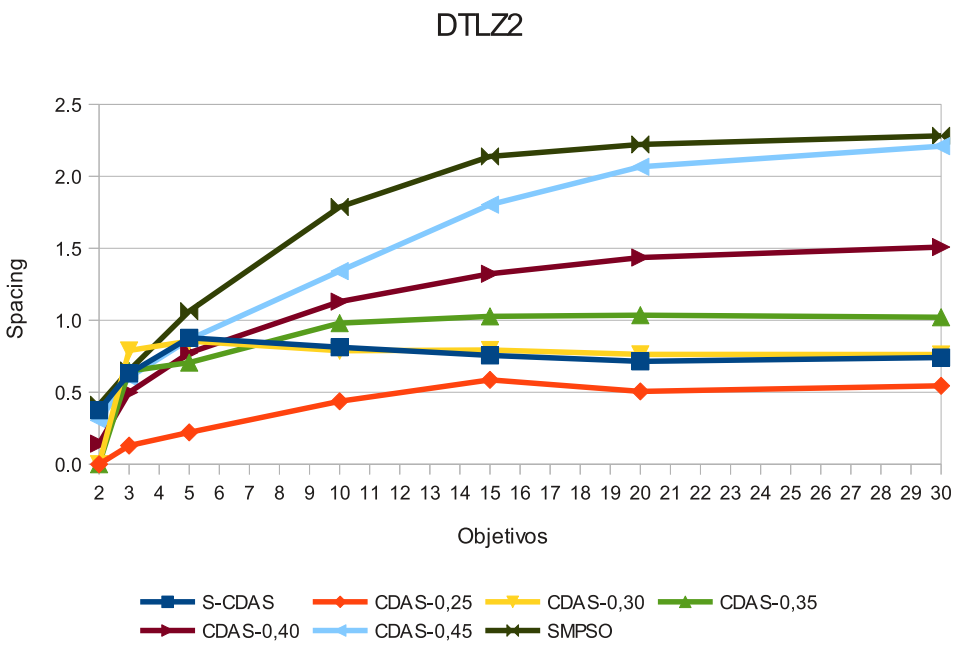


Figura 4.7: Valores de Spacing ao otimizar o problema DTLZ2

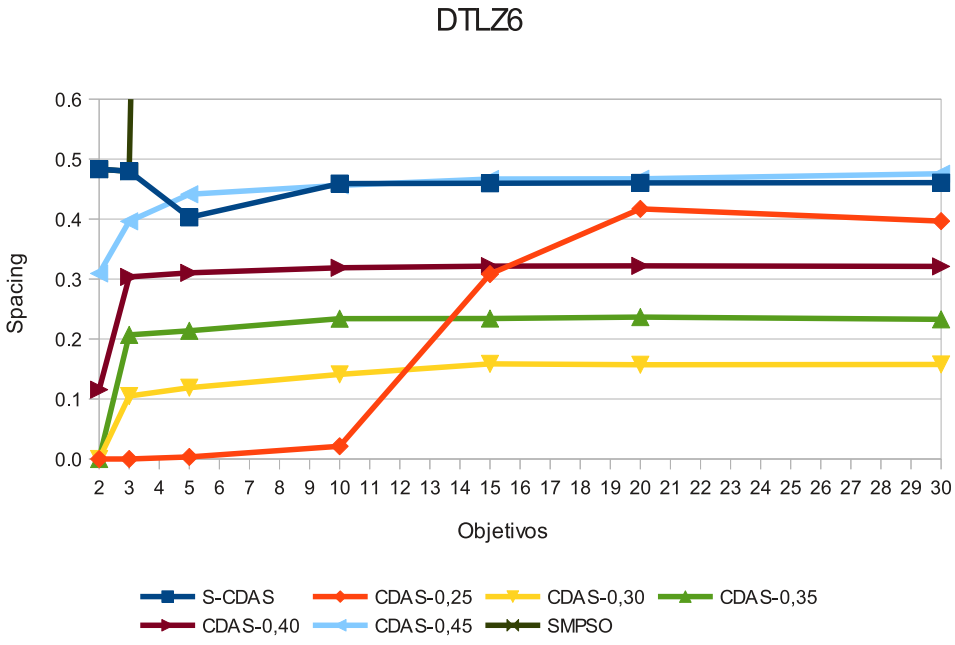


Figura 4.8: Valores de Spacing ao otimizar o problema DTLZ6

Outro fator que influencia negativamente o Spacing é o número de soluções compondo a fronteira. A técnica CDAS com baixos valores de S seleciona poucas soluções para a fronteira conforme observado em [12], isto facilita com que a distribuição destas soluções

Tabela 4.3: Melhor configuração de controle da área de dominância de acordo com o Spacing para os problemas DTLZ2 e DTLZ4 de acordo com o teste de Friedman.

Prob	**Obj**	**Melhores esquemas de controle da área de dominância**
		Spacing
DTLZ2	**2**	CDAS-0.25, CDAS-0.3, CDAS-0.35
	3	CDAS-0.25, CDAS-0.4
	5	CDAS-0.25, CDAS-0.35
	10	S-CDAS, CDAS-0.25, CDAS-0.3
	15	S-CDAS, CDAS-0.25, CDAS-0.3
	20	S-CDAS, CDAS-0.25, CDAS-0.3
	30	S-CDAS, CDAS-0.25, CDAS-0.3
DTLZ6	**2**	CDAS-0.25, CDAS-0.3, CDAS-0.35
	3	CDAS-0.25, CDAS-0.3
	5	CDAS-0.25, CDAS-0.3
	10	CDAS-0.25, CDAS-0.3
	15	CDAS-0.3, CDAS-0.35
	20	CDAS-0.3, CDAS-0.35
	30	CDAS-0.3, CDAS-0.35

tenha uma distância mais parecida. A S-CDAS, por outro lado, seleciona um maior número de soluções para compor o arquivo externo, entregando, no final uma fronteira mais diversificada como demonstrado pelos melhores valores de IGD, entretanto este maior número de soluções prejudica o espaçamento entre estas, gerando piores resultados.

4.3 Análise dos resultados

Analisando os dados dos experimentos, é possível concluir que, como mostrado em estudos anteriores [12], a utilização de alguma técnica de controle da área de dominância causa uma melhora de desempenho ao comparar os resultados com os do SMPSO original, especialmente para grandes números de objetivos, quando o desempenho do SMPSO se deteriora muito.

Ao comparar os resultados de ambos os métodos de controle da área de dominância, o S-CDAS apresenta uma convergência tão boa quanto o CDAS com baixos valores de S, mas gera conjuntos de soluções com melhor diversidade que o CDAS com maiores valores de S em muitos casos. Além do ganho em desempenho, o S-CDAS não requer o parâmetro externo S, se tornando mais fácil de usar.

A ideia original do S-CDAS era expandir a fronteira gerada para cobrir uma maior

porção do espaço de objetivos apresentando maior diversidade. A técnica atingiu seu objetivo no MOPSO, apresentando um desempenho geral melhor que o do CDAS.

CAPÍTULO 5

A INFLUÊNCIA DE TÉCNICAS DE CONTROLE DA ÁREA DE DOMINÂNCIA E MÉTODOS DE ESCOLHA DE LÍDERES

Este capítulo investiga a influência de métodos de escolha de líder e técnicas de controle da área de dominância no desempenho da metaheurística MOPSO ao otimizar problemas com muitos objetivos. Para isto, primeiramente a Seção 5.1 apresenta diversos métodos de escolha de líder disponíveis na literatura, além de apresentar um novo. Estes métodos foram usados em outros trabalhos para otimizar problemas com dois ou três objetivos, enquanto neste trabalho eles são comparados ao resolver problemas com muitos objetivos.

Conforme o estudo apresentado no Capítulo 4 todos os resultados obtidos usando uma das técnicas de controle da área de dominância são tão bons quanto, ou melhores que os resultados obtidos usando o SMPSO sem controle da área de dominância. Portanto nesta seção os métodos de líder são comparados utilizando os métodos CDAS e S-CDAS que apresentam bons resultados em cenários distintos.

A Seção 5.2 apresenta um estudo empírico utilizando duas técnicas de controle da área de dominância e seis métodos de escolha de líderes, e a Seção 5.3 apresenta uma análise dos resultados obtidos nos experimentos. Parte do trabalho apresentado neste capítulo foi publicado nos artigos [7] e [24].

5.1 Métodos de escolha do líder

Esta seção apresenta uma revisão sobre quatro métodos de escolha de líder que obtiveram bons resultados na literatura além de propor o novo método Oposto cujo objetivo é introduzir diversidade à busca. A abordagem aleatória, que é comumente utilizada como parâmetro de comparação também é apresentada.

Os métodos apresentados são: distância de agrupamento (Seção 5.1.1), WSum (soma ponderada) (Seção 5.1.2), NWSum (Seção 5.1.3) e Sigma (Seção 5.1.4). O novo método Oposto é apresentado na Seção 5.1.5, e o método Aleatório, utilizado como referência de desempenho para os demais é mostrado na Seção 5.1.6. A seguir cada método é detalhado.

5.1.1 Distância de agrupamento

A distância de agrupamento (*Crowding Distance (CD)*), proposta em [13] é um estimador de diversidade que foi extensivamente empregado em algoritmos evolutivos multiobjetivo para promover diversidade [31]. Esta métrica é usada para obter uma estimativa da densidade de soluções em volta de um ponto particular na população. Para isto é usada a distancia média dos dois pontos vizinhos a este em cada um dos objetivos, este valor $i_{distancia}$ serve como uma estimativa do tamanho do maior cuboide abrangendo o ponto $\vec{i}$ sem incluir qualquer outro ponto da população [13].

O cálculo da distância de agrupamento é feito conforme o Algoritmo 3

Algoritmo 3: Pseudocódigo do cálculo da distância de agrupamento

```
l = |I|
para Cada i faça
  | I[i]_distancia = 0
fim para
para Cada m faça
  | I =classifique(I, m)
  | I[1]_distancia = I[l]_distancia = ∞
  | para i = 2 até (l − 1) faça
  |   | I[i]_distancia = I[i]_distancia + (I[i + 1] · m − I[i − 1] · m)
  | fim para
fim para
```

No Algoritmo 3, $\mathcal{I}[i] \cdot m$ representa o m-ésimo valor objetivo do i-ésimo indivíduo no conjunto $\mathcal{I}$. l representa o número de soluções em $\mathcal{I}$, e $i = (1, ..., n)$.

O algoritmo inicia com l armazenando o número de soluções contidas em $\mathcal{I}$, em seguida a $i_{distancia}$ de cada partícula é inicializada com 0. No próximo passo cada objetivo é selecionado por vez e a população é classificada de acordo com este objetivo. As soluções ocupando a primeira e a ultima posições (soluções extremas do eixo) recebem o valor ∞

para $i_{distancia}$. Em seguida o valor de $i_{distancia}$ para os outros pontos é calculado, a cada objetivo a $i_{distancia}$ é acrescida da diferença dos valores objetivo dos vizinhos de i.

Para usar a distância de agrupamento como método de seleção de líderes, dois candidatos a líder são escolhidos aleatoriamente do repositório e o que apresentar a maior distância de agrupamento está localizado numa região menos povoada, portanto é selecionado. Esta é a técnica de seleção de líder originalmente usada no SMPSO.

5.1.2 WSum (soma ponderada)

O método da soma ponderada (WSum) foi proposto em [4], e consiste em uma soma ponderada dos valores objetivo. O valor é calculado conforme a Equação 5.1. Na qual $f_j(\vec{x})$ é o j-ésimo valor objetivo da partícula $\vec{x}$, e $f_j(\vec{l})$ é o j-ésimo valor objetivo do líder $\vec{l}$ e m é o número de objetivos.

$$F = \sum_{j=0}^{m} \frac{f_j(\vec{x})}{\sum_{k=0}^{m} f_k(\vec{x})} f_j(\vec{l}) \tag{5.1}$$

O líder que obtiver a menor soma ponderada é selecionado para a partícula $\vec{x}$. Neste método o líder selecionado será o mais próximo possível do eixo oposto ao da partícula $\vec{x}$, com isso, este método introduz diversidade na busca, seu comportamento geral é demonstrado na Figura 5.1.

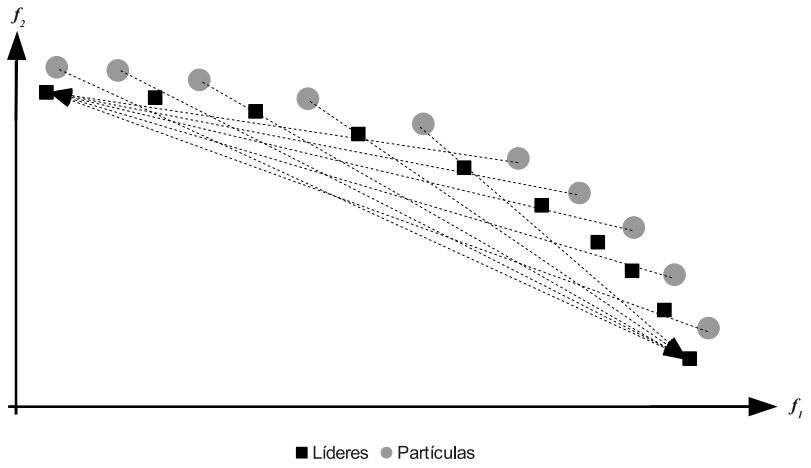


Figura 5.1: Ilustração do Método WSum

Nesta figura, os quadrados pretos representam candidatos a líder contidos no repo-

sitório, os círculos cinza representam partículas da busca, e as linhas tracejadas apontam qual líder uma partícula escolheria durante a busca para um problema de minimização biobjetivo. É possível notar que neste exemplo as partículas escolhem líderes próximos aos eixos, entretanto distantes da sua posição atual.

5.1.3 NWSum

O método NWSum, proposto em [31], é uma variante do método WSum. No WSum a partícula com a menor soma ponderada é escolhida como guia, entretanto no NWSum, a partícula com a maior soma ponderada é escolhida.

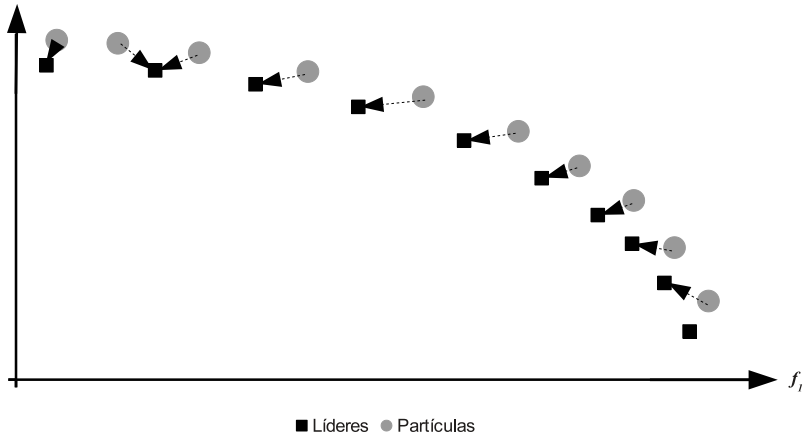


Figura 5.2: Ilustração do Método NWSum

A Figura 5.2 mostra o comportamento do método NWSum, na qual os quadrados pretos representam candidatos a líder contidos no repositório, os círculos cinza representam partículas da busca, e as linhas tracejadas apontam qual líder uma partícula escolheria durante a busca para um problema de minimização biobjetivo. Neste exemplo cada partícula seleciona um líder próximo, este comportamento tende a melhorar a convergência.

5.1.4 Sigma

No método Sigma, o líder é escolhido de acordo com sua distância Sigma. Para cada solução no espaço de objetivos $\vec{f}(\vec{x})$, um vetor $\vec{\sigma}$ é atribuído. Em alguns casos o vetor Sigma pode possuir posições negativas sem comprometer seu desempenho, pois são cal-

culadas as distâncias entre vetores Sigma. Para um problema de dois objetivos, o vetor Sigma é definido da seguinte maneira:

$$\vec{\sigma} = \frac{f_1(\vec{x})^2 - f_2(\vec{x})^2}{f_1(\vec{x})^2 + f_2(\vec{x})^2} \tag{5.2}$$

Para problemas com mais de dois objetivos, o vetor Sigma é um vetor com $\binom{m}{2}$ elementos, representando as combinações da Equação 5.2 para todos os objetivos. O líder para uma partícula do enxame é a solução do repositório com a menor distância euclidiana entre seu vetor Sigma e o vetor Sigma da partícula [29].

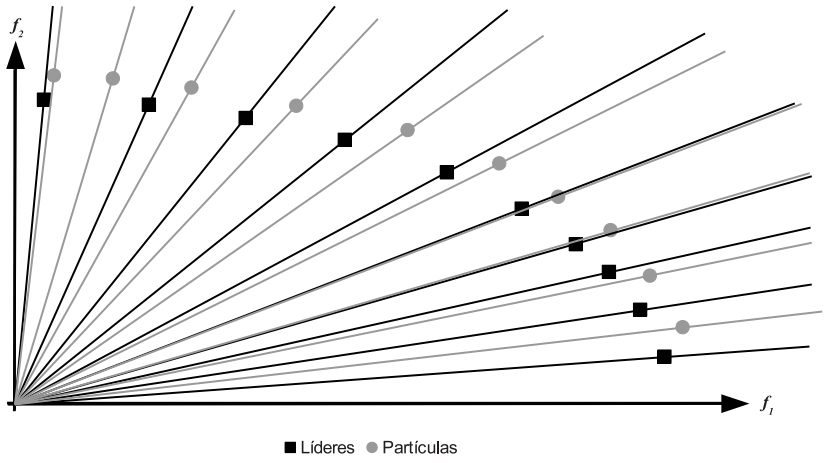


Figura 5.3: Ilustração do Método Sigma [29]

A Figura 5.3 mostra o comportamento do método Sigma, na qual os quadrados pretos representam candidatos a líder contidos no repositório e os círculos cinza representam partículas da busca em um problema de minimização biobjetivo. Tanto para os candidatos a líder quanto para as partículas são calculados vetores Sigma, representados pelas linhas pretas e cinzas respectivamente, estes vetores partem da origem e cruzam as soluções. Cada partícula, neste caso, seleciona o líder que tiver o vetor Sigma mais próximo do seu.

5.1.5 Oposto

A ideia do método Oposto é introduzir diversidade à busca do algoritmo MOPSO movendo as partículas para diferentes áreas do espaço de objetivos. Basicamente, dada uma partícula $\vec{x}$, com vetor objetivo $\vec{f}(\vec{x})$, o novo líder de $\vec{x}$ é uma solução no repositório com

um vetor objetivo em uma região oposta a $\vec{x}$, por exemplo, a partícula escolhe um líder que tem bons valores nos objetivos em que $\vec{x}$ não tem bons resultados. Então, se $\vec{x}$ tem o pior valor para um objetivo j, $\vec{x}$ irá selecionar como líder uma solução com os melhores valores neste objetivo. Entretanto, se a partícula tem valores similares para todos os objetivos, ela irá selecionar como líder uma solução que apresenta valores similares para os objetivos também.

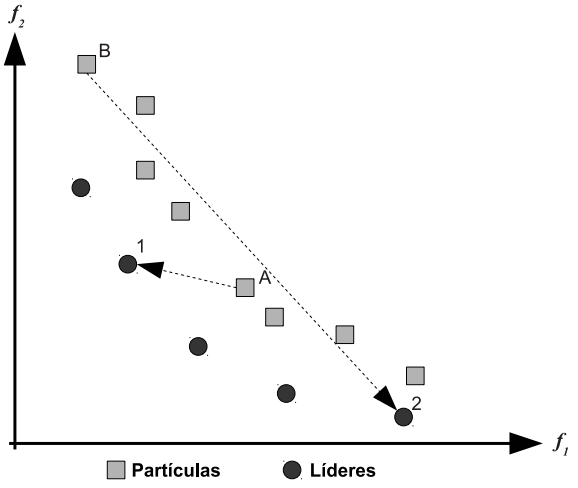


Figura 5.4: Ilustração do Método Oposto

A Figura 5.4 mostra um exemplo do método Oposto, para um problema de maximização em um espaço de objetivos bidimensional. A **Solução B** tem bons valores no objetivo 1, mas valores ruins no objetivo 2, nesta situação a Solução B irá selecionar como líder o **Líder 2**, que está localizado na região oposta no espaço de objetivos. O mesmo comportamento é observado para **Solução A**, que tem o objetivo 2 levemente melhor que o objetivo 1, então ela escolherá o **Líder 1** que tem o objetivo 1 levemente melhor que o objetivo 2. Com este comportamento as partículas visitarão regiões diferentes do espaço de busca, introduzindo assim mais diversidade à busca.

Para identificar a região em que $\vec{x}$ está localizada, é usado o vetor $\vec{f}_{avg}(\vec{x})$, no qual:

$$\vec{f}_{avg}(\vec{x}) = (f_1(\vec{x}) - \overline{f}(\vec{x}), f_2(\vec{x}) - \overline{f}(\vec{x}), ..., f_m(\vec{x}) - \overline{f}(\vec{x})) \tag{5.3}$$

$\overline{f}(\vec{x})$ é o valor médio do vetor $\vec{f}(\vec{x})$. Com o vetor $\vec{f}_{avg}(\vec{x})$ é possível identificar quais valores

objetivo estão muito abaixo ou acima da média, e identificar a região em que a partícula está localizada. Então, primeiro é calculado o $\vec{f_{avg}}(\vec{x})$ para todas as soluções, incluindo os líderes. Depois, o líder com o $\vec{f_{avg}}$ mais diferente de $\vec{f_{avg}}(\vec{x})$ é selecionado para a partícula $\vec{x}$. Para definir esta diferença $\vec{f_{avg}}(\vec{x})$ é adicionado ao $\vec{f_{avg}}$ de cada líder, quanto menor for a norma do vetor resultante, maior é a diferença entre os vetores objetivo.

5.1.6 Aleatório

A abordagem aleatória, é a mais simples e computacionalmente mais barata, foi primeiramente proposta em [10], e seleciona aleatoriamente um líder no repositório para cada partícula.

Este método beneficia a diversidade em uma fronteira bem distribuída, mas em situações em que os líderes estão agrupados em uma região, a probabilidade de escolher um líder de uma zona densamente populada é maior, isto pode resultar em uma perda de diversidade indesejada [31].

5.2 Experimentos

Esta seção tem o objetivo de investigar qual combinação entre método de escolha de líder e técnica de controle da área de dominância apresenta o melhor desempenho quando aplicados ao MOPSO, além de analisar se a utilização de diferentes técnicas de controle da área de dominância influencia no comportamento dos métodos de seleção de líder. Somente o problema DLTZ2 foi usado nestes experimentos por ser um problema pertinente para avaliar a escalabilidade dos algoritmos, e devido ao grande número de execuções efetuadas não houve tempo hábil para a execução usando outros problemas, ficando esta execução como sugestão para trabalhos futuros.

Os resultados dos experimentos realizados nesta seção são apresentados através de gráficos nas Figuras 5.5, 5.6 e 5.7, nos quais cada curva representa uma instância composta por um método de escolha de líder e uma técnica de controle da área de dominância. Cada ponto representa a média dos valores obtidos com o indicador de qualidade em cada uma

das trinta execuções efetuadas para cada número de objetivos. O número de iterações utilizado durante cada execução foi fixado em 100, tanto a população quanto o tamanho do repositório foram mantidos em 250. O valor da variável S requerida pelo método CDAS foi mantido sempre conforme os melhores valores obtidos em [12], sendo 0.45 para 2 e 3 objetivos, 0.35 para 5 e 0.30 para 10, 15, 20 e 30 objetivos.

Observe que nestes experimentos os valores de GD e IGD chegam à ordem de 10^{-5} o que pode ser considerado desprezível mesmo para o pior algoritmo. Para decidir se existem ou não diferenças estatisticamente significantes entre os algoritmos os testes estatísticos foram aplicados e seus resultados encontram-se nas Tabelas 5.1, 5.2 e 5.3.

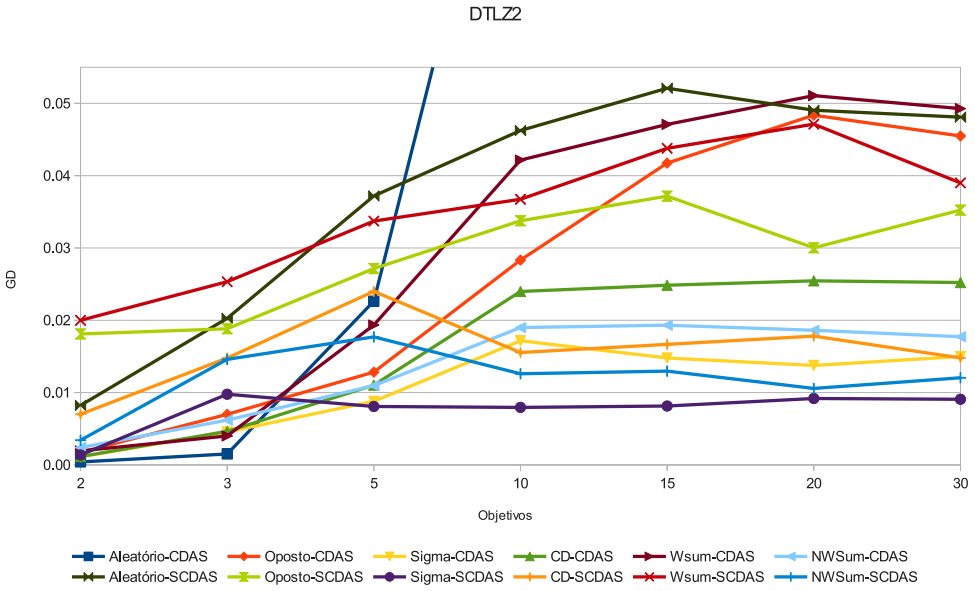


Figura 5.5: Valores de GD ao otimizar o problema DTLZ2

Tabela 5.1: Métodos de escolha de líder e controle da área de dominância com o melhor GD de acordo com o teste de Friedman.

Obj	**GD**
2	Aleatório-CDAS, Sigma-CDAS e CD-CDAS
3	Aleatório-CDAS e WSum-CDAS
5	Sigma-CDAS, CD-CDAS, NWSum-CDAS e Sigma-SCDAS
10	Sigma-CDAS, Sigma-SCDAS, CD-SCDAS e NWSum-SCDAS
15	Sigma-CDAS, Sigma-SCDAS, CD-SCDAS e NWSum-SCDAS
20	Sigma-CDAS, Sigma-SCDAS, CD-SCDAS e NWSum-SCDAS
30	Sigma-CDAS, Sigma-SCDAS, CD-SCDAS e NWSum-SCDAS

Primeiramente é analisada a convergência das instâncias através da métrica GD. O gráfico com os valores de GD (Figura 5.5), mostra que para dois e três objetivos o método

de escolha de líder aleatório com controle da área de dominância CDAS apresentou o melhor desempenho, enquanto os piores resultados foram obtidos com o S-CDAS usando vários métodos de seleção de líder. O teste estatístico do GD na Tabela 5.1 confirma os maus resultados do S-CDAS para dois e três objetivos, não apresentando nenhuma instância com os melhores resultados.

Entre cinco e trinta objetivos, o gráfico do GD mostra os melhores desempenhos com os método de escolha de líder Sigma e NWSum usando a técnica de controle da área de dominância S-CDAS. O teste estatístico da Tabela 5.1 confirma o bom comportamento do Sigma, apresentando o melhor desempenho tanto com CDAS quanto com o S-CDAS. O S-CDAS também apresentou bom comportamento, mostrando melhor desempenho com os métodos de seleção de líder Sigma, CD e NWSum.

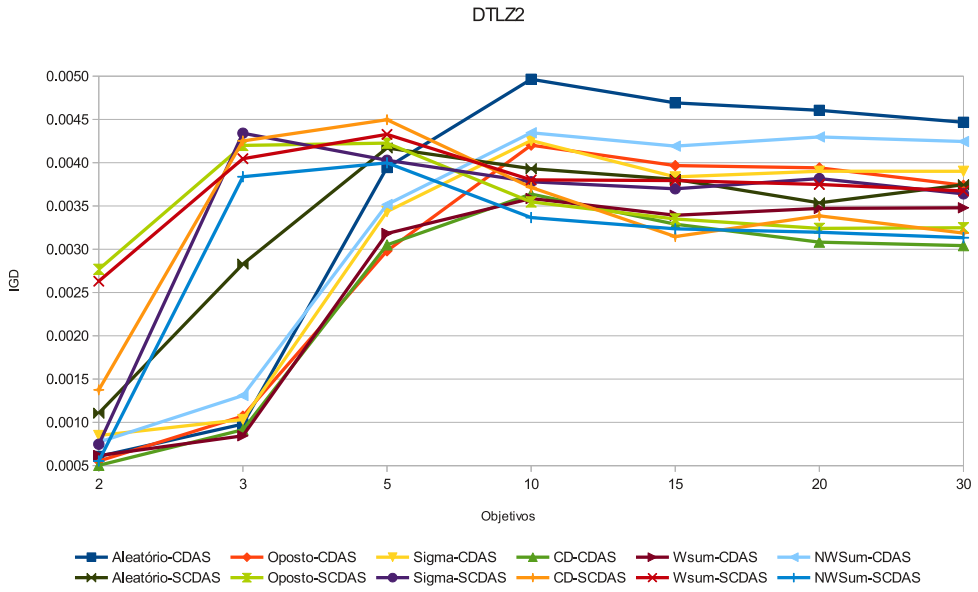


Figura 5.6: Valores de IGD ao otimizar o problema DTLZ2

Tabela 5.2: Métodos de escolha de líder e controle da área de dominância com o melhor IGD de acordo com o teste de Friedman.

Obj	IGD
2	Aleatório-CDAS, Oposto-CDAS, CD-CDAS, WSum-CDAS, Aleatório-SCDAS, Sigma-SCDAS, CD-SCDAS e NWSum-SCDAS
3	Aleatório-CDAS, Oposto-CDAS, Sigma-CDAS, CD-CDAS e WSum-CDAS
5	Oposto-CDAS, Sigma-CDAS, CD-CDAS, WSum-CDAS e NWSum-CDAS
10	CD-CDAS, WSum-CDAS, Aleatório-SCDAS, Oposto-SCDAS, Sigma-SCDAS, CD-SCDAS, WSum-SCDAS e NWSum-SCDAS
15	CD-CDAS, WSum-CDAS, Aleatório-SCDAS, Oposto-SCDAS, Sigma-SCDAS, CD-SCDAS e NWSum-SCDAS
20	CD-CDAS, WSum-CDAS, Aleatório-SCDAS, Oposto-SCDAS, CD-SCDAS e NWSum-SCDAS
30	CD-CDAS, WSum-CDAS, Oposto-SCDAS, CD-SCDAS e NWSum-SCDAS

A diversidade é analisada aqui a partir das métricas IGD e Spacing. A partir do gráfico com os valores de IGD na Figura 5.6, para dois objetivos o método de escolha de líder CD

com controle da área de dominância CDAS apresentou o melhor desempenho, entretanto, no teste estatístico da Tabela 5.2, todas as instâncias aparecem com os melhores valores, exceto o Oposto-SCDAS e o Sigma-CDAS. Para três e cinco objetivos, o CDAS apresentou desempenho superior ao S-CDAS em todas as instâncias, e os métodos de escolha de líderes não apresentaram influência significativa, apesar de o NWSum e o Oposto não estarem entre os melhores para três e cinco objetivos respectivamente.

Entre dez e trinta objetivos, o método de seleção de líder CD apresentou bom desempenho com ambas as técnicas de controle da área de dominância, enquanto o método WSum apresentou melhores desempenhos com a técnica CDAS mas não tão bons com a S-CDAS principalmente por sua tendência de fazer pressão contra os eixos, o que falta no CDAS. A técnica de controle da área de dominância S-CDAS aparece com os melhores resultados usando diversos métodos de seleção de líder, o que indica que esta técnica depende menos do método de seleção de líderes para manter uma boa diversidade.

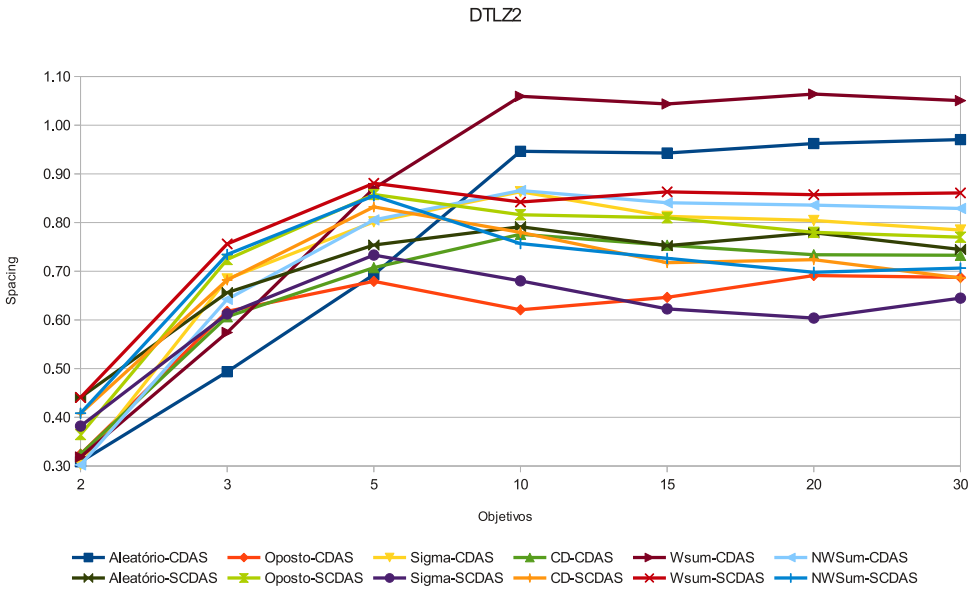


Figura 5.7: Valores de Spacing ao otimizar o problema DTLZ2

Tabela 5.3: Métodos de escolha de líder e controle da área de dominância com o melhor Spacing de acordo com o teste de Friedman.

Obj	Spacing
2	Aleatório-CDAS, Oposto-CDAS, Sigma-CDAS, CD-CDAS, WSum-CDAS e NWSum-CDAS
3	Aleatório-CDAS e WSum-CDAS
5	Aleatório-CDAS, Oposto-CDAS, CD-CDAS, Aleatório-SCDAS e Sigma-SCDAS
10	Oposto-CDAS, Sigma-SCDAS e NWSum-SCDAS
15	Oposto-CDAS, CD-CDAS, Aleatório-SCDAS, Sigma-SCDAS, CD-SCDAS e NWSum-SCDAS
20	Oposto-CDAS, CD-CDAS, Aleatório-SCDAS, Oposto-SCDAS, Sigma-SCDAS, CD-SCDAS e NWSum-SCDAS
30	Oposto-CDAS, Sigma-CDAS, CD-CDAS, Aleatório-SCDAS, Oposto-SCDAS, Sigma-SCDAS, CD-SCDAS e NWSum-SCDAS

A métrica Spacing, representada pelo gráfico da Figura 5.7 e pela Tabela 5.3 mostra, para dois e três objetivos a técnica CDAS com melhor desempenho. O método de seleção de líder não apresenta diferença para dois objetivos, entretanto, para três, os métodos Aleatório e WSum apresentam melhor comportamento. De cinco a trinta objetivos, a técnica S-CDAS apresenta desempenho levemente superior ao apresentar mais instâncias com melhores resultados no teste estatístico, e os métodos de seleção de líder que se destacam são o Oposto com CDAS e o Sigma com S-CDAS.

5.3 Análise dos resultados

Analisando os dados dos experimentos é possível concluir que os métodos de escolha de líder Sigma, NWSum e distância de agrupamento se destacam.

O Sigma, especialmente por sua boa convergência com ambos os métodos de controle da área de dominância, uma vez que este método tende a selecionar líderes próximos à posição atual das partículas, permitindo à partícula continuar explorando a região em que esta já obtém bons resultados, melhorando a convergência.

A distância de agrupamento se destaca especialmente por sua diversidade com ambas as técnicas de controle da área de dominância. Este método foi desenvolvido para introduzir diversidade na busca, e atinge seu objetivo ao selecionar líderes em regiões menos povoadas e com isso as partículas acabam sendo atraídas e encontram soluções não dominadas para povoar estas regiões.

O NWSum obteve um bom desempenho geral com o método S-CDAS. Este comportamento pode ser explicado principalmente por sua característica de selecionar líderes próximos das partículas, entretanto selecionando poucas partículas por líder. Com esta característica, o método é capaz de deixar cada partícula explorar uma pequena região por bastante tempo, aumentando a convergência. Ao mesmo tempo é capaz de espalhar as partículas entre os líderes já encontrados, aumentando a diversidade em fronteiras bem distribuídas. Isto explica seu bom desempenho com o método S-CDAS, que produz fronteiras melhor distribuídas.

O método distância de agrupamento apresentou boa convergência com o S-CDAS,

e o WSum apresentou uma diversidade inferior com o S-CDAS do que com o CDAS, isto indica que a técnica de controle da área de dominância apresenta influência sobre os métodos de seleção de líder.

O comportamento geral obtido entre os métodos de controle da área de dominância foi semelhante ao obtido no Capítulo 4, com o S-CDAS obtendo um desempenho geral melhor para dez ou mais objetivos, e o CDAS apresentando um melhor comportamento para dois, três e cinco objetivos.

Por fim é possível concluir que a técnica de controle da área de dominância selecionada apresenta uma influência maior no processo de busca do que o método de escolha de líderes, especialmente para muitos objetivos.

CAPÍTULO 6

SMPSO BASEADO EM PONTOS DE REFERÊNCIA

Este capítulo apresenta um novo MOPSO chamado SMPSO baseado em pontos de referência *(Reference Based SMPSO (RB-SMPSO))* que utiliza pontos de referência criados num hiperplano, e assim aumenta a distribuição das soluções obtidas, e consequentemente a diversidade e a convergência à fronteira real. Esta abordagem é baseada no algoritmo M-NSGA-II [14] proposto recentemente. Seu funcionamento básico permanece similar ao NSGA-II [13], entretanto seu diferencial é um mecanismo de seleção aprimorado, que ao invés de usar o operador de distância de agrupamento, usa um conjunto predefinido de pontos de referência para dar preferência a membros não dominados da população próximos a cada um desses pontos de referência. O funcionamento básico do NSGA-II é rapidamente descrito abaixo para esclarecer as novas ideias apresentadas pelo M-NSGA-II.

A Seção 6.1 detalha o algoritmo NSGA-II original para apresentar modificações introduzidas pelo M-NSGA-II, detalhadas na Seção 6.2, enquanto a Seção 6.3 explica como essas modificações foram introduzidas no MOPSO. A Seção 6.4 traz uma breve explicação de como gerar os pontos de referência usados aqui. A Seção 6.5 traz um estudo conduzido para avaliar o ganho de desempenho proporcionado pela nova técnica ao SMPSO, portanto o novo RB-SMSPO e o SMPSO tradicional usando os métodos de seleção de líderes Sigma e distância de agrupamento são comparados ao otimizar os problemas DTLZ2 e DTLZ4 entre dois e vinte objetivos. Nesta seção foi efetuado um estudo preliminar, portanto mais simplificado que os demais, incluindo a otimização até vinte objetivos e as métricas de desempenho mais significativas (GD e IGD). A Seção 6.6 traz uma análise dos resultados obtidos.

6.1 *Nondominated Sorting Genetic Algorithm II*

O *Nondominated Sorting Genetic Algorithm II (NSGA-II)* foi proposto em [13] como uma versão aprimorada do *Nondominated Sorting Genetic Algorithm (NSGA)* [37]. O NSGA-II é um dos algoritmos evolutivos mais estudados na atualidade para a otimização de problemas multiobjetivo [36, 2, 14], esta popularidade é devida principalmente à sua facilidade de implementação e qualidade das aproximações da fronteira geradas em diversos problemas.

O funcionamento básico do NSGA-II é o seguinte: Primeiramente uma população P_0 é criada aleatoriamente, em seguida esta população é classificada em níveis de não dominância $(F_1, F_2, ...)$, sendo F_1 composto de todas as soluções não dominadas com relação à população total, F_2 contendo todas as soluções não dominadas após a retirada das soluções que compõem F_1, e assim por diante.

Operadores de seleção por torneio binário, recombinação e mutação são usados para criar uma população filha Q_0 de tamanho N. A partir da primeira geração a nova população passa a ser gerada usando uma combinação entre a população pai e a população filha. O procedimento para uma geração $t > 0$ é mostrado no Algoritmo 4.

Algoritmo 4: Pseudocódigo do NSGA-II

início
  $R_t = P_t \cup Q_t$
  F=classifique(R_t)
  enquanto $|P_{t+1}| < N$ **faça**
    cálculo da distância de agrupamento (F_i)
    $P_{t+1} = P_{t+1} \cup F_i$
  fim enqto
  ordene($P_{t+1}, \geq_n$)
  $P_{t+1} = P_{t+1}[0 : N]$)
  Q_{t+1}= construa nova população (P_{t+1})
  $t = t + 1$
fim

No Algoritmo 4 primeiramente, uma população combinada $R_t = P_t \cup Q_t$ é formada e classificada em níveis de não dominância. A nova população pai P_{t+1} é formada adicionando soluções a partir de F_1 até o número de soluções em P_{t+1} exceder N. Depois, as soluções do último nível aceito (F_l) são ordenadas usando o operador de comparação de

agrupamento ($\geq_n$), que compara as soluções de acordo com o nível de não dominância (i_n) e a distância de agrupamento (i_d), neste comparador, dadas duas soluções i e j é definido que i é preferível a j denotado por ($i \geq_n j$) se ($i_n < j_n$) ou (($i_n = j_n$) e ($i_d > j_d$)). Neste caso as primeiras N soluções são selecionadas, assim $|P_{t+1}| = N$. Em seguida esta população é usada para a seleção, *crossover* e mutação para criar a nova população Q_{t+1}, também de tamanho N.

6.2 M-NSGA-II

O M-NSGA-II utiliza os mesmos procedimentos básicos do NSGA-II, entretanto usa como critério adicional de seleção um conjunto de pontos de referência criados num hiperplano, mais detalhes sobre estes pontos serão discutidos na Seção 6.4. A seguir são apresentados os principais aspectos do M-NSGA-II.

O M-NSGA-II segue os procedimentos do NSGA-II, incluindo a seleção e a divisão em níveis de acordo com a dominância. Entretanto, nele todos os membros da população do nível 1 ao $(l-1)$ são incluídos em P_{t+1} e para escolher os demais $k = N - \sum_{i=1}^{l-1} F_i$ membros da ultima fronteira F_l, todos os membros do nível 1 ao nível l são considerados, constituindo o conjunto S_t.

Para determinar um grau de agrupamento das soluções em S_t próximas a cada ponto de referência, todas as soluções são projetadas no hiperplano criado e associadas com seu ponto de referência mais próximo, então para cada ponto de referência haverá um grupo de soluções associadas. Com H pontos de referência, deve haver uma média de $P_{ideal} = \frac{N}{H}$ soluções por ponto de referência. Se qualquer grupo próximo a um ponto de referência tem menos soluções que P_{ideal}, este ponto é dito deficiente, e um valor de deficiência ($P_{ideal} - P_{atual}$) é calculado, no qual P_{atual} é o número atual de soluções associadas a este ponto de referência.

Então o ponto de referência mais deficiente de todos é escolhido e a solução em F_l mais próxima a este ponto é selecionada. O valor de deficiência deste ponto de referência é reduzido em um, o próximo ponto de referência mais deficiente é escolhido, e a solução mais próxima é identificada. Este processo continua até todos os membros de F_l serem

selecionados para completar os N membros de P_{t+1}. No final, o grupo de soluções de cada ponto de referência é checado, e se estiver vazio, este ponto de referência é considerado extinto, e seu P_{ideal} é distribuído entre seus pontos de referência vizinhos. Normalmente isto acontece em fronteiras desconexas, nas regiões em que não é possível encontrar soluções não dominadas.

Depois que P_{t+1} é formado, este é usado para criar uma nova população descendente Q_{t+1} aplicando os operadores de seleção por torneio, recombinação e mutação. O operador de seleção por torneio considera duas soluções de P_{t+1} e seleciona a melhor. Neste caso é usada uma hierarquia de considerações. Primeiramente, se uma solução pertence a um nível de não dominância melhor que a outra, a primeira é escolhida, como no NSGA-II. Segundo, se ambas as soluções pertencerem ao mesmo nível, mas estão em grupos de soluções de diferentes pontos de referência, os valores de deficiência são comparados, e a solução associada ao ponto de referência mais deficiente é escolhida. Terceiro, se ambas as soluções estão no mesmo nível de não dominância, e estão associadas ao mesmo ponto de referência, a mais próxima a este ponto é escolhida.

Uma vez que os pontos de referência estão bem separados uns dos outros, os membros da população P_{t+1} também tendem a estar distantes entre si. Portanto, é usado um operador de recombinação que cria soluções descendentes próximas aos pais. Depois que a população descendente Q_{t+1} é criada, uma população combinada $R_{t+1} = P_{t+1} \cup Q_{t+1}$ é formada e o procedimento descrito é aplicado novamente.

6.3 RB-SMPSO

Como o SMPSO apresenta a característica indesejada de convergir para pequenas regiões do espaço de objetivos em determinadas condições, surgiu a ideia de utilizar uma abordagem baseada no M-NSGA-II aplicada ao MOPSO para aumentar a diversidade da busca através do aumento da área da fronteira coberta pelas soluções encontradas.

Entretanto, dadas as diferenças fundamentais de funcionamento entre as metaheurísticas, diversas modificações foram necessárias para utilizar a abordagem de pontos de referência no MOPSO. No M-NSGA-II, as informações geradas com os pontos de referência são usa-

das em duas ocasiões: Na seleção dos membros finais para constituir a nova população, e na seleção de soluções para a recombinação. Nenhuma destas etapas existem diretamente no MOPSO, portanto outras maneiras de utilizar estes pontos de referência precisam ser estudadas.

Uma maneira direta de usar estes pontos de referência no MOPSO seria durante a seleção dos líderes, fazendo com que os candidatos a líder mais próximos de cada um dos pontos de referência fossem privilegiados. No entanto, na seleção dos líderes, somente o conteúdo do arquivo externo pode ser considerado, portanto, se as soluções presentes no arquivo estiverem agrupadas, não será possível selecionar líderes próximos a cada ponto de referência, tornando a abordagem menos eficiente.

Uma alternativa encontrada nesse estudo foi a associação de soluções aos pontos de referência no procedimento de arquivamento, assim, além do critério de não dominância, as soluções mais próximas aos pontos de referência são priorizadas e assim uma maior parte da fronteira de Pareto é coberta com um número limitado de soluções.

No procedimento de poda padrão do SMPSO, até que o arquivo externo esteja cheio, todas as soluções não dominadas encontradas são incluídas. Caso uma ou mais soluções passem a ser dominadas por uma nova solução, estas são removidas. Quando o número de soluções atinge o limite definido, caso uma nova solução não dominada seja encontrada e ela não domine nenhuma das soluções já pertencentes ao arquivo, ela é incluída temporariamente, em seguida a distância de agrupamento de todas as soluções do arquivo é calculada, e a solução que apresentar menor valor é removida, uma vez que este valor representa o tamanho do cuboide contendo esta solução, portanto quanto menor a distância de agrupamento, maior o agrupamento.

No novo procedimento de poda proposto aqui, até que o arquivo atinja seu limite, todas as soluções não dominadas encontradas são adicionadas normalmente. Uma vez que o limite seja atingido, se uma nova solução não dominada é encontrada e ela não domina nenhuma outra solução do arquivo, as seguintes operações são executadas:

1. Encontrar o ponto de referência $\vec{r}_c$ mais próximo à nova solução $\vec{f}(\vec{x}_{new})$;

2. Encontrar a solução $\vec{f}(\vec{x}_{ar})$ do arquivo mais próxima a $\vec{r}_c$;

3. Verificar qual solução ($\vec{f}(\vec{x}_{new})$) ou ($\vec{f}(\vec{x}_{ar})$) é mais próxima de $\vec{r}_c$;

4. Manter/incluir a solução mais próxima no arquivo e descartar a mais distante.

Nos passos 1 e 2, a distância é calculada da seguinte forma: primeiramente ambas as soluções são transladadas para um hiperplano normalizado ($0 \sim 1$) usando a Equação 6.1.

$$f'_i(\vec{x}) = \frac{f_i(\vec{x})}{\sum_{j=1}^{m} f_j(\vec{x})} \tag{6.1}$$

Na qual $\vec{f'}(\vec{x})$ é um vetor objetivo transladado para o hiperplano normalizado, $f_i(\vec{x})$ é o i-ésimo objetivo do vetor objetivo $\vec{f}(\vec{x})$, $\vec{x}$ representa uma solução e m representa o número de objetivos do problema. Nesta equação, a ideia é traçar um vetor a partir da origem, cruzando a partícula a ser transladada e identificar o ponto em que este vetor intercepta o hiperplano em $\sum_{i=1}^{m} \vec{f_i}(\vec{x}) = 1$.

Uma vez que o vetor objetivo transladado $\vec{f'}(\vec{x})$ de ambas as soluções é definido, a distância Euclidiana entre eles é calculada. Este procedimento é usado para fazer com que a distância reflita melhor a região em que a solução se encontra, evitando que a sua distância à fronteira interfira.

No passo 3, é usada a distância Euclidiana diretamente, uma vez que, neste caso a proximidade da fronteira é um fator importante.

Estas operações são efetuadas para aumentar a diversidade no procedimento de busca. A diversidade é induzida pois quando uma solução é encontrada próxima a um dado ponto de referência, este sempre terá pelo menos uma solução associada a ele, a menos que estas soluções sejam dominadas por soluções mais distantes, o que não é frequente ao otimizar muitos objetivos. Um aumento na convergência também é introduzido devido à preferência de soluções próximas aos pontos de referência, as quais, em problemas de minimização não convexos são soluções mais próximas à origem.

Neste procedimento não foi levado em consideração o número de soluções já associados a $\vec{r}_c$, entretanto esta é uma possível solução para os problemas apresentados por esta abordagem, discutidos na Seção 6.6, e constitui uma importante sugestão para trabalhos futuros.

Para o correto funcionamento desta técnica o número de partículas definido para o enxame deve sempre ser igual ou maior que o número de pontos de referência.

Uma vez que todos os candidatos a líder do arquivo externo tendem a estar próximos da fronteira e bem distribuídos ao longo dela, um método de seleção de líderes que beneficia todos estes candidatos igualmente é proposto aqui. Neste novo método de seleção, ao invés de as partículas selecionarem seus líderes, o oposto acontece. Cada candidato do repositório seleciona as partículas mais próximas de si para liderar, evitando que estas partículas executem movimentos desnecessários.

Nos experimentos foi notado que a geração de movimentação nas partículas através do espaço de objetivos piora a convergência destas em relação à fronteira real, portanto o objetivo aqui é limitar a movimentação das partículas nos outros eixos, para aumentar a pressão de seleção em direção à fronteira de Pareto, entretanto esta convergência normalmente não é rápida o suficiente para estagnar a partícula em regiões sub-ótimas.

No novo método de seleção, primeiramente é calculado o número de partículas que cada líder vai escolher através de $n = \frac{p}{r}$, com n representando o número de partículas por líder, p o tamanho da população, e r o número de candidatos contidos no repositório. Normalmente se utiliza o número de partículas igual ao tamanho do repositório, resultando em $n = 1$ quando o repositório estiver cheio, entretanto o número de partículas e o tamanho do repositório podem ser diferentes, sendo obrigatório para o correto funcionamento desse algoritmo que $p \geq r$.

Em seguida, cada líder calcula a sua distância para todas as partículas, usando a Equação 6.1, e seleciona as n partículas mais próximas. Com este procedimento, se espera que os líderes escolham partículas em suas respectivas regiões e que as partículas cubram toda a fronteira.

6.4 Pontos de referência

O procedimento para a criação dos pontos de referência descrito aqui se baseia no apresentado em [14] e proposto originalmente em [11]. Neste procedimento, os pontos de referência são criados num hiperplano entre 0 e 1. A concentração destes é dada pela

variável p que representa o número de divisões ao longo de cada eixo do hiperplano. O número total de pontos de referência é definido pela Equação 6.2.

$$H = \binom{m+p-1}{p} \tag{6.2}$$

Por exemplo, para um problema com três objetivos ($m = 3$), os pontos de referência são criados num triângulo plano com vértices em ([1,0,0], [0,1,0], [0,0,1]). Se quatro divisões por eixo ($p = 4$) são escolhidas, $H = \binom{6}{4}$ ou 15 pontos de referência serão criados [14].

No M-NSGA-II, para manter o algoritmo eficiente para qualquer escala de objetivos, o hiperplano é normalizado, para isto, primeiramente o membro ideal da população S_t é determinado identificando o menor valor $\overline{f}_i$ de cada função objetivo e construindo o ponto ideal $\overline{z} = (\overline{f}_1, \overline{f}_2, ..., \overline{f}_m)$. Cada valor objetivo de S_t é transladado ($f'_i = f_i - \overline{f}_i$) para que o ponto ideal do S_t transladado se torne um vetor de zeros. Então o ponto extremo em cada objetivo é identificado resultando em m soluções extremas. Estas m soluções extremas são usados para constituir um hiperplano que é, então estendido para alcançar os eixos objetivo transladados. Os pontos de referência são então recalculados neste hiperplano.

No RB-SMPSO são trabalhados problemas normalizados, por isso o hiperplano não é normalizado e os pontos de referência são gerados usando um incremento de tamanho $\frac{1}{p}$ para cada eixo, além disso, uma restrição é usada, na qual, se $\sum_{i=1}^{m} f_i = 1$ então o ponto é adicionado ao hiperplano.

6.5 Experimentos

Esta seção descreve os experimentos feitos neste capítulo para avaliar o desempenho do RB-SMPSO. O novo algoritmo é comparado com SMPSO original usando o método de escolha de líder distância de agrupamento (CD), e outra versão deste usando o método de escolha de líder Sigma. Nestes experimentos não foram usados métodos para controle da área de dominância. Os três algoritmos são comparados para avaliar suas diferenças em termos de convergência à fronteira real e diversidade na cobertura desta à medida que o número de objetivos aumenta para identificar qual deles apresenta o melhor desempenho.

Os problemas de teste usados nestes experimentos foram o DTLZ2 e o DTLZ4. O DLTZ2 foi usado por ser um dos problemas mais simples da família DTLZ, portanto pertinente para avaliar a escalabilidade do algoritmo. O DTLZ4 foi usado porque o algoritmo RB-SMPSO foca seu funcionamento em melhorar a diversidade das fronteiras obtidas, capacidade esta que é desafiada pelo problema DTLZ4.

Os números de iterações do algoritmo foram fixados de acordo com o problema em 200 e 500 respectivamente, o número de objetivos variou entre dois e vinte, além disso, a população e o tamanho do repositório foram sempre mantidos em 300. A população foi aumentada para 300 (100 nos outros experimentos) porque os números de pontos de referência variam de acordo com o número de objetivos e chegam a um máximo de 275, e o número de iterações foi aumentado (250 nos outros experimentos), porque com o RB-SMPSO as diferenças de desempenho são percebidas com a execução mais prolongada. Por este ser um estudo preliminar a métrica *Spacing* não foi utilizada, e o número de objetivos foi limitado a vinte (trinta nos outros experimentos).

Observe que nestes experimentos os valores de GD e IGD chegam à ordem de 10^{-4} o que pode ser considerado desprezível mesmo para o pior algoritmo. Para decidir se existem ou não diferenças estatisticamente significantes entre os algoritmos os testes estatísticos foram aplicados e seus resultados encontram-se nas Tabelas 6.2, 6.3.

Tabela 6.1: Quantidade de pontos de referência de acordo com o número de objetivos.

Obj	**Camada ext.**	**Camada int.**	**Pontos**
2	50	-	51
3	12	-	91
5	6	-	210
10	3	2	275
15	2	1	135
20	2	1	230

A Tabela 6.1 exibe as divisões por eixo (p) e a quantidade de pontos utilizados para cada número de objetivos. Conforme afirmado em [14] se a quantidade de divisões por eixo for menor que o número de objetivos, não serão gerados pontos intermediários, ou seja, somente serão gerados pontos nas bordas do hiperplano, portanto para evitar um número muito grande de pontos de referência, a partir de dez objetivos são usadas duas camadas

de pontos de referência com poucas divisões por eixo. Este procedimento é detalhado em [14].

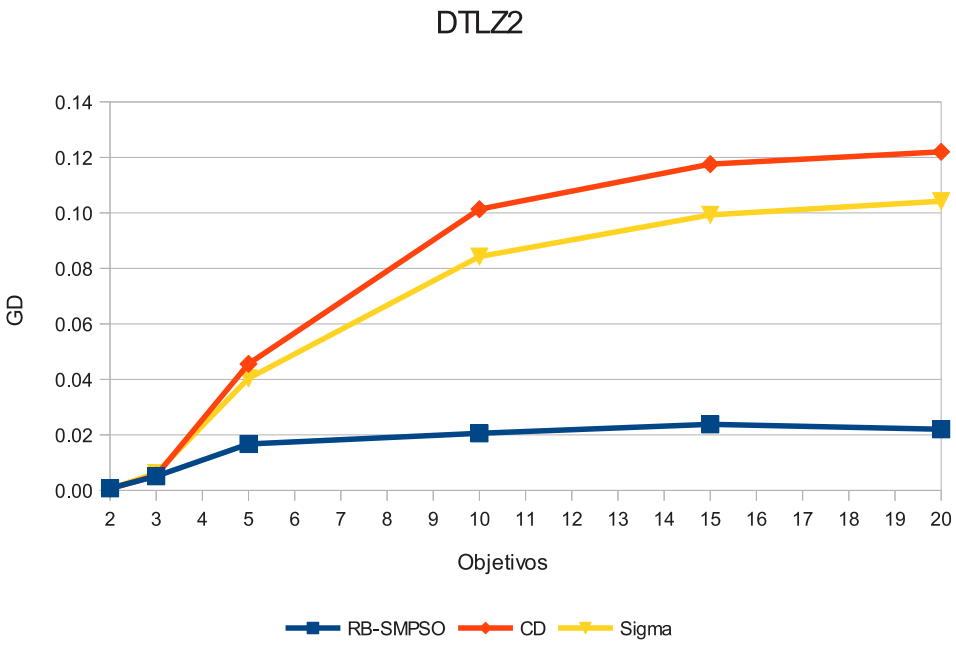


Figura 6.1: Valores de GD ao otimizar o problema DTLZ2.

Na Figura 6.1 é exibido o valor de GD para o problema DTLZ2 representando a convergência à fronteira real. A partir deste gráfico é possível notar que o RB-SMPSO obteve resultados superiores a partir de cinco objetivos, enquanto que para dois e três, todos os algoritmos obtiveram desempenho similar, entretanto, a partir da Tabela 6.2 mostrando os melhores algoritmos cujos resultados não apresentam diferença estatística, é possível verificar que para dois objetivos os melhores desempenhos são obtidos com o SMPSO usando ambos os métodos de escolha de líder, e para três objetivos os três algoritmos apresentam diferenças estatisticamente insignificantes de desempenho. Entre cinco e vinte objetivos o teste estatístico confirma que o RB-SMPSO apresenta os melhores resultados.

A Figura 6.2 mostra o gráfico do IGD para o problema DTLZ2, representando a diversidade obtida com os algoritmos. A partir deste gráfico é possível notar que o RB-SMPSO apresenta melhor desempenho a partir de cinco objetivos, enquanto que para dois e três nota-se pouca diferença entre os resultados. A Tabela 6.2 contendo os algoritmos com

Tabela 6.2: Melhores algoritmos ao otimizar o problema DTLZ2 de acordo com o teste de Friedman.

Obj	Melhores algoritmos	
	GD	**IGD**
2	CD e Sigma	CD
3	RB-SMSPO, CD e Sigma	RB-SMSPO e CD
5	RB-SMSPO	RB-SMSPO
10	RB-SMSPO	RB-SMSPO
15	RB-SMSPO	RB-SMSPO
20	RB-SMSPO	RB-SMSPO

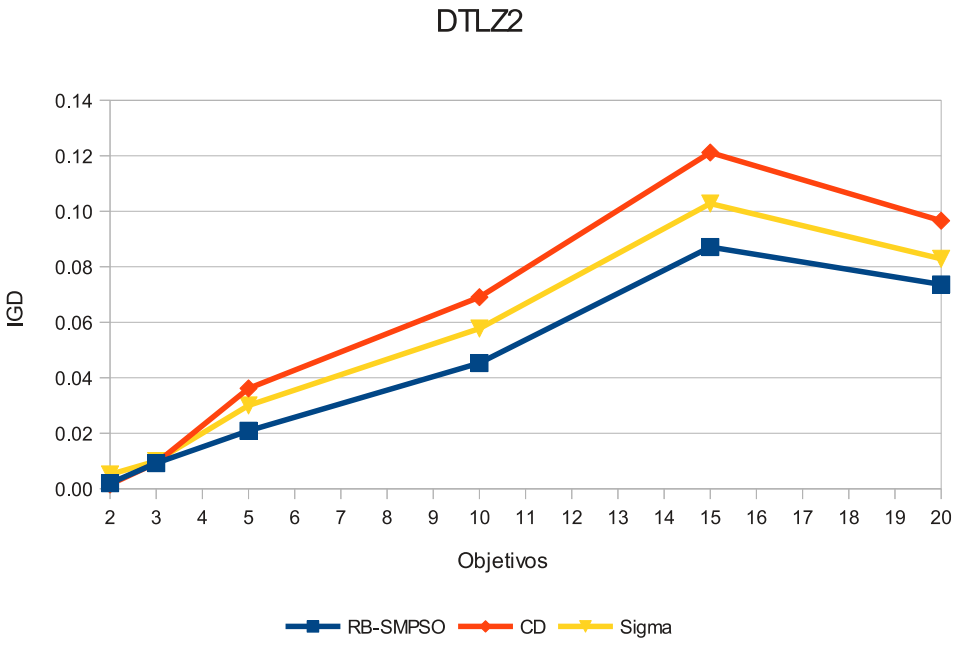


Figura 6.2: Valores de IGD ao otimizar o problema DTLZ2.

melhor desempenho de acordo com o teste de Friedman mostra que para dois objetivos, o SMPSO com método de escolha de líder CD apresenta os melhores resultados, enquanto que para três objetivos o melhor desempenho é obtido com RB-SMPSO e SMPSO com CD. Entre cinco e vinte objetivos o teste de Friedman confirma o bom desempenho do RB-SMPSO.

Os valores de GD para o problema DTLZ4 são mostrados através de um gráfico na Figura 6.3, na qual é possível notar que o algoritmo SMPSO com método de escolha de líder Sigma apresenta os melhores resultados para todos os números de objetivos, exceto para dois, no qual os melhores valores são obtidos com o SMPSO usando o método de escolha de líder CD. A Tabela 6.3 contendo os algoritmos com melhores resultados de

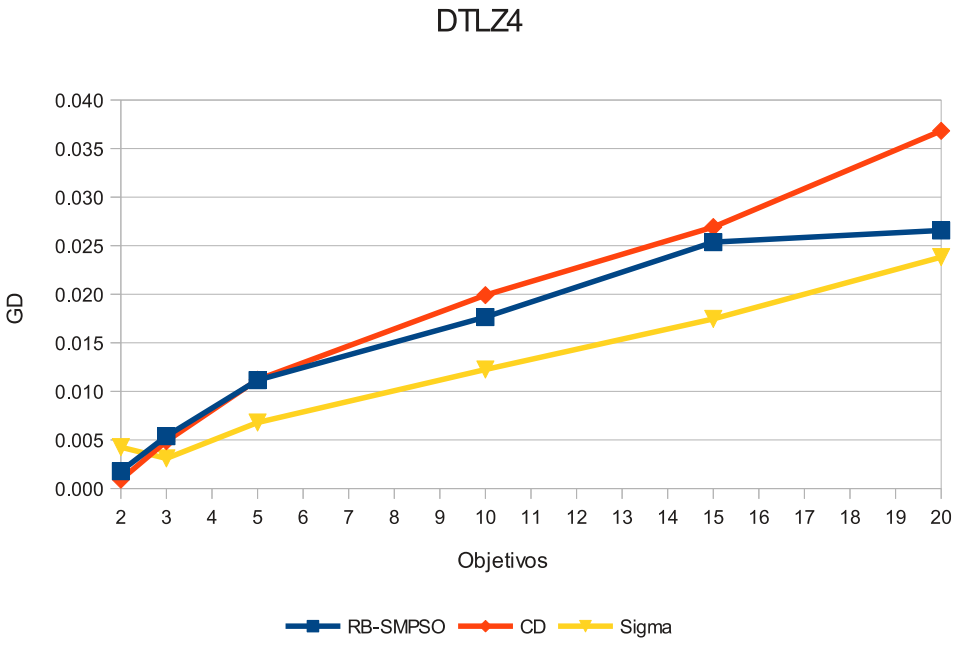


Figura 6.3: Valores de GD ao otimizar o problema DTLZ4.

Tabela 6.3: Melhores algoritmos ao otimizar o problema DTLZ4 de acordo com o teste de Friedman.

Obj	**Melhores algoritmos**	
	GD	**IGD**
2	RB-SMSPO e CD	RB-SMSPO e CD
3	CD e Sigma	Sigma
5	Sigma	Sigma
10	Sigma	Sigma
15	Sigma	Sigma
20	Sigma	Sigma

acordo com o teste de Friedman, mostra que para dois objetivos os melhores desempenhos são obtidos com o RB-SMPSO e com o SMPSO usando o método de escolha de líder original, enquanto que para três objetivos o SMPSO usando ambos os métodos de escolha de líder obtém os melhores resultados. Entre cinco e vinte objetivos o SMPSO com Sigma apresenta os melhores resultados.

A partir dos valores de IGD mostrados através da Figura 6.4 representando a diversidade das soluções obtidas é possível observar que o algoritmo SMPSO com método de escolha de líderes Sigma obtém os melhores resultados para todos os números de objetivos, exceto dois, neste o RB-SMPSO e o SMPSO com CD apresentam resultados parecidos. A partir do teste estatístico mostrado na Tabela 6.3 verifica-se que os algoritmos

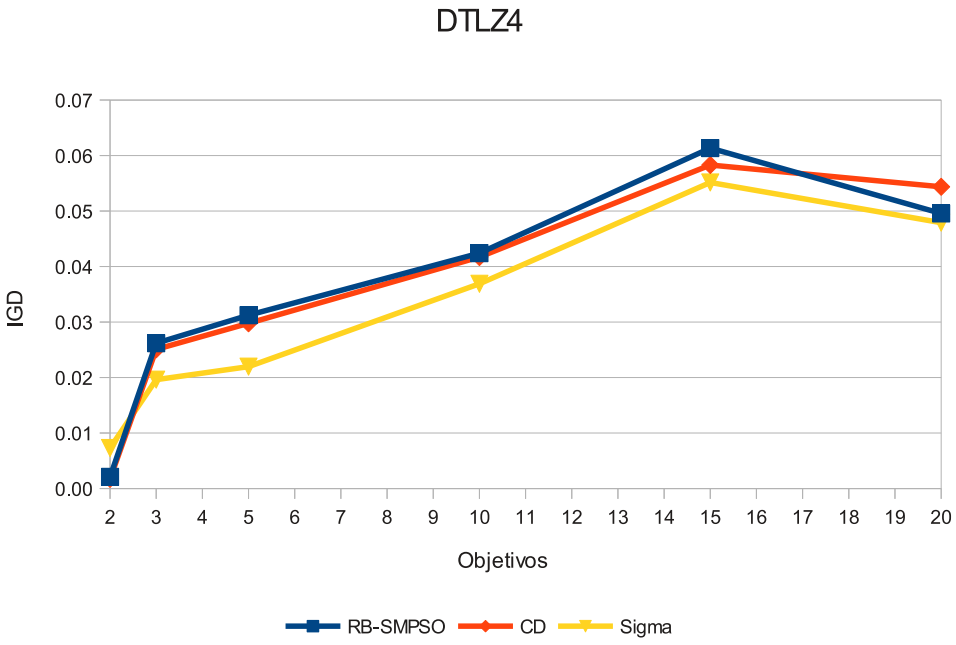


Figura 6.4: Valores de IGD ao otimizar o problema DTLZ4.

RB-SMPSO e SMPSO com CD apresentam os melhores resultados para dois objetivos, enquanto o SMPSO com Sigma obtém os melhores valores para todos os outros números de objetivos.

6.6 Análise dos resultados

Nestes experimentos o número de partículas foi aumentado em relação aos experimentos anteriores para que sempre fosse maior que o número de pontos de referência, que chega a um máximo de 275. Portanto se optou por usar 300 partículas em todos os números de objetivos. A quantidade de iterações também foi aumentada para que fosse possível obter uma maior diferença entre os resultados obtidos, pois durante os estudos preliminares os resultados com 100 iterações não mostraram diferenças expressivas. Consequentemente o custo computacional também foi aumentado, entretanto nos experimentos realizados os três algoritmos utilizaram os mesmos parâmetros, portanto não foi feita uma análise de custos computacionais neste trabalho.

Analisando os dados dos experimentos, é possível concluir que o novo RB-SMPSO apresentou um bom comportamento nos estudos realizados com o problema DTLZ2, ob-

tendo melhores resultados que as outras abordagens tanto em diversidade quanto em convergência para muitos objetivos. Entretanto, nos estudos realizados com o problema DTLZ4 os resultados obtidos com o RB-SMPSO não foram tão bons quanto os obtidos com as outras abordagens.

No problema DTLZ2, que é mais fácil para otimizar, as soluções obtidas foram bem distribuídas, o que ajuda a atingir uma melhor convergência, e aumenta a diferença de desempenho em relação às outras abordagens.

No problema DTLZ4, que é mais difícil, o RB-SMPSO acaba preso em pequenas regiões do espaço de objetivos, pois apesar de ser um algoritmo que estimula a diversidade, ele não traz nenhum mecanismo para garantir isto, e acaba preso nos ótimos locais oferecidos por este problema.

CAPÍTULO 7

CONCLUSÃO

Neste trabalho foram apresentados alguns conceitos gerais sobre os MOPSOs, especialmente os voltados à otimização de muitos objetivos. Dentre as principais dificuldades encontradas estão a deterioração da capacidade de busca causada pelo aumento do número de soluções não dominadas, resultando em dificuldades para convergir à fronteira real e o aumento exponencial no número de soluções requeridas para aproximar toda a fronteira de Pareto, causando problemas em obter diversidade sobre toda a fronteira com um número limitado de soluções.

Para amenizar estes problemas, diferentes abordagens foram exploradas neste trabalho. Primeiramente foi estudada a utilização de duas técnicas de controle da área de dominância das soluções, CDAS e S-CDAS. Ao comparar os resultados dos estudos empíricos, verificou-se que a S-CDAS apresenta um desempenho geral, comparando convergência e diversidade, melhor que a CDAS, além da vantagem de não necessitar de parâmetros adicionais.

A segunda técnica utilizada neste trabalho foi a alteração do método de seleção de líder do MOPSO, para isto, foram estudados seis métodos de seleção de líderes, quatro disponíveis na literatura, um proposto neste trabalho, e o aleatório que é frequentemente usado como parâmetro de comparação. Seu comportamento foi analisado de acordo com duas técnicas de controle da área de dominância, e as melhores combinações foram identificadas através de estudos empíricos. A partir destes estudos foi possível concluir que o método distância de agrupamento apresenta um desempenho melhor com a técnica S-CDAS, enquanto que o WSum se comporta melhor utilizando CDAS, indicando que a técnica de controle da área de dominância tem influência no comportamento de alguns métodos de seleção de líder.

Apesar disso, em geral os métodos apresentaram resultados semelhantes com as técnicas

de controle da área de dominância diferentes, com o método NWSum apresentando os melhores resultados usando S-CDAS, o método Sigma se destacando pela convergência com ambas as técnicas de controle da área de dominância, e o método distância de agrupamento apresentando boa diversidade em geral.

A ultima técnica empregada no MOPSO foi a utilização de um conjunto de pontos de referência bem distribuídos para aumentar a proporção da fronteira real coberta pelas soluções obtidas, com isso aumentando a diversidade e a convergência à fronteira real. Os resultados dos estudos empíricos indicam que o novo RB-SMPSO proposto aqui apresentou um bom desempenho no problema DTLZ2 que é mais simples, principalmente com muitos objetivos. Já no problema DTLZ4, que é mais complexo, o novo algoritmo acaba preso pequenas regiões do espaço de objetivos e apresenta desempenho inferior às outras abordagens.

Em geral, todos os estudos realizados apresentaram melhorias de desempenho em relação ao SMPSO, especialmente no contexto de muitos objetivos. Estes bons resultados incentivam a continuação dos trabalhos desenvolvidos aqui para obter melhorias ainda mais significativas na otimização de MaOPs utilizando a metaheurística MOPSO. A divisão da busca em múltiplos enxames e o refinamento da abordagem usando pontos de referência são os principais trabalhos a serem desenvolvidos futuramente.

Outra sugestão para trabalhos futuros é a utilização do teste estatístico de Friedman para análise geral, agrupando números de objetivos para se determinar os algoritmos que se comportam melhor em intervalos de objetivos.

BIBLIOGRAFIA

[1] Salem Fawaz Adra. *Improving Convergence, Diversity and Pertinency in Multiobjective Optimisation.* Tese de Doutorado, Department of Automatic Control and Systems Engineering, The University of Sheffield, UK, outubro de 2007.

[2] S.F. Adra e P.J. Fleming. Diversity management in evolutionary many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 15(2):183 –195, abril de 2011.

[3] Johannes Bader, Kalyanmoy Deb, e Eckart Zitzler. Faster hypervolume-based search using monte carlo sampling. Matthias Ehrgott, Boris Naujoks, Theodor J. Stewart, e Jyrki Wallenius, editors, *Multiple Criteria Decision Making for Sustainable Energy and Transportation Systems*, volume 634 of *Lecture Notes in Economics and Mathematical Systems*, páginas 313–326. Springer Berlin Heidelberg, 2010.

[4] Jürgen Branke e Sanaz Mostaghim. About selecting the personal best in multi-objective particle swarm optimization. *Parallel Problem Solving from Nature - PPSN IX*, Lecture Notes in Computer Science, páginas 523–532. Springer Berlin / Heidelberg, 2006.

[5] A. Britto e A. Pozo. Using archiving methods to control convergence and diversity for many-objective problems in particle swarm optimization. *IEEE Congress on Evolutionary Computation*, páginas 1 –8, junho de 2012.

[6] M.G. Castillo Tapia e C.A.C. Coello. Applications of multi-objective evolutionary algorithms in economics and finance: A survey. *IEEE Congress on Evolutionary Computation, 2007*, páginas 532 –539, setembro de 2007.

[7] O. R. Castro Jr, A. Britto, e A. Pozo. A comparison of methods for leader selection in many-objective problems. *IEEE Congress on Evolutionary Computation*, páginas 1 –8, junho de 2012.

[8] Olacir Rodrigues Castro Junior, Andre Britto, e Aurora Pozo. Self-controlling dominance particle swarm optimization. *Brazilian Symposium on Neural Networks*, páginas 172 –177, outubro de 2012.

[9] Carlos A. Coello Coello, Gary B. Lamont, e David A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[10] C. A. Coello Coello e M. S. Lechuga. Mopso: a proposal for multiple objective particle swarm optimization. *Proceedings of the Evolutionary Computation on 2002*, volume 2 of *CEC '02*, páginas 1051–1056, Washington, DC, USA, 2002. IEEE Computer Society.

[11] Indraneel Das e J. E. Dennis. Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems. *SIAM Journal on Optimization*, 8(3):631–657, março de 1998.

[12] Andre B. de Carvalho e Aurora Pozo. Measuring the convergence and diversity of CDAS multi-objective particle swarm optimization algorithms: A study of many-objective problems. *Neurocomputing*, 75(1):43 – 51, 2012. Brazilian Symposium on Neural Networks (SBRN 2010) International Conference on Hybrid Artificial Intelligence Systems (HAIS 2010).

[13] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, e T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimisation: Nsga-ii. *Proceedings of the 6th International Conference on Parallel Problem Solving from Nature*, PPSN VI, páginas 849–858, London, UK, 2000. Springer-Verlag.

[14] Kalyanmoy Deb e Himanshu Jain. An Improved NSGA-II Procedure for Many-Objective Optimization Part I: Solving Problems with Box Constraints. Relatório técnico, Indian Institute of Technology, Kanpur, 2012.

[15] Kalyanmoy Deb, Lothar Thiele, Marco Laumanns, e Eckart Zitzler. Scalable test problems for evolutionary multiobjective optimization. Ajith Abraham, Lakhmi Jain,

e Robert Goldberg, editors, *Evolutionary Multiobjective Optimization*, Advanced Information and Knowledge Processing, páginas 105–145. Springer London, 2005.

[16] Janez Demsar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

[17] J.S.H. Dominguez e G.T. Pulido. A comparison on the search of particle swarm optimization and differential evolution on multi-objective optimization. *IEEE Congress on Evolutionary Computation*, páginas 1978 –1985, junho de 2011.

[18] J. J. Durillo, A. J. Nebro, F. Luna, C. A. Coello Coello, e E. Alba. Convergence speed in multi-objective metaheuristics: Efficiency criteria and empirical study. *International Journal for Numerical Methods in Engineering*, 84(11):1344–1375, 2010.

[19] Juan J. Durillo, José García-Nieto, Antonio J. Nebro, Carlos A. Coello, Francisco Luna, e Enrique Alba. Multi-objective particle swarm optimizers: An experimental comparison. *Proceedings of the 5th International Conference on Evolutionary Multi-Criterion Optimization*, EMO '09, páginas 495–509, Berlin, Heidelberg, 2009. Springer-Verlag.

[20] Sean Ekins, J Dana Honeycutt, e James T Metz. Evolving molecules using multi-objective optimization: applying to adme/tox. *Drug Discovery Today*, 15:451–460, 2010.

[21] E.J. Hughes. Multiple single objective pareto sampling. *The 2003 Congress on Evolutionary Computation, 2003.*, volume 4, páginas 2678 – 2684 Vol.4, dezembro de 2003.

[22] Evan Hughes. Radar waveform optimisation as a many-objective application benchmark. Shigeru Obayashi, Kalyanmoy Deb, Carlo Poloni, Tomoyuki Hiroyasu, e Tadahiko Murata, editors, *Evolutionary Multi-Criterion Optimization*, volume 4403 of *Lecture Notes in Computer Science*, páginas 700–714. Springer Berlin / Heidelberg, 2007.

[23] H. Ishibuchi, N. Tsukamoto, e Y. Nojima. Evolutionary many-objective optimization: A short review. *IEEE Congress on Evolutionary Computation*, páginas 2419 –2426, junho de 2008.

[24] Olacir Castro Júnior, André Britto, e Aurora Pozo. A study on the influence of domination control techniques and leader selection methods in many-objective problems. *ENIA 2012*, outubro de 2012.

[25] J. Kennedy e R. Eberhart. Particle swarm optimization. *Proceedings of IEEE International Conference on Neural Networks*, páginas 1942 –1948 vol.4, nov/dec de 1995.

[26] Peter Lindroth, Michael Patriksson, e Ann-Brith Strömberg. Approximating the pareto optimal set using a reduced set of objective functions. *European Journal of Operational Research*, 207(3):1519 – 1534, 2010.

[27] Antonio López Jaimes e Carlos A. Coello Coello. Some techniques to deal with many-objective problems. *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers*, GECCO '09, páginas 2693–2696, New York, NY, USA, 2009. ACM.

[28] Antonio López Jaimes e Carlos Artemio Coello Coello. Study of preference relations in many-objective optimization. *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, GECCO '09, páginas 611–618, New York, NY, USA, 2009. ACM.

[29] S. Mostaghim e J. Teich. Strategies for finding good local guides in multi-objective particle swarm optimization (mopso). *Proceedings of the 2003 IEEE Swarm Intelligence Symposium, 2003. SIS '03*, páginas 26 – 33, abril de 2003.

[30] A.J. Nebro, J.J. Durillo, J. Garcia-Nieto, C.A. Coello Coello, F. Luna, e E. Alba. Smpso: A new pso-based metaheuristic for multi-objective optimization. *Computational intelligence in multi-criteria decision-making, 2009.*, páginas 66 –73, abril de 2009.

[31] Nikhil Padhye, Juergen Branke, e Sanaz Mostaghim. Empirical comparison of mopso methods: guide selection and diversity preservation. *Proceedings of the Eleventh Congress on Evolutionary Computation*, CEC'09, páginas 2516–2523, Piscataway, NJ, USA, 2009. IEEE Press.

[32] Vilfredo Pareto. *Cours D'Economie Politique, volume I and II.* F. Rouge, Lausanne, 1896.

[33] {R Development Core Team}. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2011.

[34] M. Reyes-Sierra e C.A.C. 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.

[35] Hiroyuki Sato, Hernán E. Aguirre, e Kiyoshi Tanaka. Controlling dominance area of solutions and its impact on the performance of moeas. *Proceedings of the 4th international conference on Evolutionary multi-criterion optimization*, EMO'07, páginas 5–20, Berlin, Heidelberg, 2007. Springer-Verlag.

[36] Hiroyuki Sato, Hernán Aguirre, e Kiyoshi Tanaka. Self-controlling dominance area of solutions in evolutionary many-objective optimization. *Simulated Evolution and Learning*, volume 6457 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2010.

[37] N. Srinivas e Kalyanmoy Deb. Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computing*, 2(3):221–248, setembro de 1994.

[38] Tse Guan Tan e Jason Teo. Evolving opposition-based pareto solutions: Multiobjective optimization using competitive coevolution. HamidR. Tizhoosh e Mario Ventresca, editors, *Oppositional Concepts in Computational Intelligence*, volume 155 of *Studies in Computational Intelligence*, páginas 161–206. Springer Berlin Heidelberg, 2008.

[39] D.A. Van Veldhuizen e G.B. Lamont. On measuring multiobjective evolutionary algorithm performance. *Proceedings of the 2000 Congress on Evolutionary Computation*, volume 1, páginas 204 –211, 2000.

[40] David A. Van Veldhuizen e Gary B. Lamont. Evolutionary computation and convergence to a pareto front. *Stanford University, California*, páginas 221–228. Morgan Kaufmann, 1998.

[41] Bambang Wahono, Harutoshi Ogai, Masatoshi Ogawa, Jin Kusaka, e Yasumasa Suzuki. Diesel engine optimization control methods for reduction of exhaust emission and fuel consumption. *IEEE/SICE International Symposium System Integration (SII)*, páginas 722 –727, dezembro de 2012.

[42] Huang Wei, Feng Li, He Zijun, Cui Junzhao, e Zhang Li. Transmission network planning with n-1 security criterion based on improved multi-objective genetic algorithm. *4th International Conference on Electric Utility Deregulation and Restructuring and Power Technologies (DRPT)*, páginas 1250 –1254, julho de 2011.

[43] L. While, L. Bradstreet, e L. Barone. A fast way of calculating exact hypervolumes. *IEEE Transactions on Evolutionary Computation*, 16(1):86 –95, fevereiro de 2012.

[44] Guangrui Zhang, M. Mahfouf, G. Panoutsos, e Shen Wang. A multi-objective particle swarm optimization algorithm with a dynamic hypercube archive, mutation and population competition. *IEEE Congress on Evolutionary Computation*, páginas 1 –7, junho de 2012.

[45] Qingfu Zhang e Hui Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712 –731, dezembro de 2007.

[46] E. Zitzler e L. Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257 –271, novembro de 1999.

[47] Eckart Zitzler, Kalyanmoy Deb, e Lothar Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8(2):173–195, junho de 2000.