

UNIVERSIDADE FEDERAL DO PARANÁ

ANDREI STRICKLER

ANÁLISE DO IMPACTO DAS ESTRATÉGIAS DE SELEÇÃO DE TRADICIONAIS  
MOEAS EM MOEDAS: CMA-ES E UMDA

CURITIBA PR

2017

ANDREI STRICKLER

ANÁLISE DO IMPACTO DAS ESTRATÉGIAS DE SELEÇÃO DE TRADICIONAIS  
MOEAs EM MOEDAs: CMA-ES E UMDA

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Informática no Programa de Pós-Graduação em Informática, setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Ciência da Computação*.

Orientador: Profa. Dra. Aurora Trinidad Ramirez Pozo.

Co-orientador: Prof. Dr. Roberto Santana Hermida.

CURITIBA PR

2017

---

S917a

Strickler, Andrei

Análise do impacto das estratégias de seleção de tradicionais MOEAs em MOEDAs: CMA-ES e UMDA / Andrei Strickler. – Curitiba, 2017.  
50 f ; il. color : 30 cm.

Dissertação - Universidade Federal do Paraná, Setor de Ciências Exatas, Programa de Pós-Graduação em Informática, 2017.

Orientador: Aurora Trinidad Ramirez Pozo – Co-orientador: Roberto Santana Hermida.

Bibliografia: p. 45-50.

1. Algoritmo. 2. Algoritmos computacionais. 3. Algoritmos genéticos. 4. Probabilidades. I. Universidade Federal do Paraná. II. Pozo, Aurora Trinidad Ramirez. III. Hermida, Roberto Santana . IV. Título.


CDD: 005.1

---

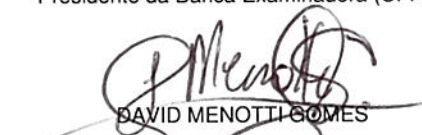
## TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em INFORMÁTICA da Universidade Federal do Paraná foram convocados para realizar a arguição da Dissertação de Mestrado de **ANDREI STRICKLER** intitulada: **Análise do impacto das estratégias de seleção de tradicionais MOEAs em MOEDAs: CMA-ES e UMDA**, após terem inquirido o aluno e realizado a avaliação do trabalho, são de parecer pela sua Aprovação.


Curitiba, 20 de Fevereiro de 2017.



AURORA TRINIDAD RAMIREZ POZO  
Presidente da Banca Examinadora (UFPR)



DAVID MENOTTI GOMES  
Avaliador Interno (UFPR)



JULIO CESAR NIEVOLA  
Avaliador Externo (PUC/PR)



ROBERTO SANTANA HERMIDA  
Co-orientador - Avaliador Externo (UBC)





*à minha família e aos meus amigos*



# Agradecimentos

Primeiramente, agradecer à Deus pelo amparo em todos os momentos de minha caminhada.

Gostaria de agradecer à professora Aurora Pozo. Pela justa e austera orientação recebida ao longo desses dois anos. Agradeço também ao professor Roberto Santana Hermida, por sua importante co-orientação e pelo incentivo no decorrer deste trabalho.

Aos membros da banca por terem gentilmente aceito o convite para a participação da avaliação desta dissertação.

Aos colegas dos grupos de pesquisa da UFPR, em especial ao C-Bio/GrES, sempre aptos em ajudar com sugestões valorosas e em proporcionar momentos de descontração ímpares.

À minha família pelo suporte, força e dedicação ao longo dessa etapa desafiante de minha caminhada. Por fornecerem condições para que eu me dedicasse exclusivamente ao mestrado. Obrigado pela confiança depositada em mim e por sempre estarem ao meu lado.

À minha mãe Mariclei por me presentear com sua imensa dedicação, amor, confiança e apoio. Sem você eu não teria conseguido.

À minha noiva Ana Carolina pelo amor, paciência, carinho, apoio, etc. Obrigado pela compreensão em momentos de sacrifício para que eu pudesse concluir mais esta etapa.

À alguns amigos, em especial, João, Francisco, Gerônimo e José Elias.

Aos meus amigos que cujos os nomes não vou mencionar devido ao tamanho da lista. Obrigado a todos pelos momentos de motivação e descontração.

E por fim, à TODOS aqueles que participaram da realização deste trabalho, direta ou indiretamente.





# Resumo

Pesquisas apontam que, em problemas de otimização mono-objetivo, a capacidade de busca dos algoritmos de estimação de distribuição é fortemente influenciada pelo método de seleção que implementam. O mesmo se observa em problema de otimização multi-objetivo, isto é, os métodos de seleção e as estratégias de substituição desempenham papel importante. No entanto, esta relação entre modelos probabilísticos e os métodos de seleção não tem sido alvo de pesquisas ainda. Neste trabalho, é abordada esta questão avaliando algumas variantes de estratégias de seleção e diferentes modelos probabilísticos. Isto permite detectar possíveis interações entre esses dois componentes dos algoritmos evolutivos multi-objetivo. Especialmente, foram utilizadas as estratégias de seleção utilizadas nos algoritmos NSGA-II, SPEA2 e IBEA, e os modelos probabilísticos implementados como parte do UMDA e CMA-ES, bem como o operador de *crossover* (SBX). Dois conjuntos de problemas de *benchmark* para o contexto multi-objetivo com diferentes características são usados para a análise, são eles: problemas da família DTLZ e da ferramenta COCO recentemente introduzida. Os resultados mostram que utilizar modelos probabilísticos tem uma vantagem sobre o operador genético tradicional, desconsiderando o método de seleção aplicado. Entretanto, os resultados obtidos também mostram que alguns métodos de seleção apresentam um melhor desempenho quando aplicados em conjunto com MOEDAs.

**Palavras-chave:** Algoritmos de Estimação de Distribuição, Problemas de Otimização Multi-Objetivo, Algoritmos Evolutivos Multi-Objetivo, Métodos de Seleção.



# Abstract

Researches point that, in mono-objective optimization problems, the search capability of estimation of distribution algorithms is strongly influenced by the selection method they implement. The same is true in multi-objective optimization problem, that is, the selection methods and replacement strategies play an important role. However, this relationship between probabilistic models and selection methods has not been the subject of research yet. In this work, this question is approached by evaluating some variants of selection strategies and different probabilistic models. This allows to detect possible interactions between these two components of the multi-objective evolutionary algorithms. In particular, we used the selection strategies used in the NSGA-II, SPEA2 and IBEA algorithms, and the probabilistic models implemented as part of the UMDA and CMA-ES, as well as the crossover operator (SBX). Two sets of benchmark problems for the multi-objective context with different characteristics are used for the analysis, they are: problems of the DTLZ family and the recently introduced COCO framework. The results show that using probabilistic models has an advantage over the traditional genetic operator, disregarding the applied selection method. However, the obtained results also show that some selection methods present a better performance when applied in conjunction with MOEDAs.

**Keywords:** Estimation of Distribution Algorithms, Multi-Objective Optimization Problems, Multi-Objective Evolutionary Algorithms, Selection Methods.



# Sumário

<b>1</b>	<b>Introdução</b>	<b>1</b>
1.1	Introdução . . . . .	1
1.2	Motivação e Objetivos . . . . .	3
1.3	Organização do Trabalho . . . . .	3
1.4	Notas sobre terminologia . . . . .	4
<b>2</b>	<b>Fundamentação Teórica</b>	<b>5</b>
2.1	Problemas de Otimização Multi-objetivo . . . . .	5
2.2	Algoritmos Evolutivos Multi-Objetivo . . . . .	6
2.2.1	Nondominated Sorting Genetic Algorithm II (NSGA-II) . . . . .	7
2.2.2	Strength Pareto Evolutionary Algorithm 2 (SPEA2) . . . . .	9
2.2.3	Indicator-Based Evolutionary Algorithm (IBEA) . . . . .	10
2.3	Modelos Probabilísticos . . . . .	11
2.3.1	Algoritmos de Estimção de Distribuição - EDA . . . . .	11
2.3.2	<i>Univariate Marginal Distribution Algorithm (UMDA)</i> . . . . .	13
2.3.3	Estratégia de Evolução com Adaptação da Matriz de Covariância (CMAES) . . . . .	14
2.4	Indicadores de Qualidade . . . . .	15
2.4.1	Hypervolume . . . . .	16
2.4.2	Inverted Generational Distance - IGD . . . . .	16
<b>3</b>	<b>Trabalhos Relacionados</b>	<b>19</b>
<b>4</b>	<b>Análise de métodos de seleção com estratégias de geração de soluções</b>	<b>23</b>
4.1	PISA <i>framework</i> . . . . .	24
<b>5</b>	<b>Experimentos e Resultados</b>	<b>27</b>
5.1	Problemas de Benchmark (Funções de Fitness) . . . . .	27
5.1.1	COCO . . . . .	27
5.1.2	DTLZ . . . . .	29
5.2	Algoritmos . . . . .	30
5.3	Configuração e Avaliação dos Algoritmos . . . . .	30
5.3.1	Comparação dos algoritmos com problemas DTLZ . . . . .	31
5.3.2	Comparação dos algoritmos com problemas da ferramenta COCO . . . . .	34
5.4	Discussão . . . . .	38
<b>6</b>	<b>Conclusão e Trabalhos Futuros</b>	<b>43</b>
	<b>Referências Bibliográficas</b>	<b>45</b>



# Lista de Figuras

1.1	Fluxogramas de GA (esquerda) e EDA (direita) . . . . .	2
2.1	Área de Hypervolume [69, 70] . . . . .	16
4.1	Representação da abordagem proposta . . . . .	24
4.2	Módulos da ferramenta PISA [6]. . . . .	25
4.3	Módulo <i>Variator</i> . . . . .	25
5.1	Diagrama do pós-teste de Nemenyi para Hypervolume considerando 2 e 3 objetivos. . . . .	34
5.2	Gráfico <i>boxplot</i> para métrica Hypervolume com 2 objetivos . . . . .	35
5.3	Gráfico <i>boxplot</i> para métrica $IGD_p$ com 2 objetivos . . . . .	35
5.4	Gráfico <i>boxplot</i> para métrica Hypervolume com 3 objetivos . . . . .	36
5.5	Gráfico <i>boxplot</i> para métrica $IGD_p$ com 3 objetivos . . . . .	36
5.6	Distribuição acumulada empírica (ECD) de valores-alvos alcançados no tempo de execução simulado, medido em número de avaliações de <i>fitness</i> pela dimensão do espaço de decisão (Aval/Dim) para os 58 alvos de todos algoritmos. $\{-10^{-4}, -10^{-4.2}, -10^{-4.4}, -10^{-4.6}, -10^{-4.8}, -10^{-5}, 0, 10^{-5}, 10^{-4.9}, 10^{-4.8}, \dots, 10^{-0.1}, 10^0\}$ . Resultados para todas as 55 funções bi-objetivas e todas as dimensões consideradas. . . . .	37
5.7	ECD para todos os algoritmos, como descrito na Figura 5.6. Resultados para as funções bi-objetivas que pertencem as mesmas classes e $DIM = 2$ . . . . .	39
5.8	ECD para todos os algoritmos, como descrito na Figura 5.6. Resultados para as funções bi-objetivas que pertencem as mesmas classes e $DIM = 3$ . . . . .	40
5.9	ECD para todos os algoritmos, como descrito na Figura 5.6. Resultados para as funções bi-objetivas que pertencem as mesmas classes e $DIM = 5$ . . . . .	41
5.10	ECD para os algoritmos IBEA-CMA-ES, IBEA-UMDA e NSGA-II-SBX, respectivamente, como descrito na Figura 5.6. Resultados para as classes de funções separável - separável e multi-modal - multi-modal com $DIM = 2$ . . . . .	42





# Lista de Tabelas

3.1	Sumário dos artigos relacionados apresentados . . . . .	21
5.1	Classes das funções (problemas) incluídas nas ferramenta COCO . . . . .	29
5.2	Propriedades das classes das funções presentes nos problemas DTLZ . . . . .	29
5.3	Algoritmos produzidos . . . . .	30
5.4	Resultados do teste de Kruskal-Wallis com a média da posição de cada uma das 30 rodadas (e ranking entre os algoritmos) obtidos para as diferentes variantes dos MOEAs com base na métrica $IGD_p$ . . . . .	32
5.5	Resultados do teste de Friedman (e ranking) para as diferentes variantes dos MOEAs entre problemas e número de objetivos com base na métrica $IGD_p$ . . .	33
5.6	Resultados do teste de Kruskal-Wallis com a média da posição de cada uma das 30 rodadas (e ranking entre os algoritmos) obtidos para as diferentes variantes dos MOEAs com base na métrica Hypervolume. . . . .	33
5.7	Resultados do teste de Friedman (e ranking) para as diferentes variantes dos MOEAs entre problemas e número de objetivos com base na métrica Hypervolume. .	34



# Lista de Acrônimos

ART	<i>Average Runtime</i>
CD	<i>Crowding Distance</i>
CMA-ES	<i>Covariance Matrix Adaptive - Evolutionary Strategy</i>
COCO	<i>COmparing Continuous Optimizers</i>
DTLZ	<i>Deb, Thiele, Laumanns e Zitzler</i>
EA	<i>Evolutionary Algorithm</i>
ECD	<i>Empirical Cumulative Distribution Function</i>
EDA	<i>Estimation of Distribution Algorithm</i>
EMNA	<i>Estimation of Multivariate Normal distribution Algorithm</i>
EMO	<i>Evolutionary Multi-Objective Optimization</i>
GA	<i>Genetic Algorithm</i>
HV	<i>Hypervolume</i>
IBEA	<i>Indicator-Based Evolutionary Algorithm</i>
IGD	<i>Inverted Generational Distance</i>
MI	<i>Mutual Information</i>
MND	<i>Multivariate Normal Distribution</i>
MOEA	<i>Multi-Objective Evolutionary Algorithm</i>
MOEA/D	<i>Multi-Objective Evolutionary Algorithm based on Decomposition</i>
MOEDA	<i>Multi-Objective Estimation of Distribution Algorithm</i>
MOP	<i>Multi-objective Optimization Problem</i>
NSGA-II	<i>Nondominated Sorting Genetic Algorithm II</i>
PBIL	<i>Population Based Incremental Learning</i>
PF	<i>Pareto Front</i>
PISA	<i>Platform and Programming Language Independent Interface for Search Algorithms</i>
PM	<i>Probabilistic Model</i>
PS	<i>Pareto Set</i>

SBX	<i>Simulated Binary Crossover</i>
SPEA2	<i>Strength Pareto Evolutionary Algorithm 2</i>
UMDA <sub>c</sub> <sup>G</sup>	<i>Univariate Marginal Distribution Algorithm for Gaussian models</i>

# Capítulo 1

## Introdução

### 1.1 Introdução

Diversos algoritmos têm sido propostos para resolver problemas de otimização multi-objetivo (*Multi-objective Optimization Problems* - MOPs) [12], tais problemas são compostos por dois ou mais objetivos, e um conjunto de restrições com base no problema. Os algoritmos evolutivos multi-objetivo (*Multi-Objective Evolutionary Algorithms* - MOEAs) tem como propósito encontrar um conjunto de indivíduos que satisfaça as restrições do problema, e que também mantenha um conjunto aceitável de soluções não-dominadas com diversidade no espaço de objetivos. O conjunto de soluções candidatas (vetor de variáveis de decisão) normalmente é chamado de população nos MOEAs. O conjunto de soluções ótimas é chamado de conjunto ótimo de Pareto (*Pareto optimal set* - PS), e a imagem deste no espaço de objetivos é chamada de fronteira ótima de Pareto (*Pareto optimal front* - PF).

Na busca por boas soluções, muitos MOEAs usam informações extraídas das soluções encontradas até então para gerar novas soluções. Tais informações podem ser obtidas através de análise da existência de padrões no espaço de objetivos e na correlação entre as variáveis e funções objetivo.

Devida a habilidade de explorar as relações entre as variáveis, os algoritmos de estimação de distribuição (*Estimation of Distribution Algorithms* - EDAs) [31, 39, 40, 43] ganharam atenção da comunidade de computação evolutiva. Esses algoritmos se destacam principalmente pela competência de resolver problemas nos quais algoritmos genéticos (*Genetic Algorithm* - GA) clássicos falham. De acordo com Pelikan et al. [51], esses algoritmos, também chamados de algoritmos genéticos baseado em modelos, consideram a otimização como uma série de atualizações do modelo probabilístico, iniciando com soluções candidatas factíveis geradas a partir de uma distribuição uniforme, e terminando com a amostragem/geração de soluções com base no modelo representando as regiões promissoras do espaço de busca.

Figura 1.1 apresenta as principais diferenças entre GA e EDA, onde são substituídos os métodos de *crossover* e mutação por aprendizagem de modelos e amostragem.

EDAs podem ser divididos de acordo com a complexidade do modelo probabilístico para capturar as interações entre variáveis, e eles são: modelos univariados, bivariados e multivariados [2]. O foco nesta dissertação será em EDAs para representação contínua.

Modelos univariados para o domínio contínuo como o Algoritmo de Distribuição Marginal Univariado (*Univariate Marginal Distribution Algorithm for Gaussian models* - UMDA<sub>G</sub>) [41] e o PBIL (*Population Based Incremental Learning* - PBIL<sub>c</sub>) [56] são baseados em modelos Gaussianos. Esses algoritmos usam métodos eficientes para aprender e amostrar as soluções, entretanto, tais algoritmos não são capazes de encontrar interações entre as variáveis.

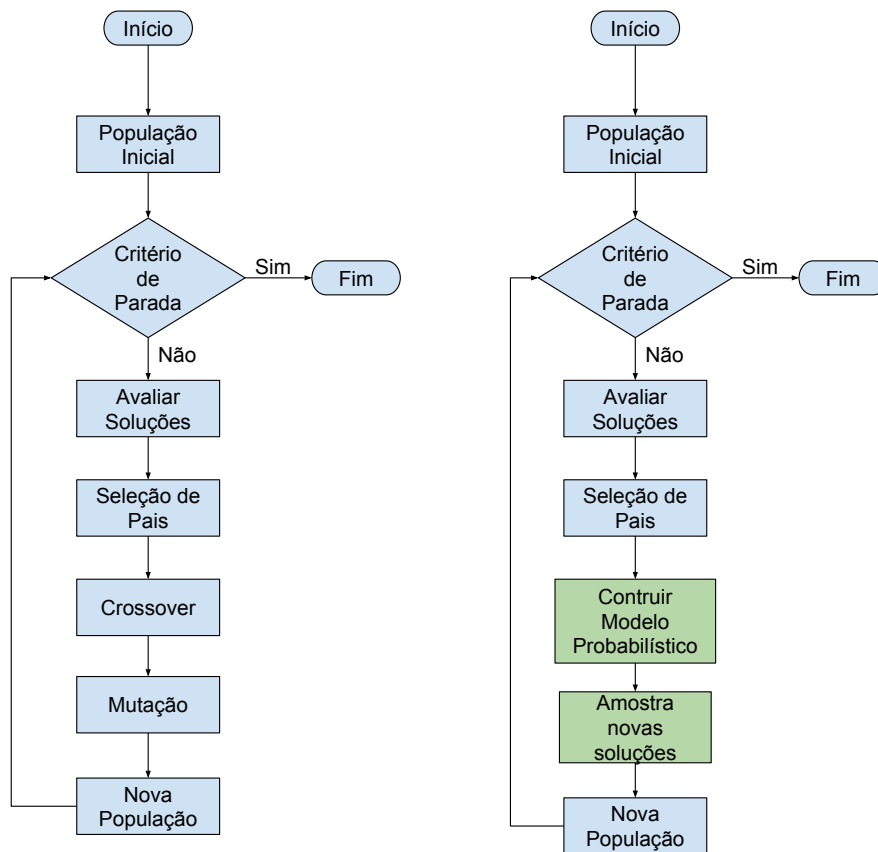


Figura 1.1: Fluxogramas de GA (esquerda) e EDA (direita)

Em contrapartida, os modelos multivariados têm a capacidade de perceber as interações entre variáveis. Nesses modelos pode-se citar: o EMNA (*Estimation of Multivariate Normal distribution Algorithm* [37]) aprende uma distribuição normal multivariada (*multivariate normal distribution* - MND) completa a partir de um conjunto de soluções selecionadas e, a Estratégia Evolutiva com Adaptação da Matriz de Covariância (*Covariance Matrix Adaptive - Evolutionary Strategy* - CMA-ES [29]) incorpora a construção de um modelo dentro de uma estratégia evolutiva para o domínio contínuo. Esse algoritmo aprende uma MND como um modelo probabilístico para gerar novas soluções.

Outro importante passo dentro de um MOEA é a seleção dos indivíduos, normalmente os mais bem adaptados ao meio, que participarão da geração/iteração seguinte. Diversos métodos de seleção foram criados ao longo dos anos. Dentre os diferentes métodos, a seleção baseada no conceito de Dominância de Pareto ganhou notoriedade com os algoritmos NSGA-II (*Nondominated Sorting Genetic Algorithm II*) e SPEA2 (*Strength Pareto Evolutionary Algorithm 2*). O algoritmo IBEA (*Indicator-Based Evolutionary Algorithm*) define medidas de desempenho do conjunto de soluções, chamadas de indicadores de qualidade, e utilizam esses indicadores como objetivo da busca, ao invés do uso da relação dominância de Pareto.

Portanto, o principal propósito desta dissertação é utilizar operadores de recombinação incluindo as abordagens de EDAs para investigar como a junção de modelos probabilísticos e métodos de seleção podem aprimorar os resultados encontrados por MOEAs clássicos em problemas de domínio contínuo.

## 1.2 Motivação e Objetivos

Esse trabalho de mestrado tem como objetivo apresentar uma investigação sobre as capacidades, vantagens e limitações de modelos probabilísticos (EDAs) quando executados com diferentes métodos de seleção de indivíduos em MOEAs. Este estudo é motivado por:

- MOEAs vem sendo reconhecidos como uma boa abordagem para explorar e encontrar uma aproximação de PF para MOPs.
- EDAs ganharam atenção devido ao fato dessas abordagens explorarem as interações entre as variáveis do problema. MO-EDAs são a extensão de EDAs para o domínio de problemas multi-objetivo.
- A existência de diferentes métodos de seleção para MOEAs em conjunto com modelos probabilísticos

Assim, através dessa motivação, são definidos os objetivos deste trabalho:

- Pesquisar e debater sobre trabalhos da área da Otimização Evolucionária Multi-objetivo (*Evolutionary Multi-Objective Optimization* - EMO) especialmente os trabalhos que apresentam algoritmos EDAs e modelos probabilísticos.
- Aplicar abordagens da literatura e explorá-las em problemas multi-objetivos conhecidos como *benchmark*.
- Efetuar análise do desempenho das abordagens enfatizando a convergência e diversidade da busca através de indicadores de qualidade.

Sendo assim, esses estudos tem como objetivo a consolidação do conhecimento adquirido na aplicação de modelos probabilísticos (EDAs) em MOPs.

## 1.3 Organização do Trabalho

Os demais capítulos deste trabalho estão organizados da seguinte maneira:

- **Capítulo 2** introduz os principais conceitos aplicados nesta dissertação, incluindo a descrição de MOPs (Seção 2.1), MOEAs (Seção 2.2), bem como os MOEAs utilizados. A seguir, são apresentados os conceitos de EDAs em conjunto com os modelos utilizados nessa dissertação. Por fim, os indicadores de qualidade são apresentados.
- **Capítulo 3** apresenta os trabalhos relacionados que motivaram o uso dos principais métodos desta dissertação.
- **Capítulo 4** evidencia os fatores fundamentais que motivaram este trabalho com a apresentação da abordagem proposta. Em conjunto, é brevemente apresentado o *framework* PISA, e em específico o módulo em que foram introduzidos os EDAs e os MOPs da ferramenta COCO.
- **Capítulo 5** apresenta os conjuntos de MOPs em que serão testados os algoritmos produzidos. A seguir é definida a configuração dos algoritmos e por fim são comentados os resultados obtidos para cada conjunto de MOPs.
- **Capítulo 6** apresenta as considerações finais deste trabalho e planeja possíveis trabalhos futuros em relação a abordagem proposta.



## 1.4 Notas sobre terminologia

Esta dissertação utiliza diversas palavras e expressões em inglês tais como *fitness*, *benchmark*, entre outros, bem como alguns acrônimos derivados do inglês como EDA, MOP, EMO, MOEA, UMDA, entre outros. Foi definido deixar esses termos em inglês devido ao seu uso comum na literatura especializada para que não haja confusão na interpretação dos termos referenciados.

## Capítulo 2

# Fundamentação Teórica

Neste capítulo serão apresentados os métodos utilizados para fundamentar essa dissertação. Inicialmente são apresentados conceitos de Problemas de Otimização Multi-Objetivo (Multi-Objective Optimization Problems - MOPs), a seguir são introduzidos as características principais de Algoritmos Evolutivos Multi-Objetivo (Multi-Objective Evolutionary Algorithms - MOEAs), bem como os utilizados no presente trabalho. Na sequência são apresentados os Modelos Probabilísticos (Probabilistic Models - PM), em conjunto com os Algoritmos de Estimação de Distribuição (Estimation of Distribution Algorithms - EDAs) que utilizam PM para extrair informações relevantes da população com o intuito de gerar novas soluções baseadas nesse conhecimento adquirido. Por fim são estabelecidos os indicadores de qualidade comumente utilizado pela comunidade de MOEAs, e também aplicados no presente trabalho.

### 2.1 Problemas de Otimização Multi-objetivo

Esta seção inclui conceitos teóricos de problemas de otimização multi-objetivo. As definições formais de MOP, otimalidade de Pareto, dominância de Pareto, conjunto ótimo de Pareto, e fronteiras de Pareto. Neste trabalho o foco é a otimização de problemas multi-objetivo de minimização.

Sendo  $X = (X_1, \dots, X_n)$  um vetor de variáveis aleatórias.  $x = (x_1, \dots, x_n)$  é usado para indicar a atribuição de valores às variáveis, e  $x \in R^n$ . Uma população é representada como um conjunto de vetores  $x^1, \dots, x^N$ , tal que  $N$  é o tamanho da população. De forma similar,  $x_j^i$  representa a atribuição da  $j$ -ésima variável da  $i$ -ésima solução da população. Um MOP pode ser definido a seguir [12]:

$$\begin{aligned} \min (\text{ou max}) \quad & f(x) = (f_1(x), \dots, f_m(x)) \\ \text{sujeito a } & x \in \Omega \end{aligned} \tag{2.1}$$

onde  $x = (x_1, x_2, \dots, x_n)^T$  é o vetor de variáveis de decisão de tamanho  $n$ ,  $\Omega$  é o espaço de decisão,  $F : \Omega \rightarrow R^m$  consiste de  $m$  funções objetivas e  $R^m$  é o espaço objetivo. Otimalidade de Pareto é usado para definir o conjunto ótimo de soluções.

Otimalidade de Pareto:  $x, y \in \Omega$ ,  $x$  domina  $y$  se e somente se  $f_l(x) \leq f_l(y)$  para todo  $l \in 1, \dots, m$  e  $f_l(x) < f_l(y)$  para ao menos um  $l$ . Uma solução  $x^* \in \Omega$  é chamada de ótimo de Pareto se não existir nenhuma outra  $x \in \Omega$  que domine  $x^*$ . O grupo de todas soluções ótimas de Pareto é chamado de conjunto ótimo de Pareto (PS) e as soluções espelhadas no espaço de objetivos é chamado de fronteira de Pareto (PF), i.e.,  $PF = F(x) | x \in PS$ . Em diversas aplicações

reais, a PF é de grande interesse para os tomadores de decisão para a compreensão da natureza dos diferentes objetivos e selecionar a solução final de acordo com as preferências.

O conceito de dominância de Pareto é descrito a seguir: Um vetor  $\vec{u} = (u_1, \dots, u_k)$  domina  $\vec{v} = (v_1, \dots, v_k)$  (denotado por  $\vec{u} \leq \vec{v}$ ), se e somente se,  $\vec{u}$  é parcialmente menor que  $\vec{v}$ , i.e.,  $\forall i \in \{1, \dots, k\}, u_i \leq v_i \wedge \exists i \in \{1, \dots, k\} : u_i < v_i$ .

Uma solução factível é dita não-dominada se não existir outra solução factível melhor que a atual em alguma função objetivo sem ser pior em outra função objetivo.

O conjunto ótimo de Pareto é descrito a seguir: Para um MOP  $\vec{f}(\vec{x})$ , PS é definido como  $\mathcal{P}^* = \{\vec{x} \in \Omega | \neg \exists \vec{x}' \in \Omega, \vec{f}(\vec{x}') \leq \vec{f}(\vec{x})\}$ . E uma fronteira de Pareto é especificada da seguinte forma: Para um MOP  $\vec{f}(\vec{x})$  e um PS  $\mathcal{P}^*$ , a fronteira é definida como  $\mathcal{PF}^* = \{\vec{f}(\vec{x}), \vec{x} \in \mathcal{P}^*\}$ .

Obter a PF de um MOP é um dos maiores alvos da otimização multi-objetivo. Entretanto, uma PF pode conter muitos pontos (soluções). Na prática, uma PF irá conter um número limitado de soluções, assim, surgem dois pontos questionáveis. Primeiro, o processo de otimização deve retornar as soluções mais próximas possíveis da fronteira real, gerando convergência. Segundo, as soluções devem estar uniformemente distribuídas ao longo da fronteira, oferecendo diversidade. Gerenciando esses dois pontos, a fronteira aproximada encontrada assegura que o algoritmo está lidando com soluções (sub-)ótimas, enquanto uma distribuição uniforme das soluções significa que o algoritmo fez uma boa exploração do espaço de busca e nenhuma região é deixada inexplorada [19].

## 2.2 Algoritmos Evolutivos Multi-Objetivo

Um Algoritmo Evolutivo (Evolutionary Algorithm - EA) pode ser caracterizado por três fatores: manter um conjunto de soluções candidatas (população), selecionar a partir da população os melhores indivíduos para a próxima geração, e manipular os indivíduos selecionados com operadores genéticos, normalmente aplicando recombinação e mutação [69].

Analogamente a evolução natural, as soluções candidatas são chamadas de indivíduos e o conjunto de soluções candidatas chamado de população [69]. Cada indivíduo representa uma possível solução, i.e., um vetor de variáveis de decisão ao problema em específico. Neste trabalho a estrutura é assumida como sendo um vetor, por exemplo, um vetor de bits ou um vetor de valores reais, embora possam também ser utilizadas outras estruturas como árvores de programação [35].

Algoritmos Evolutivos Multi-Objetivos (MOEAs) [12] atraíram a atenção de pesquisadores principalmente por conta de que esses algoritmos podem ser aplicados para encontrar diversas soluções próximas às ótimas de Pareto em uma única execução. O objetivo em um MOEA é primeiramente encontrar um conjunto de soluções bem distribuídas próximas a fronteira ótima de Pareto. Ficou em evidência, a partir de diversos estudos, que existem dois objetivos distintos no desenvolvimento de um MOEA: **a)** convergência para a fronteira ótima de Pareto (PF); e **b)** manter um conjunto bem distribuído de soluções não dominadas.

No processo de seleção, o qual pode ser de forma aleatória ou determinística, os indivíduos com pouca qualidade são removidos da população, enquanto as soluções que tem alta qualidade são utilizados para gerar novos indivíduos. O objetivo é focar a busca em áreas específicas do espaço de busca e aumentar a qualidade média da população. A qualidade de um indivíduo em relação à tarefa de otimização é representada por um valor escalar, chamado *fitness* [69]. Vale ressaltar, uma vez que a qualidade de uma solução está relacionada com as funções objetivo e as restrições, um indivíduo deve primeiro ser decodificado antes de seu *fitness* ser calculado.

Ao aplicar os métodos de recombinação e mutação propõe-se que novas soluções serão geradas a partir do espaço de busca pela variação das existentes [45]. O operador de recombinação, chamado de *crossover*, leva em consideração um determinado número de soluções (pais) e gera soluções (filhos), através da recombinação dos pais. Imitando a natureza estocástica da evolução, uma probabilidade de um *crossover* acontecer está associada a este operador. Em contraste, o operador de mutação modifica indivíduos alterando pequenas partes nos vetores (soluções) associados de acordo com uma dada taxa de mutação [12].

O Algoritmo 1 apresenta os principais componentes de um MOEA. O primeiro passo é gerar aleatoriamente uma população inicial com  $N$  indivíduos factíveis. Dentro do ciclo, que é repetido por um número máximo de gerações, os indivíduos que pertencem a população atual são avaliados. Após isso, no processo de seleção uma porcentagem dos melhores indivíduos são mantidos (maioria não dominados) priorizando a convergência da fronteira de Pareto, juntamente com o processo de recombinação. Enquanto, o processo de mutação influencia na diversidade das soluções ao longo da aproximação da fronteira de Pareto.

Os métodos mais conhecidos de seleção são: Ranking por Dominância [21, 22], *Crowding Distance* [14], K-vizinhos mais próximos [57] e baseados em indicadores [71]. Esses métodos são utilizados por algoritmos conhecidos como: NSGA-II, SPEA2, IBEA, dentre outros MOEAs. Os métodos de *crossover* e mutação trabalham com a representação do problema. Para o domínio contínuo, SBX *crossover* [13] e *crossover* uniforme [59] são os mais conhecidos para recombinação, e para o processo de mutação a Mutação Polinomial [16] é amplamente utilizada.

---

**Algoritmo 1:** MOEA básico

---

**Entrada:**  $t \leftarrow 0$ .  
**Saida** População  
 $:$   
1 Gerar  $N$  soluções aleatoriamente.  
2 **Enquanto** Critério de parada não for encontrado **faça**  
3     Avaliar as soluções utilizando as funções objetivo (*fitness*)  
4     Selecione algumas soluções de acordo com o método de seleção  
5     Gerar  $N$  novas soluções com o método de *crossover* de acordo com uma probabilidade  
6     Aplicar o método de mutação de acordo com uma probabilidade  
7      $t \leftarrow t + 1$   
8     Seleciona os melhores indivíduos entre pais e filhos, e preenche a população  
9 **fim-enquanto**

---

A seguir serão descritos os algoritmos detalhando diferentes métodos de seleção utilizados neste trabalho.

### 2.2.1 Nondominated Sorting Genetic Algorithm II (NSGA-II)

O algoritmo NSGA-II utiliza elitismo para manter a diversidade. O mecanismo de elitismo utilizado consiste em combinar os melhores pais com os melhores filhos [14]. O pseudo-código do NSGA-II é apresentado no Algoritmo 2.

Este algoritmo recebe como entrada o tamanho da população  $N$ , número de gerações  $T$ . Começando por criar uma população de tamanho  $N$  chamada  $P_0$ . Então  $P_0$  é classificado de acordo com o mecanismo de ordenação por critério de dominância.  $P_0$  é submetido a um operador de torneio binário para selecionar um conjunto de soluções chamado de *mating pool*,

que será usado como pais para a recombinação. Este *mating pool* pode ser usado tanto pelos operadores de *crossover* e mutação para gerar novas soluções, ou no modelo de aprendizagem de um PM que é usado para gerar novas soluções posteriormente. Ao final deste processo, as novas soluções são incluídas em uma população chamada  $Q_0$ . Após este primeiro passo,  $P_0$  e  $Q_0$  são juntados e chamado como uma população auxiliar  $R$ .

---

**Algoritmo 2:** Pseudo-código do algoritmo NSGA-II

---

**Entrada:**  $N, T$   
**Saída:**  $P$  - Conjunto de soluções não dominadas

```

1 begin
2    $P_0 \leftarrow \text{GerarPopulação}(N)$ ;
3    $\text{FastNonDominated}(P_0)$ ;
4    $\text{MatingPool} \leftarrow \text{TorneioBinário}(P_0)$ ;
5    $Q_0 \leftarrow \emptyset$ ;
6   while  $\text{Tamanho}(Q_0) < N$  do
7      $Q_0 \leftarrow \text{Offspring} \leftarrow \text{GerarNovasSoluções}(\text{MatingPool})$ ;
      // utilizando EDAs ou operadores de crossover
8   end
9    $t \leftarrow 0$ ;
10  while  $t < T$  ou outro critério de parada do
11     $R_t \leftarrow P_t \cup Q_t$ ;
12     $\text{Fronteiras} \leftarrow \text{FastNonDominated}(R_t)$ ;
13     $P_{t+1} \leftarrow \emptyset$ ;
14     $i \leftarrow 0$ ;
15    while  $(\text{Tamanho}(P_{t+1}) + \text{Tamanho}(\text{Fronteiras})) < N$  do
16       $\text{CalcularDistânciaDeMultidão}(\text{Fronteira}_i)$ ;
17       $P_{t+1} \leftarrow P_{t+1} \cup \text{Fronteira}_i$ ;
18       $i \leftarrow i + 1$ ;
19    end
20     $\text{Ordena}(\text{Fronteira}_i, <_n)$ ;
21     $P_{t+1} \leftarrow P_{t+1} \cup \text{Fronteira}_i[1 : (N - P_{t+1})]$ ;
22     $\text{MatingPool} \leftarrow \text{TorneioBinário}(P_{t+1})$ ;
23     $Q_{t+1} \leftarrow \text{GerarNovasSoluções}(\text{MatingPool})$ ; // utilizando EDAs
      ou operadores de crossover
24     $t \leftarrow t + 1$ ;
25  end
26 end

```

---

Através da passo de ordenar pelo critério de dominância, a população auxiliar  $R$  é ordenada criando diversas fronteiras. A primeira fronteira corresponde às soluções não dominadas de toda a população, a segunda é composta por todas as soluções que passam a ser não dominadas, após retiradas as soluções da primeira fronteira, a terceira fronteira é composta por soluções que passam a ser não dominadas após a retirada da primeira e segunda fronteiras, e assim sucessivamente até todas as soluções estarem classificadas em alguma fronteira [14]. Então, as fronteiras são incluídas sequencialmente em  $P_t$ , até que  $P_t$  tenha tamanho  $N$ . Se ao incluir a última fronteira, para preencher  $P_t$ , o seu tamanho ultrapasse  $N$ , então as soluções são ranqueadas de acordo com uma outra medida, chamada distância de multidão (*Crowding*

*Distance* - CD), que tem como objetivo manter a diversidade das soluções. A distância de multidão calcula o quão distante está uma solução de seus vizinhos da mesma fronteira visando a estabelecer uma ordem decrescente que privilegia as soluções mais espalhadas no espaço de busca. Soluções que estão no limite do espaço de busca apresentam só um vizinho, mas são as mais diversificadas da fronteira, elas recebem altos valores para estarem no topo da ordenação [14]. Apenas as melhores soluções desta ultima fronteira são utilizadas para preencher  $P_t$ .

Após criar e preencher  $P_t$  com as melhores soluções, o torneio binário é utilizado para criar a *mating pool*, que é usada para gerar as novas soluções, iniciando um novo ciclo do algoritmo. Por fim, quando um critério de parada é alcançado, o algoritmo retorna  $P$  como a população resultante.

### 2.2.2 Strength Pareto Evolutionary Algorithm 2 (SPEA2)

O algoritmo SPEA2 [66] apresenta como principal diferença em relação ao algoritmo NSGA-II a forma de ranquear e calcular o *fitness* das soluções, e a utilização de elitismo através de um arquivo externo. O pseudo-código do algoritmo SPEA2 é apresentado no Algoritmo 3.

---

#### Algoritmo 3: Pseudo-código do algoritmo SPEA2

---

```

Entrada:  $N, N_a, T$ 
Saida:  $P$ 
:
1 begin
2    $P \leftarrow \text{GerarPopulação}(N)$ ;
3    $P_a \leftarrow \text{GeraArquivoDePopulaçãoVazio}()$ ;
4    $t \leftarrow 0$ ;
5   while  $t < T$  ou outro critério de parada do
6      $Uniao \leftarrow P \cup P_a$ ;
7     for  $solucao_i \in Uniao$  do
8        $s_{i,raw} \leftarrow \text{CalcularRawFitness}(s_i, Uniao)$ ;
9        $s_{i,densidade} \leftarrow \text{CalcularDensidadeDasSolucoes}(s_i, Uniao)$ ;
10       $s_{i,fitness} \leftarrow s_{i,raw} + s_{i,densidade}$ ;
11    end
12     $P_a \leftarrow \text{ObtemSoluçõesNãoDominadas}(Uniao)$ ;
13    if ( $\text{Tamanho}(P_a) < N_a$ ) then
14       $\text{PreencheComAsMelhores}(Uniao, P_a, \text{Tamanho}(P_a))$ ;
15    else if  $\text{Tamanho}(P_a) > N_a$  then
16       $\text{RemoveAsMaisSimilares}(P_a, N_a)$ ;
17     $MatingPool \leftarrow \text{TorneioBinário}(P_a)$ ;
18     $P \leftarrow \text{GerarNovasPopulação}(MatingPool)$ ; // utilizando EDAs
        ou operadores de crossover
19    end
20 end

```

---

Como visto, as entradas do Algoritmo 3 são semelhantes as entradas do NSGA-II, com exceção do parâmetro  $N_a$  que representa o tamanho do arquivo externo. Para cada geração do SPEA2 todas as soluções têm um valor calculado chamado *strength* que é utilizado para definir o *fitness* da solução. O valor de *strength* de uma solução  $s_i$  corresponde ao número de indivíduos,

que pertencem tanto ao arquivo externo como à população e que dominam a solução  $s_i$  [66]. A soma de todos os valores de *strength* de soluções não dominadas pelo indivíduo  $i$  corresponde ao *fitness* de uma solução  $s_i$  (linha 10 do Algoritmo 3). O valor de *fitness* igual a zero (0) indica que um determinado indivíduo não é dominado por nenhuma outra solução. Por outro lado, valores altos de *strength* representam soluções dominadas por vários outros indivíduos [66].

Como o tamanho do arquivo externo é determinado por parâmetro, duas situações podem ocorrer: na primeira delas o número de soluções não dominadas é menor que o tamanho do arquivo, neste caso o arquivo é preenchido com soluções dominadas (linha 13 do Algoritmo 3). No segundo caso, o arquivo pode estar com mais soluções não dominadas que seu limite, então uma operação para eliminar as soluções é efetuada (linha 15 do Algoritmo 3). Somente indivíduos que pertencem ao arquivo externo sobrevivem para uma próxima geração. Como no NSGA-II, uma população *mating pool* é criada, e então soluções da população e do arquivo externo podem ser selecionadas. Essa população *mating pool* pode ser usada tanto pelos operadores de *crossover* e de mutação para gerar novas soluções, quanto no modelo de aprendizagem de um PM que é usado para gerar novas soluções.

### 2.2.3 Indicator-Based Evolutionary Algorithm (IBEA)

No contexto multi-objetivo, indicadores vem sendo utilizados para avaliar a qualidade de uma fronteira de Pareto aproximada. O Hypervolume é um exemplo de indicador usado para a avaliação e comparação de fronteiras de Pareto aproximadas. No algoritmo IBEA, indicadores de qualidade são utilizados para avaliar o conjunto de soluções não dominadas [68]. Para usar o IBEA, é necessário definir qual indicador será usado para associar cada par de soluções a um valor escalar. Um dos indicadores mais utilizados é o Hypervolume devido à sua capacidade de avaliar a convergência e a diversidade do processo de pesquisa ao mesmo tempo [4]. O pseudo-código do IBEA é apresentado no Algoritmo 4.

---

#### Algoritmo 4: Pseudo-código do algoritmo IBEA

---

**Entrada:**  $N, T, k$

**Saida**  $P$

:

1 **begin**

2      $P \leftarrow \text{GerarPopulação}(N);$

3      $m \leftarrow 0;$

4     **while**  $m \leq T$  ou outro critério de parada **do**

5          $\text{MatingPool} \leftarrow \text{TorneioBinário}(P);$

6          $Q \leftarrow \text{GerarNovasPopulação}(\text{MatingPool});$  // utilizando EDAs  
              ou operadores de *crossover*

7          $P \leftarrow P \cup Q;$

8          $m \leftarrow m + 1;$

9         **while**  $\text{Tamanho}(P) > N$  **do**

10              $x^* \leftarrow \text{EncontraOPiorIndividuoPeloFitness}();$

11              $\text{RemoverIndividuoDaPopulação}(x^*, P);$

12         **end**

13     **end**

14 **end**

---

A equação de *fitness* do algoritmo IBEA é dada pela Equação 2.2 e é utilizado para calcular a contribuição de uma dada solução para o valor do indicador de uma população, onde  $k$  é um fator de escala (maior que 0, normalmente 0.05) que depende de  $I_{Hy}$ , que representa o valor do indicador de qualidade para o problema.

$$f(x_i) = \sum_{x_j \in (P - x_i)} -e^{\frac{-I_{Hy}(x_j, x_i)}{k}} \quad (2.2)$$

O valor de  $f(x_i)$  corresponde a qualidade que a aproximação da fronteira de Pareto perde se a solução  $x_i$  for removida da população [72], com base no valor de  $I_{Hy}$ , neste caso, o indicador Hypervolume.

O Algoritmo 4 recebe como parâmetros o tamanho da população  $N$ , número máximo de avaliações  $T$  e fator de escala  $k$ . Começando por criar uma população  $P$  de tamanho  $N$ . Em seguida, repete o seguinte processo até que um critério de parada seja satisfeito: uma população *mating pool* é criada a partir da população  $P$  através do torneio binário, que é usado para gerar novas soluções, através dos operadores de *crossover* e de mutação, ou por aprendizado e amostragem de um modelo probabilístico. Essas novas soluções são adicionadas a uma população auxiliar  $Q$ . Em seguida,  $Q$  é adicionado a  $P$ , e se o tamanho de  $P$  excede o tamanho  $N$ , o pior indivíduo avaliado pelo indicador é removido da população até que  $|P| = N$ . Quando o algoritmo para, é retornado um conjunto de soluções contidas na população [72].

## 2.3 Modelos Probabilísticos

Esta seção introduz importantes conceitos relacionados a modelos probabilísticos utilizados com MOEAs. Após, serão apresentados os modelos probabilísticos utilizados neste trabalho.

De forma simples, um modelo probabilístico é uma ferramenta de análise estatística que calcula com base em dados passados, a probabilidade de um evento acontecer. A modelagem probabilística oferece uma forma sistemática de adquirir esse tipo de regularidades, e portanto pode ajudar a conseguir uma resolução de problemas de forma precisa e confiável [25, 50].

A integração de modelos probabilísticos em EAs, surgiu com um novo paradigma na computação evolutiva, chamado de Algoritmos de Estimação de Distribuição (*Estimation of Distribution Algorithms* - EDA) [39, 40]. Diversos EDAs baseados na modelagem das dependências estatísticas entre as variáveis do problema têm sido propostos como uma forma de descobrir e explorar as interações de variáveis em problemas complexos [39]. A análise dos gráficos resultantes dos modelos de EDAs aprendidos durante a busca pode fornecer informações importantes sobre a estrutura do problema.

Um dos passos fundamentais dos algoritmos que contêm modelos probabilísticos é a aprendizagem do modelo probabilístico para extrair as informações relevantes que os indivíduos selecionados podem conter sobre o problema [20].

### 2.3.1 Algoritmos de Estimação de Distribuição - EDA

Algoritmos de estimação de distribuição usa modelos probabilísticos para substituir os operadores genéticos, a fim de superar algumas limitações dos algoritmos evolutivos tradicionais [34]. Estas limitações são causadas principalmente por EAs que não consideram as dependências entre as variáveis para o problema [39]. Algoritmo 5 apresenta os principais passos de uma MOEDA.



Inicialmente, de forma análoga aos MOEAs, são geradas soluções aleatórias. Ao longo de uma quantidade  $T$  de gerações, as soluções são avaliadas de acordo com as funções de *fitness*, para que um método de seleção escolha as melhores soluções com base nos valores de *fitness*. Então, um modelo probabilístico é construído a partir das informações obtidas, por exemplo média e desvio padrão das variáveis, dos indivíduos selecionados. Por fim, são geradas novas soluções que são amostradas baseadas na distribuição representada pelo modelo probabilístico construído no passo anterior.

---

**Algoritmo 5:** Base de um algoritmo EDA

---

```

1 begin
2    $t \leftarrow 0$ .
3   Gerar  $N$  soluções aleatoriamente.
4   Enquanto  $t \leq T$  ou outro critério de parada seja encontrado faça
5     Avaliar as soluções utilizando as funções de fitness.
6     Selecione  $K$  indivíduos,  $K \leq N$ , com o método de seleção.
7     Construir um modelo probabilístico a partir dos  $K$  indivíduos.
8     Gerar  $N$  novas soluções, amostrando a partir da distribuição representada
        no modelo.
9      $t \leftarrow t + 1$ 
10  fim-enquanto
11 end

```

---

O Algoritmo 5 recebe como entradas os parâmetros,  $K$  que é o número de soluções selecionadas da população,  $N$  é a quantidade de soluções que podem ser armazenadas na população atual e  $T$  é o máximo de gerações.

Com o propósito de aplicar modelos probabilístico em vez operadores genéticos usados em EAs tradicionais, novas soluções candidatas são geradas para o problema em cada iteração usando as duas etapas a seguir:

- Estimar um modelo probabilístico baseado nas estatísticas obtidas a partir do conjunto de soluções candidatas, e
- amostrar do modelo probabilístico aprendido em novas soluções.

O passo importante que distingue os EDAs de muitas outras meta-heurísticas é a construção do modelo que tenta capturar a distribuição de probabilidade das soluções promissoras. Esta não é uma tarefa simples, pois o objetivo não é representar a população de soluções promissoras, mas representar uma distribuição mais geral que capture os padrões característicos das soluções selecionadas que determinam que essas soluções sejam melhores do que outras soluções candidatas.

Os EDAs também podem ser divididos conforme a complexidade dos modelos probabilísticos usados para capturar a interação entre as variáveis do problema: abordagens univariadas, bivariadas ou multivariadas [2]. Algoritmos univariados e contínuos como UMDA<sub>C</sub><sup>G</sup> [41] e PBIL<sub>C</sub> [56] baseiam-se em redes Gaussianas, porém por não considerarem as inter-relações das variáveis do problema, estes apresentam certas limitações para estimar o espaço de busca das novas soluções. Sendo assim algoritmos baseados em relações de duas variáveis foram propostos, como MIMIC<sub>C</sub> [41] e BMDA [49]. Porém como muitos problemas apresentam mais de duas variáveis correlacionadas, algoritmos como ECGA [30], rBOA [1] e CMA-ES [26] foram propostos com finalidade de capturar maiores informações destas relações.

Neste trabalho, a análise da aplicação de MOEDAs (*Multi-Objective* EDAs) em domínio contínuo procura produzir melhores soluções para MOPs do que outros MOEAs baseados em operadores de *crossover* e mutação tradicionais. A avaliação da influência dos diferentes tipos de modelos probabilísticos no comportamento dos MOEDAs também é um fator importante neste trabalho.

A seguir serão introduzidos MOEDAs utilizados no presente trabalho: um univariado e um multivariado,  $UMDA_C^G$  e CMA-ES, respectivamente.

### 2.3.2 *Univariate Marginal Distribution Algorithm (UMDA)*

Uma das abordagens mais simples entre os modelos probabilísticos é assumir que as variáveis do problema são independentes entre elas. Sob este pressuposto, a distribuição de probabilidade de qualquer variável individual não deve depender dos valores de quaisquer outras variáveis [31]. Os EDAs deste tipo são usualmente chamadas EDAs univariados.

O algoritmo  $UMDA_c$  proposto por Larrañaga et al. (2000) [41] é membro do grupo de EDAs univariados. Este algoritmo é uma adaptação de UMDA discreto para o domínio contínuo, tal que um modelo Gaussiano univariado é aprendido para cada variável, e estes modelos são simulados então para gerar soluções novas. Este algoritmo assume que todas as variáveis são independentes para estimar a probabilidade conjunta. Equação 2.3 apresenta o modelo gaussiano univariado aprendido de um conjunto de dados. O conjunto de dados, neste caso, é a população por completa. Um modelo Gaussiano fatoriza um distribuição:

$$f(\vec{x}, \vec{\theta}) = \prod_{i=1}^n f(x_i, \theta_i), \quad \theta_i = \{m_i, std_i\} \quad (2.3)$$

onde  $n$  é o número de variáveis de decisão,  $m_i$  e  $std_i$  são a média e desvio padrão, da  $i$ -ésima variável.

O  $UMDA_c^G$  explora um modelo univariado da população selecionada, onde  $(x_{1,r}, x_{2,r}, \dots, x_{n,r})$  são os valores de cada variável  $i$  de cada solução  $r$  [38].

$$m_i = \bar{x}_i = \frac{1}{\mu} \sum_{r=1}^{\mu} x_{i,r} \quad (2.4)$$

$$std_i = \sqrt{\frac{1}{\mu} \sum_{r=1}^{\mu} (x_{i,r} - \bar{x}_i)^2} \quad (2.5)$$

onde  $\mu$  é o número de soluções selecionadas da população para participarem do aprendizado. Como já mencionado, foram utilizadas todas as soluções da população, portanto  $\mu = N$ .

A população inicial ( $t = 0$ ) é gerada a partir de uma distribuição através do espaço de busca factível. Na primeira geração,  $m_{i,0}$  e  $std_{i,0}$  são estimados com base na população aleatória. No contexto mono-objetivo o  $UMDA_c^G$  usa a seleção por corte, onde os  $\tau$  indivíduos da população com os melhores valores de função objetivo são mantidos para construir e adaptar o modelo de busca. Para a versão multi-objetivo, o método de seleção por corte foi mantido. Portanto, apenas dois parâmetros de algoritmo devem ser especificados para uma implementação do  $UMDA_c^G$ : o tamanho da população  $N$  e o tamanho da seleção  $\mu$  [24].

### 2.3.3 Estratégia de Evolução com Adaptação da Matriz de Covariância (CMAES)

O algoritmo que trabalha com Estratégia de Evolução com Adaptação da Matriz de Covariância (Covariance Matrix Adaptation Evolution Strategy CMA-ES), foi introduzido por [29] e é um otimizador estado da arte para funções mono-objetivas no contexto contínuo [28].

O algoritmo CMA-ES funciona por amostragem de soluções de uma distribuição normal multi-variável baseada em um vetor de médias  $\vec{m}$ , uma matriz de covariância  $\mathbf{C}$  de tamanho  $n \times n$ , e um tamanho de passo  $\sigma$  [5]. A busca persiste adaptando iterativamente esses parâmetros para obter melhores soluções. Para conseguir isso, as soluções geradas são classificadas com base na sua qualidade. Em seguida, essas soluções classificadas são usadas em equações ponderadas para atualizar os parâmetros da distribuição para a próxima geração.

Este algoritmo é considerado um EDA, já que os novos indivíduos são amostrados de acordo com uma distribuição gaussiana multivariada. A matriz de covariância descreve a dependência de cada par de variáveis da distribuição. A adaptação da matriz de covariância é equivalente a aprendizagem de um modelo da função objetivo [33].

O modelo probabilístico estimado em cada geração é uma combinação de informações coletadas ao longo de várias gerações. Em vez de estimar um novo modelo probabilístico em cada geração, o algoritmo adapta o modelo durante a evolução.

O algoritmo CMA-ES inicia com o vetor de médias sendo inicializado com a média do valor normalizado do espaço de busca  $[0, 1]$  portanto  $\vec{m} = (0.5, \dots, 0.5)$ , a matriz de covariância  $\mathbf{C}$  é inicializada como uma matriz identidade, os vetores de trajetória são inicializados  $\vec{p}_\sigma = \vec{p}_c = (0, \dots, 0)$  e o tamanho do passo  $\sigma = 0.3$ .

Então, as seguintes equações são utilizadas para atualizar um modelo CMA-ES:

- Primeiramente, são avaliadas e ranqueadas as soluções da população selecionada de acordo com um critério de qualidade, tal que  $\vec{x}_{i:\mu}$  representa o  $i$ -ésima melhor solução da população  $\mu$ . Utiliza-se a contribuição de Hypervolume como medida de qualidade, tendo apresentado bons resultados em [10].
- Calcula os novos valores do vetor de médias ( $\vec{m}$ ) de acordo com a combinação de pesos das  $\mu$  melhores soluções. Equação 2.6 apresenta esse cálculo:

$$\vec{m} = \sum_{i=1}^{\mu} w_i \vec{x}_{i:\mu}, \sum_{i=1}^{\mu} w_i = 1, w_i > 0 \quad (2.6)$$

- Atualiza o vetor de trajetória  $\vec{p}_\sigma$  através da Equação 2.7:

$$\vec{p}_\sigma = (1 - c_\sigma) \vec{p}_\sigma + \sqrt{c_\sigma(2 - c_\sigma) \mu_{eff}} \mathbf{C}^{(t)-\frac{1}{2}} \frac{\vec{m}^{(t+1)} - \vec{m}^{(t)}}{\sigma^{(t)}} \quad (2.7)$$

onde a variância efetiva acumulada é dada por  $\mu_{eff} = \frac{1}{\sum_{i=1}^{\mu} w_i^2}$ , e  $\frac{1}{c_\sigma}$  identifica uma variável que calcula um passo atrás do vetor de trajetória  $\vec{p}_\sigma$ , definido como  $c_\sigma = \frac{\mu_{eff} + 2}{n + \mu_{eff} + 5}$ .

- Baseado no cálculo do vetor de trajetória, é atualizada a variável de tamanho do passo  $\sigma$  como indicado na Equação 2.8:

$$\sigma = \sigma \exp \left( \frac{c_\sigma}{d_\sigma} \left( \frac{\|\vec{p}_\sigma\|}{E\|\mathcal{N}(0, I)\|} - 1 \right) \right) \quad (2.8)$$

onde o parâmetro de amortecimento é definido como  $d_\sigma = 1 + 2 \max(0, \sqrt{\frac{\mu_{eff}-1}{n+1}} - 1)$ , e a expectativa da normal de um vetor aleatório  $\mathcal{N}(0, I)$  é definido por  $E||\mathcal{N}(0, I)|| \approx \sqrt{n}(1 - \frac{1}{4n} + \frac{1}{21n^2})$ .

- Atualiza o vetor de trajetória  $\vec{p}_c$  através da Equação 2.9:

$$\vec{p}_c = (1 - c_c)\vec{p}_c + h_\sigma \sqrt{c_c(2 - c_c)\mu_{eff}} \frac{\vec{m}^{(t+1)} - \vec{m}^{(t)}}{\sigma^{(t)}} \quad (2.9)$$

onde  $\frac{1}{c_c}$  é a variável (um passo atrás) do vetor de trajetória  $\vec{p}_c$  definida como  $c_c = \frac{4+\mu_{eff}/n}{n+4+2\mu_{eff}/n}$ . A função Heaviside pode bloquear a atualização de  $\vec{p}_c$  se  $||\vec{p}_\sigma||$  for grande, impedindo um aumento rápido do eixo da matriz de covariância  $\mathbf{C}$  em um ambiente linear (quando o tamanho do passo é muito pequeno), é definido como  $h_\sigma = 1$  se  $\frac{||\vec{p}_\sigma||}{\sqrt{1-(1-c_\sigma)^{2(t+1)}}} < (1.5 + \frac{1}{n-0.5}E||\mathcal{N}(0, I)||)$ , ou 0 caso contrário.

- Atualiza a matriz de covariância  $\mathbf{C}$  baseado nos cálculos anteriores através da Equação 2.10:

$$\mathbf{C}^{t+1} = (1 - c_{cov}) \mathbf{C}^{(t)} + \frac{c_{cov}}{\mu_{cov}} \left( p_c p_c^T + \delta(h_\sigma) \mathbf{C}^t \right) + c_{cov} \left( 1 - \frac{1}{\mu_{cov}} \right) \sum_{i=1}^{\mu} w_i OP \left( \frac{\mathbf{x}_{i:\mu}^{(t+1)} - \mathbf{m}^{(t)}}{\sigma^{(t)}} \right) \quad (2.10)$$

onde o peso relativo é  $c_{cov} = \frac{1}{\mu_{cov}} \frac{2}{(n+\sqrt{2})^2} + (1 - \frac{1}{\mu_{cov}}) \min(1, \frac{2\mu_{eff}-1}{(n+2)^2+\mu_{eff}})$ ,  $\delta(h_\sigma^{(t+1)}) = (1 - h_\sigma^{(t+1)})c_c(2 - c_c) \leq 1$  é de pouca relevância e pode ser definido como 0. O produto de um vetor com ele próprio é denotado como  $OP: \mathbb{R}^n \rightarrow \mathbb{R}^{n \times n}, \vec{x} \mapsto \vec{x}\vec{x}^T$ , o qual resulta em uma matriz de grau um com autovetores  $\vec{x}$  e autovalores  $||\vec{x}||^2$ .

Para gerar novas soluções a partir do modelo, o algoritmo CMA-ES utiliza a Equação 2.11, empregando o vetor de médias ( $\vec{m}$ ), tamanho do passo ( $\sigma$ ) e a matriz de covariância ( $\mathbf{C}$ ) atualizados.

$$\vec{x} \sim \mathcal{N}(\vec{m}, (\sigma)^2 \mathbf{C}) \quad (2.11)$$

## 2.4 Indicadores de Qualidade

Esta seção apresenta os indicadores de qualidade utilizados neste trabalho para avaliar os resultados obtidos na avaliação experimental.

Para medir a performance dos MOEAs, métricas chamadas de indicadores de qualidade foram propostas ao longo dos anos. Alguns dos mais utilizados são:

- Distancia Geracional Invertida (Inverted Generational Distance - IGD)
- Hypervolume (HV)

A métrica IGD mede a distância média de todas as soluções no conjunto ótimo de Pareto (PS) em relação as soluções mais próximas na fronteira de Pareto obtida. Esta métrica é o oposto da distância geracional (Generational Distance - GD) [54]). O valor de Hypervolume

de um conjunto de soluções é medido com o tamanho da porção do espaço de objetivo que é dominado por essas soluções coletivamente [64]. O IGD e o Hypervolume podem ser usados para medir tanto a diversidade das soluções quanto a convergência da fronteira de Pareto.

A comparação de desempenho de um ou mais métodos de otimização multi-objetivo é uma tarefa complexa. Os pontos principais na otimização multi-objetivo são: convergência e diversidade de soluções.

### 2.4.1 Hypervolume

Uma métrica amplamente utilizada na avaliação de MOEAs é o indicador de qualidade chamado Hypervolume (HV). Nesta métrica, calcula-se o volume da área coberta entre os pontos das soluções na fronteira de Pareto  $P$  (soluções não dominadas) e um ponto de referência  $R$ . Cada solução  $i \in P$ , constitui um hipercubo,  $v_i$  com referência a um ponto  $R$  [68]. Este ponto de referência pode ser encontrado construindo um vetor com os piores valores da função objetivo. A união de todos os hipercubos encontrados é o resultado desta métrica e, quanto maior o valor de HV melhores são resultados. Valores mais altos de HV indicam que há uma maior dispersão entre as soluções em  $P$  e indicam que há uma melhor convergência para a fronteira de Pareto quando as funções objetivo são de minimização. Se a meta para as funções objetivo é maximização, valores menores são esperados de HV quando um mesmo ponto de referência é utilizado.

O valor de HV corresponde à área formada pela união de todos os retângulos, como indicado na Figura 2.1.

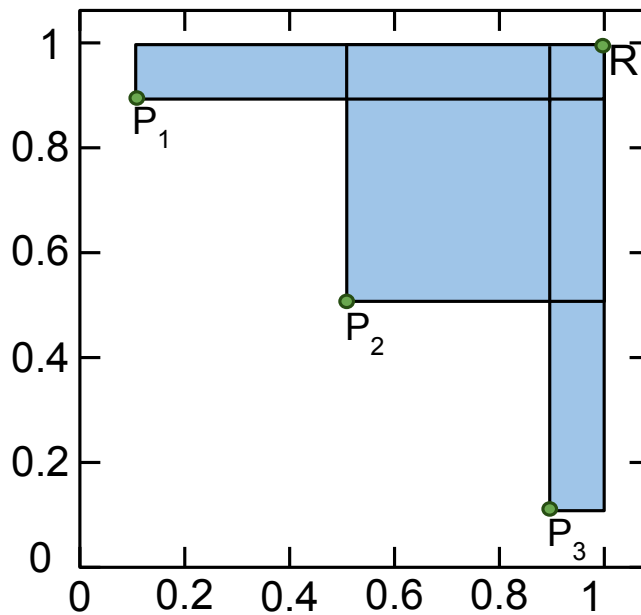


Figura 2.1: Área de Hypervolume [69, 70]

### 2.4.2 Inverted Generational Distance - IGD

Esta métrica mede a distância mínima entre cada ponto do conjunto ótimo de Pareto,  $PF_{true}$ , em relação à fronteira aproximada de Pareto,  $PF_{approx}$ , encontrada em cada rodada

independente. O IGD permite observar se a fronteira de Pareto converge para o conjunto ótimo de Pareto e se existe diversidade de soluções. Equação 2.12 define a métrica (original) IGD [67]:

$$IGD(X,Y) = \frac{\sqrt[p]{\sum_{i=1}^n d_{i,(YX)}^2}}{n} \quad (2.12)$$

onde  $|Y| = n$  é o número de soluções que pertencem à  $PF_{true}$ ,  $p = 2$  e  $d_{i,(YX)}^2$  é a distância Euclideana mínima entre o ponto  $i \in Y$  e o ponto mais próximo em  $X$ . Um valor menor para este indicador significa que  $X$  ou  $PF_{approx}$  é uma melhor aproximação de  $Y$  ou  $PF_{true}$ .

A extensão sugerida por [55], demonstra que para diminuir a influência de soluções muito distantes usa-se um fator  $\Delta_p$ , inserindo na parte inferior da fração (ver Equação 2.12, onde apenas existia  $|Y|$ ) a raiz  $p$ -ésima. Diversos valores de  $p$  foram testados em [55] e o utilizado é  $p = 2$  [11, 62]. Equação 2.13 define a métrica (sugerida)  $IGD_p$  [55]:

$$IGD_p(X,Y) = \sqrt[p]{\frac{\sum_{i=1}^n d_{i,(YX)}^2}{n}} \quad (2.13)$$



## Capítulo 3

### Trabalhos Relacionados

Neste capítulo serão apresentados os trabalhos relacionados aos assuntos abordados no presente trabalho. Esses trabalhos foram selecionados baseado na relação destes com a abordagem proposta neste trabalho. Em uma primeira porção, os trabalhos utilizados como parte do desenvolvimento desse trabalho serão apresentados. Esses trabalhos são focados em modelos probabilísticos,  $UMDA^G$  e CMA-ES, para otimização multi-objetivo. A seguir, serão apresentados os trabalhos que analisam a aplicação de algoritmos de estimação de distribuição (EDAs) com métodos de seleção para o contexto mono-objetivo. Por fim, são apresentados trabalhos que motivaram a utilização de métodos de seleção, arquivamento e substituição de MOEAs conceituados na literatura.

Uma variedade de algoritmos de otimização que utilizam modelos probabilísticos para o domínio contínuo foram propostos ao longo dos anos, e esse fato estimulou o uso de EDAs neste trabalho. Ainda foram apresentados diversos EDAs para o contexto multi-objetivo, chamando esta classe de MOEDAs (Multi-Objetivo EDAs). MOEDAs, bem como os EDAs, são focados em aprender e utilizar as dependências entre as variáveis de decisão do problema. Muitos MOEDAs [52] consistem numa modificação de EDAs existentes cuja forma de manter os indivíduos de uma população para a geração futura é substituída por uma de um MOEA existente.

A utilização do modelo probabilístico  $UMDA^G$  [41] foi tomada por se tratar do modelo mais simples dentre os EDAs para o domínio contínuo. O  $UMDA^G$  calcula valores de média e desvio padrão para cada variável de forma independente. Alguns trabalhos também incluíram a aplicação do  $UMDA^G$  no contexto mono e multi-objetivo para uma variedade diferente de problemas, como em: [24, 58].

No trabalho de Soto et al. [58], o algoritmo UMDA tradicional, e uma variante utilizando distribuição empírica, foram avaliados em conjunto com outros EDAs baseados em cópulas. Foram utilizadas quatro funções de *fitness* que oferecem escalabilidade para  $m$  objetivos (*Sphere*, *Griewank*, *Ackley* e *Summation Cancellation*). UMDA obteve desempenho melhor que os baseados em cópula para as funções que não compreendem interação entre as variáveis. Já para a última função, o algoritmo UMDA não conseguiu resultados satisfatórios devido às interações que existem entre as variáveis do problema.

Em [24], o algoritmo UMDA foi utilizado para realizar a comparação de desempenho com o algoritmo (proposto)  $BayEDA^G$ . O algoritmo proposto também faz uso de modelo gaussiano univariado. Os resultados obtidos indicam um desempenho pouco melhor do algoritmo proposto em relação ao  $UMDA^G$ . Esses resultados melhores podem ser explicados dado que algoritmo proposto amostrar novos indivíduos utilizando média e desvio padrão em conjunto com o auxílio de indivíduos amostrados da inversa de uma distribuição  $\chi^2$  (qui-quadrado).



Para o contexto multi-objetivo, o algoritmo MO-CMA-ES mantém o uso de uma população de indivíduos que adapta a estratégia de busca. O método de seleção é baseado no ranking de não dominados [14] utilizando a distância de multidão ou a contribuição de hypervolume [4] como segundo critério de ranqueamento. Outros MOEDAs que associaram o uso do algoritmo CMA-ES como parte de uma abordagem multi-objetivo foram propostas em [10, 65].

Em [10] foi proposto incorporar novas funções de pesos que auxiliam a inserção de informações do problema na adaptação de um modelo com o algoritmo MO-CMA-ES. Além da função tradicional, a qual utiliza pesos iguais para todas as soluções, foram testadas funções logarítmica, linear e (a proposta) onde os pesos são definidos de acordo com o valor de indicadores de qualidade. Cada uma das funções de peso foi testada com diferentes estratégias de ranqueamento, sendo elas: Distância de multidão (CD), contribuição de Hypervolume e contribuição de  $R_2$ . Este trabalho utilizou os indicadores  $IGD_p$  e Hypervolume aproximado como forma de medir a qualidade das soluções obtidas nos resultados com problemas da família DTLZ e WFG através de muitos objetivos (3,5,8,10,15 e 20). Por fim, comparando os algoritmos em termos de estratégia de ranqueamento, apontaram que utilizar a função logarítmica e a (proposta) com indicador de qualidade, em conjunto com o método de ranqueamento através da contribuição de Hypervolume superaram as outras abordagens (tanto para  $IGD_p$  quanto para Hypervolume), indicando que o uso de uma métrica mais poderosa teve um bom impacto nos resultados.

Com o intuito de investigar o Algoritmo Evolutivo Multi-Objetivo baseado em Decomposição (*Multi-Objective Evolutionary Algorithm based on Decomposition* - MOEA/D), o trabalho de Zapotecas-Martinez et al. [65], analisa o fato de explorar as características do CMA-ES ao invés de aplicar os operadores da Evolução Diferencial (*Differential Evolution* - DE) também usados no MOEA/D. Para o MOEA/D, são criadas subpopulações que lidam com apenas um subproblema cada. Assim, o CMA-ES é aplicado em cada subpopulação, construindo e adaptando um modelo para cada subpopulação. Enfim, é incorporada a estratégia de injetar soluções promissoras da subpopulação em subproblemas vizinhos, melhorando a adaptação da matriz de covariância caso essas soluções consigam fornecer informações relevantes do problema.

Trabalhos incluindo alguns MOEDAs baseados no algoritmo CMA-ES, utilizaram os problemas da ferramenta COCO para avaliar as abordagens propostas [3, 42]. Um fato interessante é que nesses trabalhos que utilizaram a ferramenta COCO para avaliação, optou-se pelo uso do método de seleção do algoritmo NSGA-II. Alguns trabalhos dissertam sobre a diferença de desempenho com diferentes métodos de seleção em EDAs mono-objetivos [8, 9, 53].

Em [8], foi investigado o impacto de métodos de seleção de soluções com alto, baixo, um misto (baixo e alto) valor de *fitness* em construir, ou atualizar, modelos eficientes para o contexto mono-objetivo. A premissa em inserir soluções com baixa qualidade para a próxima geração, anexa diversidade ao processo de busca. Foram testados diversas porções  $p$  da população a ser selecionada para a próxima geração, incluindo o caso onde não houvesse seleção. Problemas como *Onemax* e *MAXSAT* do contexto discreto (binário) foram utilizados como função de *fitness* para diversificar e intensificar a investigação sobre os métodos de seleção. Os resultados mostraram que, de fato, a importância de escolher um método de seleção é fundamental para compor um algoritmo eficiente. Tanto que em determinados casos, o método de seleção que seleciona a mesma quantidade de soluções com valores altos e baixos de *fitness* supera os resultados obtidos pelo método de seleção que mantém apenas as soluções que tem altos valores de *fitness*.

No trabalho de Brownlee et al. [9], a análise é bem similar a realizada no trabalho acima mencionado. Neste trabalho, foi analisado o efeito em solucionar apenas as melhores, ou piores, soluções com o método de truncamento e de torneio (com diferentes quantidades de participantes: 2,3,5 e 10) para aprender, e construir, modelos eficientes. Fator considerável é que foi acrescentada nesta análise a importância em verificar a possibilidade da origem de possíveis erros, como: perda de informação e o falso aprendizado de dependências entre as variáveis, que podem comprometer na construção dos modelos. Também utilizando o domínio discreto, problemas como *Onemax*, *checkerboard* e *trap-5* foram utilizados como função de *fitness* neste trabalho. A seleção de soluções de alto e baixo valores de *fitness* alcançaram resultados similares para todos os problemas testados, reforçando a utilidade em incluir soluções de baixa aptidão na construção do modelo.

O trabalho de Santana et al. [53] apresenta a concepção de seleção customizada, ou personalizada. Este formato de seleção é dada por definir uma probabilidade (distribuição de probabilidade de Boltzmann) para cada indivíduo da população, no qual o modelo é construído usando o cálculo de informação mútua (*Mutual Information* - MI) através da distribuição univariada e bivariada. Para fim de comparação foi utilizado o algoritmo UMDA. As soluções são amostradas novas soluções com base no modelo aprendido e introduzidas soluções elitistas (com alto valor de *fitness*). Como função de *fitness* foi utilizado o problema de predição de estrutura de proteínas.

Diversos trabalhos chamados de *survey* como em [44,46,47,60] apontam a utilização de MOEAs tradicionais como um importante passo para obter soluções (sub-)ótimas. Diferentes MOEAs que implementam métodos de seleção, arquivamento e substituição como NSGA-II, SPEA2 e IBEA estão entre os regularmente utilizados. Ainda que esses algoritmos não são os estado-da-arte para o contexto multi-objetivo tais tem grande representatividade dentro da comunidade de MOEAs [47]. Um fator que estimulou o uso destes foi devido as diferentes características apresentadas em cada uma das técnicas, dado que a partir da utilização dos diferentes operadores de recombinação gera-se algoritmos capazes de explorarem regiões variadas do espaço de busca. A seguir, a Tabela 3.1 apresenta um sumário dos trabalhos relacionados acima comentados com o título, ano da publicação, principal tópico relacionado a este trabalho e a referência.

Tabela 3.1: Sumário dos artigos relacionados apresentados

	Título	Ano	Principal tópico relacionado	Referência
1	<i>Bayesian inference in estimation of distribution algorithms</i>	2007	UMDA	[24]
2	<i>Modeling with Copulas and Vines in Estimation of Distribution Algorithms</i>	2012	UMDA	[58]
3	<i>Transfer weight functions for injecting problem information in the multi-objective CMA-ES</i>	2016	CMA-ES	[10]
4	<i>Injecting CMA-ES into MOEA/D</i>	2015	CMA-ES	[65]
5	<i>Anytime Bi-Objective Optimization with a Hybrid Multi-Objective CMA-ES (HMO-CMA-ES)</i>	2016	CMA-ES e COCO framework	[42]
6	<i>Benchmarking RM-MEDA on the Bi-objective BBOB-2016 Test Suite</i>	2016	COCO framework	[3]
7	<i>Approaches to selection and their effect on fitness modelling in an Estimation of Distribution Algorithm</i>	2008	Seleção em EDAs mono-objetivo	[8]
8	<i>Customized Selection in Estimation of Distribution Algorithms</i>	2014	Seleção em EDAs mono-objetivo	[53]
9	<i>Influence of Selection on Structure Learning in Markov Network EDAs: An Empirical Study</i>	2012	Seleção em EDAs mono-objetivo	[9]
10	<i>Evolutionary Multi-objective Optimisation: A Survey</i>	2015	Survey sobre MOEAs	[47]



## Capítulo 4

# Análise de métodos de seleção com estratégias de geração de soluções

Esse capítulo aborda aspectos gerais da abordagem proposta e detalhes específicos de implementação do presente trabalho. Essa abordagem tem como objetivo investigar a aplicação de diferentes modelos probabilísticos para MOPs de *benchmark* distintos, bem como, analisar a aplicação em conjunto, de diferentes métodos de seleção dos indivíduos com modelos probabilísticos. A exploração de modelos probabilísticos como alternativa aos operadores de recombinação vem sendo tópico para muitas pesquisas, surgindo os EDAs e MOEDAs, porém o que não foi encontrado na literatura é a análise de diferentes métodos de seleção com a aplicação de EDAs.

Figura 4.1 apresenta a representação da abordagem proposta. Os MOEAs são compostos por diferentes métodos de seleção, vários operadores de recombinação e os distintos conjuntos de problemas de *benchmark* no contexto multi-objetivo. Assim, acrescentando uma diversidade de algoritmos que serão produzidos ao combinarmos os vários métodos acima mencionados para resolver problemas com diferentes complexidades para cada conjunto de problemas.

Um dos principais tópicos de pesquisa que este trabalho considera é se a escolha do método de seleção e a estratégia utilizada para gerar as novas soluções podem ser consideradas independentemente, ou até que ponto elas têm um efeito conjunto no comportamento dos MOEAs. Nesta dissertação esta incluída uma análise aprofundada que contém dois modelos probabilísticos (UMDA<sub>c</sub><sup>G</sup> e CMA-ES) para amostrar novas soluções, a avaliação dos algoritmos com o conjunto de problemas de *benchmark* (DTLZ e COCO), e a investigação da qualidade das soluções geradas utilizando diferentes indicadores de qualidade (Hypervolume e IGD<sub>p</sub>).

O tópico de como a relação entre o método de seleção e a estratégia de modelos probabilísticos influencia os resultados dos algoritmos evolutivos não foi investigado em detalhes para o contexto multi-objetivo. Além das dificuldades específicas dos problemas, conforme relatado para problemas mono-objetivo (ver Capítulo 3), quando considerado o contexto multi-objetivo, existem outros fatores que podem influenciar a eficiência da busca realizada pelos algoritmos. Esses fatores incluem métodos de substituição e estratégias de arquivamento. Neste trabalho, a relação entre o modelo de aprendizagem, as estratégias de seleção, substituição e arquivamento, no comportamento dos MOEAs são estudados.

Uma característica distinta da abordagem aqui descrita é considerar, por um lado, somente o papel do operador de recombinação e, por outro lado, o efeito conjunto de todos os outros componentes dos MOEAs (por exemplo, seleção, substituição, arquivamento). Ao fazer essa separação dos componentes dos MOEAs, podemos focar nossa análise no impacto que diferentes estratégias para selecionar, substituir e arquivar as soluções, têm na eficácia da

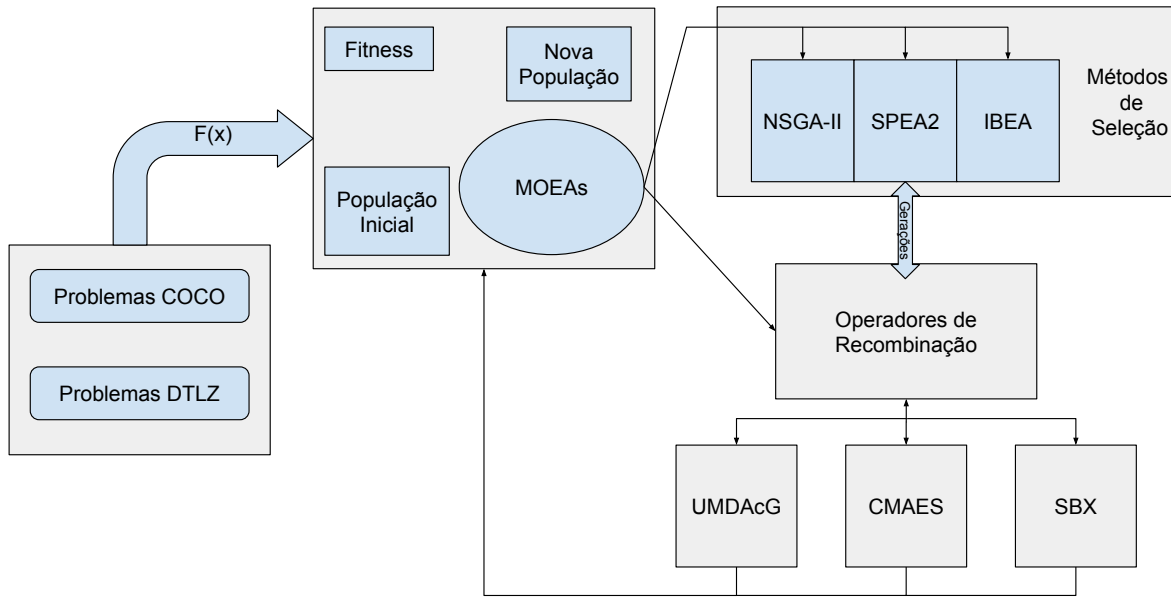


Figura 4.1: Representação da abordagem proposta

modelagem probabilística. Todos os MOEAs previamente comentados serão estudados usando esta abordagem.

Esta abordagem permite investigar duas suposições principais. Uma das expectativas é que alguns métodos para selecionar e arquivar as soluções possam ser mais apropriados para explorar os benefícios da modelagem probabilística. Também considera-se que o sucesso da combinação da aplicação de modelos probabilísticos, e dos outros componentes dos MOEAs, dependerá das características dos problemas (funções de *fitness*) que estão sendo otimizados.

Para apoiar essas premissas, fora separada a fase de experimentação em duas análises diferentes. Foram utilizados dois conjuntos de problemas de *benchmark* distintos, os da ferramenta COCO e os problemas DTLZ, para avaliar o desempenho dos algoritmos produzidos em diversas funções de *fitness* que apresentam complexidades variadas.

## 4.1 PISA *framework*

Aproveitando da modularidade oferecida pelo *framework* PISA [6], uma plataforma para desenvolvimento e execução de algoritmos de busca no contexto multi-objetivo em linguagem C/C++, foram acoplados os algoritmos MOEDAs (ver Subseção 2.3.1). A plataforma PISA trabalha com dois módulos principais, sendo um chamado de **selector** e outro de **variator**.

Basicamente, o módulo **selector** é composto pelos métodos de seleção, arquivamento e substituição dos MOEAs. Já a parte que cabe ao módulo **variator** é formada pelos métodos de recombinação (*crossover*), mutação, e avaliação (funções de *fitness*). A Figura 4.2 apresenta os módulos que envolvem a execução de um algoritmo com a ferramenta PISA.

O módulo **monitor** é considerado como o automatizador de rodadas dos algoritmos. Dessa forma, esse módulo trabalha supervisionando os executáveis dos módulos principais. Quando uma nova rodada independente deve ser executada, o monitor informa aos módulos principais quais são os parâmetros que cada um deve utilizar, a fim de gerar uma **fronteira de Pareto - PF** aproximada obtida. Outro fator implementado para o *framework* PISA é o módulo que realiza a **análise de desempenho** dos algoritmos executados. Neste módulo, alguns

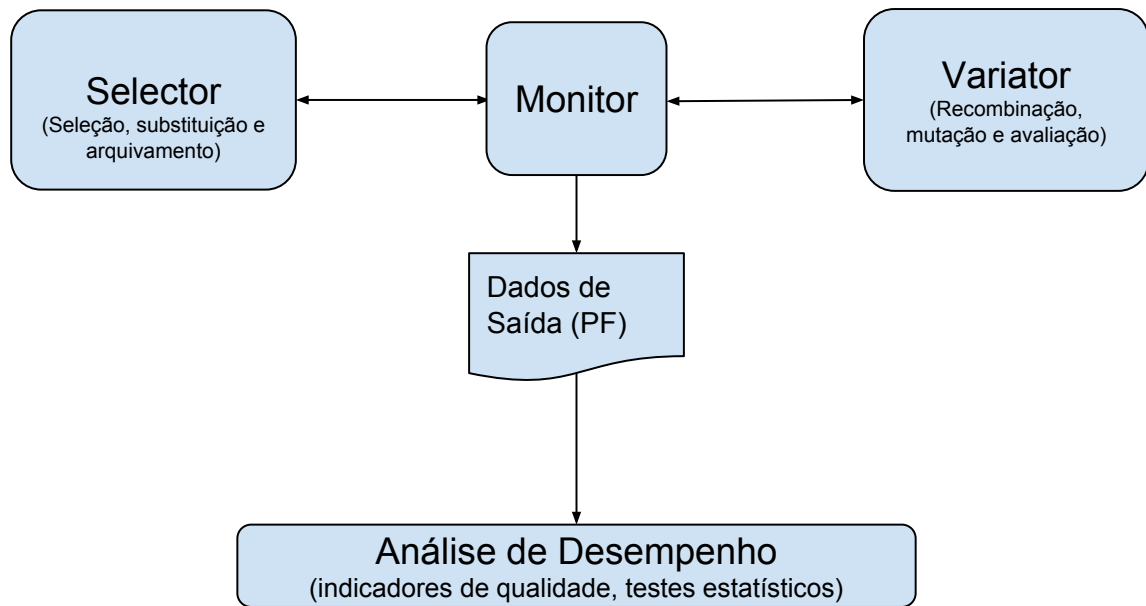


Figura 4.2: Módulos da ferramenta PISA [6].

indicadores de qualidade estão disponíveis (*epsilon*,  $R_2$  e Hypervolume), bem como testes estatísticos (Kruskal-Wallis, Mann-Whitney, entre outros).

Os MOEDAs (ver Subseção 2.3.1), e os problemas da ferramenta COCO (ver Seção 5.1), apresentados neste trabalho, foram acoplados ao módulo *variator* conforme a Figura 4.3. Os operadores e funções que são representadas com cor cinza já estavam implementadas no *framework*, e os que tem o fundo de cor verde são as inseridas a partir deste trabalho. Vale lembrar que não foi proposta a inserção de nenhum operador de mutação ao *framework* PISA neste trabalho, portanto, fora apenas utilizada a mutação polinomial [15] já presente no *framework*.

Nos próximos capítulos foram avaliados e analisados os métodos aqui apresentados, através da execução de experimentos para avaliar o desempenho dos algoritmos levando em consideração a união dos métodos de seleção com os operadores de recombinação para resolver problemas de otimização no contexto multi-objetivo.

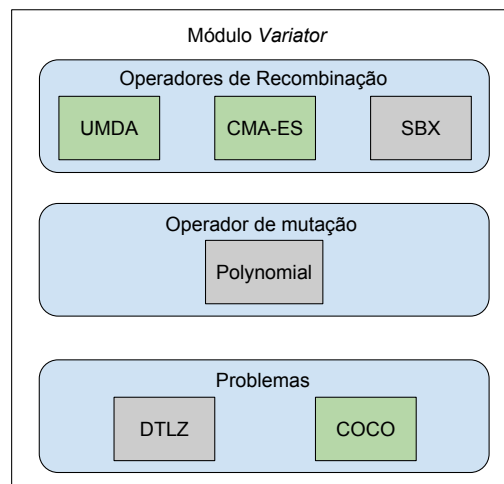


Figura 4.3: Módulo *Variator*



## Capítulo 5

# Experimentos e Resultados

Este capítulo apresenta os resultados obtidos seguindo a metodologia descrita no Capítulo 4. Os resultados são apresentados para as implementações dos diferentes modelos probabilísticos, descritos na seção 2.3.1, e para os diferentes métodos de seleção, descritos na seção 2.2. Os experimentos realizados consideraram dois diferentes conjuntos de problemas *benchmark*: conjunto de problemas da ferramenta COCO e problemas da família DTLZ, descritos a seguir nas seção 5.1.

Para a fase de experimentação foram definidos os principais objetivos, e são eles:

1. Avaliação dos diferentes modelos probabilísticos como alternativa aos tradicionais operadores de recombinação;
2. Análise do comportamento do modelos probabilísticos quando aplicados com diferentes métodos de seleção;
3. Investigar os algoritmos produzidos em respeito da qualidade dos resultados obtidos; e
4. Avaliar todos os algoritmos em diferentes conjuntos de problemas.

Avaliando os diferentes modelos probabilísticos permite uma análise de diferente perspectivas dos resultados obtidos. Podendo mostrar em que tipo de problemas cada modelo probabilístico consegue obter melhor performance. O comportamento de cada algoritmo produzido depende não apenas do modelo probabilístico, como também do método de seleção e dos problemas (funções de *fitness*).

### 5.1 Problemas de Benchmark (Funções de Fitness)

Nesta seção serão apresentados os problemas (funções de *fitness*) utilizados nesse trabalho. Cada conjunto de problemas tem diferentes complexidades em suas funções de *fitness*.

#### 5.1.1 COCO

Comparar diversos algoritmos de otimização com diferentes características é muitas vezes uma tarefa difícil e tediosa. Geralmente, os pesquisadores querem comparar os algoritmos sob diferentes domínios de dificuldade como não separabilidade, multi-modalidade e severo condicionamento, além disso é importante poder comparar diretamente algoritmos com diferentes estruturas populacionais, como estado estacionário (*steady state*), população única, multi-população e variantes de decomposição.



A ferramenta Comparador de Otimizadores Contínuos (COMparing Continuous Optimizers - COCO) [27] foi desenvolvida para automatizar a tarefa de realizar estudos experimentais cientificamente sólidos envolvendo otimizadores numéricos. A plataforma fornece problemas de *benchmark*, modelos experimentais e ferramentas para processar e visualizar o resultado de um ou mais otimizadores.

O processamento dos indicadores de qualidade baseia-se em tempos de execução, medidos em número de avaliações de *fitness* até atingir um, ou vários, valores-alvo do indicador de qualidade. Recentemente, a plataforma foi reescrita para lidar com otimizadores e problemas multi-objetivos, para isso, foi proposto um novo conjunto de problemas bi-objetivos e novos mecanismos de avaliação de desempenho [7, 61].

O novo conjunto de *benchmarks* bi-objetivo foi chamado “bbob-biobj”. Este conjunto foi criado combinando um subconjunto dos 24 problemas mono-objetivo do pacote de teste original “bbob”. Combinando as 24 funções originais sem permutações resultaria em 300 problemas. No entanto, ter tantas funções seria impraticável em termos de tempo total de execução. Assim, foram escolhidas duas funções representativas de cada um dos cinco domínios de dificuldade disponíveis para não introduzir viés em relação a qualquer domínio específico.

As combinações em pares resultaram em 55 funções bi-objetivo da versão final do conjunto chamada “bbob-biobj” [7, 61]. Mais precisamente, as 24 funções mono-objetivas originais são agrupadas em cinco classes de funções onde cada classe engloba funções com propriedades semelhantes, tais são:

- Funções separáveis (separável);
- Funções de baixo ou moderado condicionamento (moderado);
- Funções de alto condicionamento e unimodais (severas);
- Funções multi-modais com estrutura global adequada (multi-modal); e
- Funções multi-modais com fraca estrutura global (fracamente estruturada).

As 55 funções são agrupadas em 15 classes de acordo com os domínios de dificuldade de seus subproblemas componentes como apresentado na Tabela 5.1. Cada uma dessas funções é fornecida em seis dimensões (2, 3, 5, 10, 20 e 40) e com um grande número de instâncias possíveis [7].

Uma das características mais notáveis da ferramenta COCO é seu novo mecanismo de avaliação de desempenho que considera um indicador de qualidade baseado no Hypervolume do arquivo externo  $A_t$  em vez do valor objetivo de uma função mono-objetivo. Este arquivo externo contém todas as soluções não dominadas obtidas até determinado instante em um algoritmo executado. Usar um arquivo externo é uma prática relevante em aplicações do mundo real, além disso, usar um arquivo externo para avaliação de desempenho permite comparar algoritmos com tamanhos de população e estrutura diferentes. O arquivo externo é normalizado e avaliado por um indicador de qualidade que pode ser o Hypervolume, se o nadir como ponto de referência estiver dominado por pelo menos um ponto no arquivo. Caso contrário, a qualidade do arquivo é dada pela distância entre o ponto mais próximo de  $A_t$  à região de interesse delimitada pelo ponto nadir.

Os valores-alvo são baseados em uma precisão  $\Delta I$  e um valor indicador de Hypervolume de referência  $I^{ref}$ , que é uma aproximação do valor indicador de  $ICOCO_{HV}$  da Fronteira de Pareto [7]. Os resultados da avaliação de desempenho são apresentados como funções de distribuição empírica (empirical distribution function - EDF), onde a proporção de problemas resolvidos dentro de um especificado (no eixo  $x$ ) é exibida.

Tabela 5.1: Classes das funções (problemas) incluídas nas ferramenta COCO

Classe	Função	Domínio F1	Domínio F2
1	f1,f2,f11	separável	separável
2	f3,f4,f12,f13	separável	moderado
3	f5,f6,f14,f15	separável	severa
4	f7,f8,f16,f17	separável	multi-modal
5	f9,f10,f18,f19	separável	fracamente estruturado
6	f20,f21,f28	moderado	moderado
7	f22,f23,f29,f30	moderado	severa
8	f24,f25,f31,f32	moderado	multi-modal
9	f26,f27,f33,f34	moderado	fracamente estruturado
10	f35,f36,f41	severa	severa
11	f37,f38,f42,f43	severa	multi-modal
12	f39,f40,f44,f45	severa	fracamente estruturado
13	f46,f47,f50	multi-modal	multi-modal
14	f48,f49,f51,f52	multi-modal	fracamente estruturado
15	f53,f54,f55	fracamente estruturado	fracamente estruturado

### 5.1.2 DTLZ

O conjunto de problemas de *benchmark* DTLZ, criados por Deb et al. [17], são problemas multi-objetivos escaláveis para qualquer número de objetivos. Esta é uma característica importante que facilitou muitas pesquisas para a comunidade que lida com problemas de muitos objetivos (*many-objective*) [32]. Tabela 5.2 apresentar os problemas DTLZ e suas propriedades.

Tabela 5.2: Propriedades das classes das funções presentes nos problemas DTLZ

Problema	Propriedades
DTLZ1	Linear e Multimodal
DTLZ2	Concavo
DTLZ3	Concavo e Multimodal
DTLZ4	Concavo e Tendencioso
DTLZ5	Concavo e Degenerado
DTLZ6	Concavo, Degenerado e Multimodal
DTLZ7	Desconectado e Multimodal

No trabalho de Deb et al. [17], os autores sugeriram o uso de diferente números de variáveis de decisão de acordo com o problema e o número de objetivos. Foi recomendado  $n = M + k - 1$ , onde  $M$  é o número de objetivos e  $k$  depende de cada problema.  $k = 5$  é recomendado para o problema DTLZ1,  $k = 20$  é sugerido para o problema DTLZ7 e  $k = 10$  é indicado para os problemas restantes.

## 5.2 Algoritmos

Nesta seção são apresentados os nove algoritmos produzidos e utilizados nas seções a seguir do presente trabalho.

Tabela 5.3: Algoritmos produzidos

NSGA-II - SBX	SPEA2 - SBX	IBEA - SBX
NSGA-II - UMDA <sub>c</sub> <sup>G</sup>	SPEA2 - UMDA <sub>c</sub> <sup>G</sup>	IBEA - UMDA <sub>c</sub> <sup>G</sup>
NSGA-II - CMA-ES	SPEA2 - CMA-ES	IBEA - CMA-ES

Na Tabela 5.3 cada algoritmo produzido surgiu da união de um modelo probabilístico ou operador de recombinação (ver Seção 2.3) em cada linha da tabela, e um método de seleção (ver Seção 2.2) em cada coluna da tabela.

## 5.3 Configuração e Avaliação dos Algoritmos

Esta seção descreve a configuração adotada para realização dos experimentos com o intuito de comparar os modelos probabilísticos UMDA<sub>c</sub><sup>G</sup> e CMA-ES como alternativa aos tradicionais operadores de recombinação para o domínio contínuo, além de comparar o comportamento desses modelos quando aplicados com diferentes métodos de seleção extraídos de MOEAs clássicos, como: NSGA-II, SPEA2 e IBEA.

Para comparar os algoritmos, foram organizadas duas formas diferentes de avaliar os algoritmos, sendo um para os problemas da ferramenta COCO e o outro para os problemas da família DTLZ. Para os problemas da ferramenta COCO, foi utilizado o método de pós-processamento incluído na ferramenta, a qual gera tabelas e figuras conforme a média de tempo de execução (*Average Runtime* - aRT), quantidade de alvos alcançados e a função de distribuição empírica (*Empirical Cumulative Distribution Function* - EDF ou ECD) de acordo com as 10 rodadas independentes de cada algoritmo. No outro caso, para os problemas da família DTLZ, foram executadas 30 rodadas independentes tendo computado as métricas de qualidade de Hypervolume e IGD<sub>p</sub>.

Para todos os algoritmos foram definidos valores para alguns parâmetros globais. Os parâmetros são:

- Tamanho da População: 100;
- Máximo de Gerações: 500;
- Máximo de Avaliações: 50000;
- Taxa de *Crossover*: 100%;
- Taxa de Mutação: 1%;
- Índice de Distribuição ( $\eta_c$  e  $\eta_m$ ) : 20.

Vale ressaltar que quando definida a taxa de *crossover* igual a 100%, isso significa que todos os indivíduos que passarem pelo método SBX, gerarão novos indivíduos, assim permitindo uma comparação justa com os modelos probabilísticos que utilizam informações da população

por completa para extrair o modelo. O índice de distribuição  $\eta_c$  é usado em uma parte específica do operador SBX e seu resultado funciona da seguinte maneira: se um grande valor de  $\eta_c$  é definido dá-se uma maior probabilidade de criar novas soluções próximas aos pais; e se um pequeno valor de  $\eta_c$  é definido, isso permite que soluções distantes sejam geradas. Em [15] é sugerido um valor entre 15 e 20 para gerar soluções próximas aos pais. A mutação polinomial modifica uma solução para cada posição dependendo de uma determinada taxa de probabilidade usando o parâmetro  $\eta_m$ . Este parâmetro segue o mesmo contexto que é usado no operador SBX.

Todos os algoritmos foram implementados usando o *framework* PISA [6] que é uma plataforma independente para algoritmos de busca, principalmente focada em algoritmos de otimização multi-objetivo. Esta plataforma é organizado em dois grandes módulos: o *variator* e o *selector*. Estes dois módulos se comunicam entre si através de arquivos de texto contendo parâmetros em comum. Os parâmetros utilizados pelas diferentes variantes foram:

- NSGA-II, IBEA and SPEA2: torneio binário
- IBEA: indicador Hypervolume; kappa ( $k$ ) = 0.05;  $\rho$  = 1.1
- SPEA2: tamanho do arquivo = tamanho da população.

Em relação à comparação entre os diferentes operadores de recombinação, utilizamos um cenário de base para os MOEAs, onde é aplicado o operador SBX e a mutação polinomial [15] como operadores de cruzamento e mutação, respectivamente.

### 5.3.1 Comparação dos algoritmos com problemas DTLZ

O processo de comparação foi iniciado investigando o desempenho dos algoritmos produzidos para os problemas da família DTLZ.

Para avaliar a qualidade das fronteiras de Pareto geradas pelos algoritmos, dois indicadores de qualidade reconhecidos na comunidade foram utilizados: a modificação do IGD conhecido como  $IGD_p$  [55] e o Hypervolume (HV) [63]. O número de variáveis de decisão foi definida de acordo com a sugestão feita em [17].

Os resultados de  $IGD_p$  e Hypervolume, em 30 independente rodadas dos algoritmos foi submetido ao teste estatístico Kruskal-Wallis [36] considerando 5% de nível de significância. O teste de Kruskal-Wallis [36] é um teste não paramétrico baseado em ranking que pode ser utilizado para determinar se existem diferenças estatisticamente significativas entre dois ou mais grupos de uma variável independente em domínio contínuo. É importante salientar que o teste de Kruskal-Wallis não aponta especificamente quais grupos de variáveis são estatisticamente diferentes uns dos outros, apenas indica que pelo menos dois grupos são diferentes. Quando foram encontradas diferenças significativas, realizou-se uma análise utilizando o teste de Nemenyi [48] para identificar diferenças particulares entre amostras [18]. De acordo com esse teste, a eficácia de dois algoritmos é significativamente diferente sempre que seus correspondentes *rankings* médio diferirem por pelo menos um determinado valor de diferença crítica.

Como existem diversos resultados a serem comparados em diferentes situações, às vezes é difícil tirar conclusões gerais. Para facilitar essa visualização, elaborou-se tabelas resumidas desconsiderando funções específicas e o números de objetivos, e focando na análise nas diferenças entre os algoritmos. Estas tabelas são geradas para cada indicador ( $IGD_p$  e Hypervolume) usando o teste estatístico de Friedman [23] também considerando um nível de significância de 5%, nas médias das 30 rodadas de cada subproblema (problema/número de objetivos).

A comparação entre os diferentes componentes do algoritmo utilizando os dois testes estatísticos é apresentada em tabelas contendo os rankings médios dos resultados obtidos em 30 rodadas do algoritmo usando cada componente. Atribuímos classificações finais aos algoritmos (apresentados entre parênteses) de acordo com suas médias. No caso de um empate estatístico (algoritmos que não apresentam diferença estatisticamente significativa), a classificação final de cada um dos algoritmos com resultados similares é igual à média dos rankings que seriam atribuídos. O(s) algoritmo(s) com os menores ranking finais são destacados em negrito.

Os resultados obtidos pelos algoritmos são apresentados nas tabelas 5.4 e 5.6, para  $IGD_p$  e HV, respectivamente. Nessas tabelas, cada coluna representa o desempenho relativo obtido por um algoritmo e cada linha indica um número específico de objetivos de um problema.

Em relação aos resultados da  $IGD$  para dois objetivos, os algoritmos que utilizam modelos probabilísticos (CMA-ES e UMDA<sup>G</sup>), tiveram bom desempenho na maioria dos problemas. Eles só perderam no problema DTLZ4 para SPEA2-SBX e empatado no problema DTLZ3 com IBEA-SBX. Para três objetivos, os algoritmos que utilizam PM tiveram desempenho semelhante entre eles, obtendo os melhores rankings em todos os problemas DTLZ, com uma exceção em DTLZ7 que tem como vencedor o algoritmo IBEA-SBX.

Tabela 5.4: Resultados do teste de Kruskal-Wallis com a média da posição de cada uma das 30 rodadas (e ranking entre os algoritmos) obtidos para as diferentes variantes dos MOEAs com base na métrica  $IGD_p$ .

Obj.	Algorithms	DTLZ1	DTLZ2	DTLZ3	DTLZ4	DTLZ5	DTLZ6	DTLZ7
2	IBEA-CMA-ES	126.53 (5.00)	93.15 (3.50)	106.62 (3.50)	103.63 (4.00)	<b>65.50 (2.50)</b>	<b>69.08 (2.50)</b>	139.87 (4.50)
	SPEA2-CMA-ES	139.60 (5.50)	205.10 (7.50)	176.07 (7.50)	196.75 (7.50)	216.97 (8.50)	204.58 (7.50)	<b>95.90 (4.00)</b>
	NSGA-II-CMA-ES	162.70 (6.00)	183.40 (6.50)	167.57 (6.50)	214.32 (8.00)	176.53 (6.50)	195.52 (7.00)	<b>89.18 (4.00)</b>
	IBEA-UMDA	108.72 (4.00)	<b>68.70 (2.50)</b>	107.73 (3.50)	97.72 (4.00)	107.13 (4.00)	138.58 (5.00)	143.63 (5.00)
	SPEA2-UMDA	<b>61.10 (2.00)</b>	135.88 (5.00)	<b>97.80 (3.00)</b>	160.22 (5.50)	142.47 (5.00)	97.23 (4.00)	169.38 (6.00)
	NSGA-II-UMDA	94.87 (3.50)	159.22 (6.00)	<b>97.80 (3.00)</b>	145.50 (4.50)	148.72 (5.00)	97.55 (4.00)	203.30 (8.00)
	IBEA-SBX	157.15 (5.50)	97.57 (4.00)	<b>99.98 (3.00)</b>	101.53 (4.00)	95.00 (4.00)	150.53 (5.50)	126.87 (4.50)
	SPEA2-SBX	195.22 (7.00)	129.42 (4.50)	184.47 (7.50)	<b>91.73 (3.50)</b>	125.93 (4.50)	153.00 (5.50)	116.93 (4.50)
	NSGA-II-SBX	173.62 (6.50)	147.07 (5.50)	181.47 (7.50)	108.10 (4.00)	141.25 (5.00)	113.42 (4.00)	134.43 (4.50)
3	IBEA-CMA-ES	82.75 (3.50)	<b>81.80 (3.50)</b>	<b>83.13 (3.50)</b>	<b>111.98 (4.00)</b>	<b>46.98 (3.00)</b>	<b>34.17 (2.00)</b>	134.75 (4.50)
	SPEA2-CMA-ES	131.37 (4.50)	<b>116.73 (3.50)</b>	<b>96.55 (3.50)</b>	181.03 (7.50)	123.15 (4.50)	102.20 (4.00)	135.22 (4.50)
	NSGA-II-CMA-ES	135.02 (4.50)	<b>76.30 (3.50)</b>	<b>90.65 (3.50)</b>	<b>116.00 (4.00)</b>	109.30 (3.50)	95.15 (3.50)	164.57 (6.50)
	IBEA-UMDA	79.77 (3.50)	<b>74.20 (3.50)</b>	<b>108.30 (3.50)</b>	<b>114.00 (4.00)</b>	<b>56.45 (3.00)</b>	111.22 (4.00)	136.90 (4.50)
	SPEA2-UMDA	<b>65.10 (2.50)</b>	<b>108.37 (3.50)</b>	<b>89.00 (3.50)</b>	<b>118.00 (4.00)</b>	101.00 (3.50)	132.40 (4.50)	217.82 (8.00)
	NSGA-II-UMDA	<b>54.37 (2.50)</b>	<b>85.60 (3.50)</b>	<b>86.97 (3.50)</b>	<b>99.27 (4.00)</b>	106.12 (3.50)	67.87 (3.00)	225.00 (8.00)
	IBEA-SBX	244.27 (8.00)	203.63 (8.00)	238.00 (8.00)	144.47 (5.00)	208.90 (8.00)	255.50 (8.00)	<b>40.20 (2.00)</b>
	SPEA2-SBX	202.27 (8.00)	232.97 (8.00)	195.27 (8.00)	206.97 (8.00)	234.20 (8.00)	214.23 (8.00)	82.97 (3.50)
	NSGA-II-SBX	224.60 (8.00)	239.90 (8.00)	231.63 (8.00)	127.78 (4.50)	233.40 (8.00)	206.77 (8.00)	82.08 (3.50)

Na análise global, considerando todos os problemas e números de objetivos, apresentados na Tabela 5.5, não podemos identificar qualquer diferença estatisticamente significativa entre as variantes comparadas. Portanto, não podemos apontar os melhores, mas analisando apenas os resultados classificados, podemos observar que a modelagem probabilística (CMA-ES e UMDA<sup>G</sup>) como operadores de recombinação atingiu os menores valores com o método de seleção IBEA. Além disso, podemos destacar que entre os grupos de operadores de recombinação, todos realizaram o melhor usando IBEA, no entanto os métodos PM melhor desempenho usando

o NSGAII do que usando o SPEA2, ao contrário do SBX que melhor desempenho com SPEA2 do que com NSGAII.

Tabela 5.5: Resultados do teste de Friedman (e ranking) para as diferentes variantes dos MOEAs entre problemas e número de objetivos com base na métrica  $IGD_p$ .

IBEA-CMA-ES	SPEA2-CMA-ES	NSGA-II-CMA-ES	IBEA-UMDA	SPEA2-UMDA	NSGA-II-UMDA	IBEA-SBX	SPEA2-SBX	NSGA-II-SBX
<b>45.0 (5.0)</b>	<b>84.0 (5.0)</b>	<b>80.0 (5.0)</b>	<b>45.0 (5.0)</b>	<b>64.0 (5.0)</b>	<b>57.0 (5.0)</b>	<b>81.0 (5.0)</b>	<b>85.0 (5.0)</b>	<b>89.0 (5.0)</b>

A análise dos resultados de Hypervolume para dois objetivos mostrados na Tabela 5.6 mostra que os algoritmos que usam modelagem probabilística têm um bom desempenho. Eles alcançaram os melhores rankings em todos os problemas DTLZ, exceto para DTLZ4, onde foram superados por SPEA2-SBX. Para problemas com três objetivos, os algoritmos que aplicaram modelagem probabilística têm bom desempenho também, alcançando os melhores rankings em todos os problemas DTLZ.

Tabela 5.6: Resultados do teste de Kruskal-Wallis com a média da posição de cada uma das 30 rodadas (e ranking entre os algoritmos) obtidos para as diferentes variantes dos MOEAs com base na métrica Hypervolume.

Obj.	Algorithms	DTLZ1	DTLZ2	DTLZ3	DTLZ4	DTLZ5	DTLZ6	DTLZ7
2	IBEA-CMA-ES	109.72 (3.50)	101.00 (4.00)	99.13 (3.50)	112.67 (4.00)	<b>71.67 (3.00)</b>	<b>73.13 (2.50)</b>	109.00 (3.50)
	SPEA2-CMA-ES	129.13 (5.00)	208.80 (8.00)	170.40 (6.50)	197.17 (7.50)	212.13 (8.50)	209.13 (7.50)	210.87 (7.50)
	NSGA-II-CMA-ES	162.73 (6.00)	180.73 (6.50)	187.27 (7.50)	211.40 (7.50)	182.40 (6.50)	191.43 (7.00)	187.47 (7.50)
	IBEA-UMDA	104.63 (3.50)	<b>76.17 (3.00)</b>	110.33 (4.00)	104.33 (4.00)	119.03 (4.00)	148.40 (5.50)	148.97 (6.00)
	SPEA2-UMDA	<b>64.20 (2.50)</b>	130.68 (4.50)	<b>95.05 (3.00)</b>	158.48 (5.50)	130.67 (4.50)	95.67 (4.00)	73.57 (3.00)
	NSGA-II-UMDA	91.83 (3.00)	152.83 (5.50)	<b>92.35 (3.00)</b>	149.57 (5.50)	143.10 (5.00)	97.90 (4.00)	<b>58.33 (2.50)</b>
	IBEA-SBX	183.63 (7.00)	102.40 (4.00)	123.30 (4.50)	105.62 (4.00)	98.73 (4.00)	154.37 (5.50)	114.57 (3.50)
	SPEA2-SBX	192.63 (7.50)	127.78 (4.50)	160.13 (6.00)	<b>71.90 (3.00)</b>	123.97 (4.50)	139.17 (5.00)	192.20 (7.50)
	NSGA-II-SBX	180.98 (7.00)	139.10 (5.00)	181.53 (7.00)	108.37 (4.00)	137.80 (5.00)	110.30 (4.00)	124.53 (4.00)
3	IBEA-CMA-ES	75.70 (3.00)	<b>82.73 (3.50)</b>	<b>87.50 (3.50)</b>	126.97 (4.50)	<b>53.40 (3.00)</b>	<b>37.67 (2.00)</b>	99.95 (4.50)
	SPEA2-CMA-ES	123.30 (4.00)	<b>114.47 (3.50)</b>	<b>105.65 (3.50)</b>	168.93 (5.50)	127.57 (4.50)	107.57 (4.00)	129.82 (4.50)
	NSGA-II-CMA-ES	147.07 (6.00)	<b>86.70 (3.50)</b>	<b>94.42 (3.50)</b>	121.00 (4.50)	104.47 (3.50)	91.70 (3.50)	133.90 (4.50)
	IBEA-UMDA	74.73 (3.00)	<b>77.80 (3.50)</b>	<b>110.67 (3.50)</b>	108.63 (4.50)	<b>61.37 (3.00)</b>	114.27 (4.00)	111.33 (4.50)
	SPEA2-UMDA	71.30 (3.00)	<b>96.83 (3.50)</b>	<b>78.18 (3.50)</b>	117.87 (4.50)	101.67 (3.50)	125.97 (4.00)	<b>34.72 (1.50)</b>
	NSGA-II-UMDA	<b>56.43 (2.50)</b>	<b>84.47 (3.50)</b>	<b>81.98 (3.50)</b>	<b>101.73 (4.00)</b>	94.53 (3.50)	65.83 (3.50)	<b>35.48 (1.50)</b>
	IBEA-SBX	245.43 (8.00)	244.93 (8.00)	245.23 (8.00)	144.33 (5.00)	244.93 (8.00)	255.50 (8.00)	216.57 (8.00)
	SPEA2-SBX	201.87 (7.50)	204.13 (8.00)	187.97 (8.00)	192.77 (7.50)	204.13 (8.00)	212.30 (8.00)	232.70 (8.00)
	NSGA-II-SBX	223.67 (8.00)	227.43 (8.00)	227.90 (8.00)	137.27 (5.00)	227.43 (8.00)	208.70 (8.00)	225.03 (8.00)

Ao considerar uma análise global usando todos os problemas e números objetivos, conforme apresentado nas Figura 5.1 e Tabela 5.7, é possível ver que as melhores classificações foram alcançadas pelo IBEA-CMA-ES e NSGA-II-UMDA. Ao contrário dos resultados da IGD, ao considerar o HV, o UMDA apresentou melhor desempenho com o NSGA-II, porém, como nos resultados anteriores, todos os métodos PM apresentaram pior desempenho com SPEA2 do que com todos os outros métodos de seleção. Ao considerar o operador SBX, este obteve desempenho melhor usando SPEA2 do que usando os outros métodos de seleção.

Importante ressaltar que os algoritmos que utilizam UMDA alcançaram melhores resultados que o CMA-ES na maioria dos problemas. Como pode ser observado na Tabela 5.6

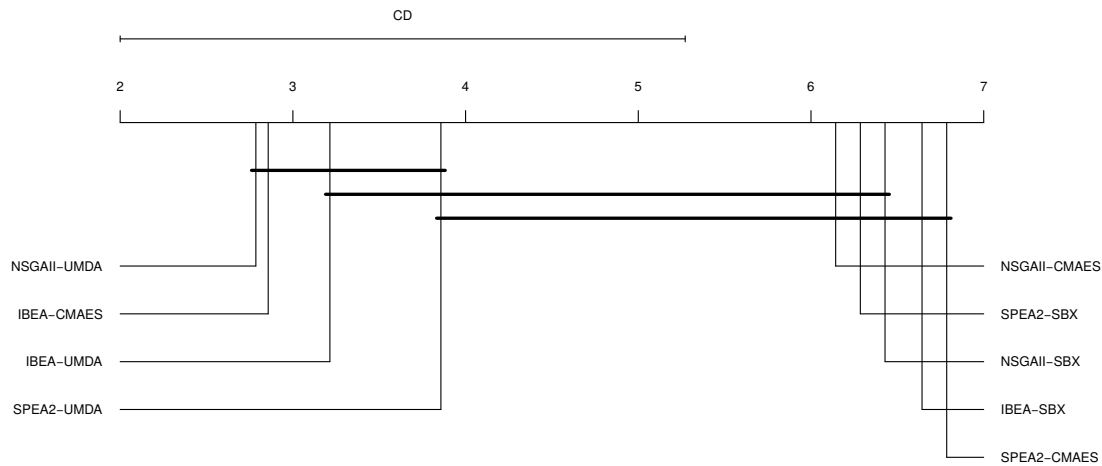


Figura 5.1: Diagrama do pós-teste de Nemenyi para Hypervolume considerando 2 e 3 objetivos.

Tabela 5.7: Resultados do teste de Friedman (e ranking) para as diferentes variantes dos MOEAs entre problemas e número de objetivos com base na métrica Hypervolume.

IBEA-CMA-ES	SPEA2-CMA-ES	NSGA-II-CMA-ES	IBEA-UMDA	SPEA2-UMDA	NSGA-II-UMDA	IBEA-SBX	SPEA2-SBX	NSGA-II-SBX
<b>40.0 (2.5)</b>	95.0 (6.5)	86.0 (6.0)	45.0 (3.5)	54.0 (5.0)	<b>39.0 (2.5)</b>	93.0 (6.5)	88.0 (6.0)	90.0 (6.5)

os algoritmos com UMDA obtiveram os melhores resultados em quatro dos sete algoritmos para dois objetivos levando em consideração o indicador de Hypervolume. Quando considerado três objetivos, em todos os problemas os algoritmos UMDA alcançaram os melhores resultados. Por fim, a análise em conjunto da 5.5 e Tabela 5.7 suporta a afirmação de que um algoritmo que utiliza um modelo simples pode se sobressair a outros mais complexos dependendo do problema em questão a ser otimizado.

Nas Figuras 5.2 até 5.5 são apresentados, em específico, os resultados obtidos nas tabelas anteriores, com o intuito de proporcionar uma melhor visualização dos valores obtidos para cada métrica de qualidade, e também em relação a quantidade de objetivos. Foi escolhido o problema DTLZ7 para mostrar a diferença encontrada entre as soluções dos algoritmos.

É visível nas Figuras 5.2 e 5.4, e nas Figuras 5.3 e 5.5 que existe a interferência nos resultados a partir da adição de mais um objetivo (2 para 3) em ambas as métricas de qualidade. Vale ressaltar que quanto maior o valor de Hypervolume, melhor, em contrapartida, para o  $IGD_p$  quanto menor o valor, melhor.

### 5.3.2 Comparação dos algoritmos com problemas da ferramenta COCO

Na segunda análise, foram utilizados os problemas da ferramenta COCO para avaliar o desempenho dos MOEAs tradicionais e as variantes que aplicam PM. Como mostrado na Tabela 5.1, este conjunto de testes é composto por uma variedade de problemas bi-objetivos, incluindo funções de diferentes níveis de dificuldade. Portanto, a seguir é apresentada a comparação entre os MOEA tradicionais (SBX) e os algoritmos produzidos utilizando PM.

Os experimentos executados com o ferramenta COCO foram avaliados em todos algoritmos para todas as 55 funções. Além disso, para avaliar a escalabilidade no número de variáveis, foi considerado usar todas as seguintes dimensões,  $DIM \in \{2, 3, 5, 10, 20, 40\}$ , dispostas na ferramenta. A métrica utilizada para comparar os algoritmos é a distribuição cumulativa empírica (ECD) do tempo de execução para um número predefinido de valores-alvo em relação

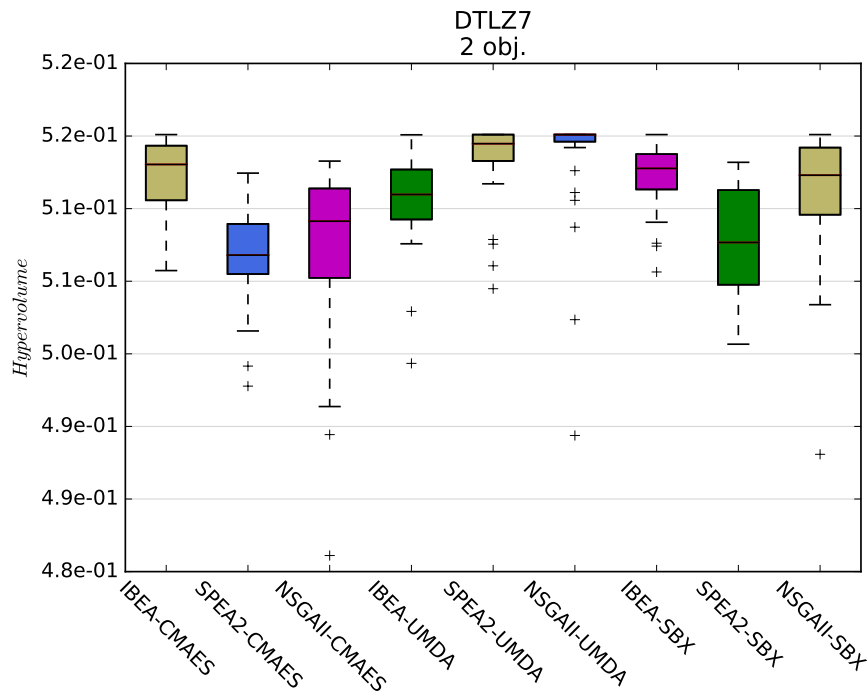


Figura 5.2: Gráfico *boxplot* para métrica Hypervolume com 2 objetivos

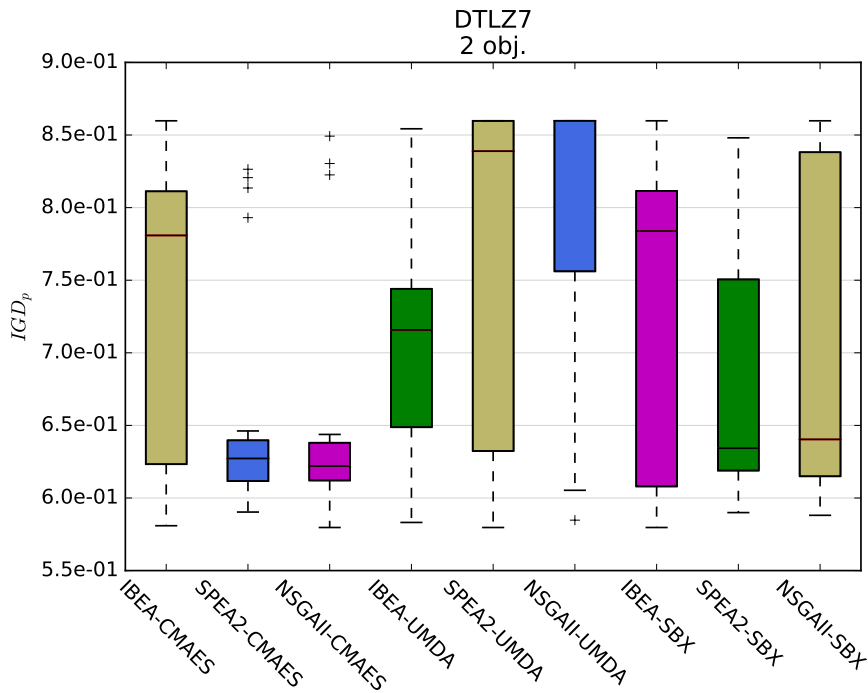


Figura 5.3: Gráfico *boxplot* para métrica  $IGD_p$  com 2 objetivos

ao indicador de qualidade ( $I_{HV}^{COCO}$ ). Basicamente, essa métrica mede quantas avaliações de *fitness* são necessárias para que um algoritmo alcance uma aproximação de fronteira de Pareto que aproxima o valor do indicador de qualidade para um determinado valor-alvo. O melhor valor da métrica é 1, significando que o algoritmo foi capaz de atingir todos os 58 valores-alvo que são cada vez mais rigorosos. O algoritmo que atinge esses objetivos de precisão com menos



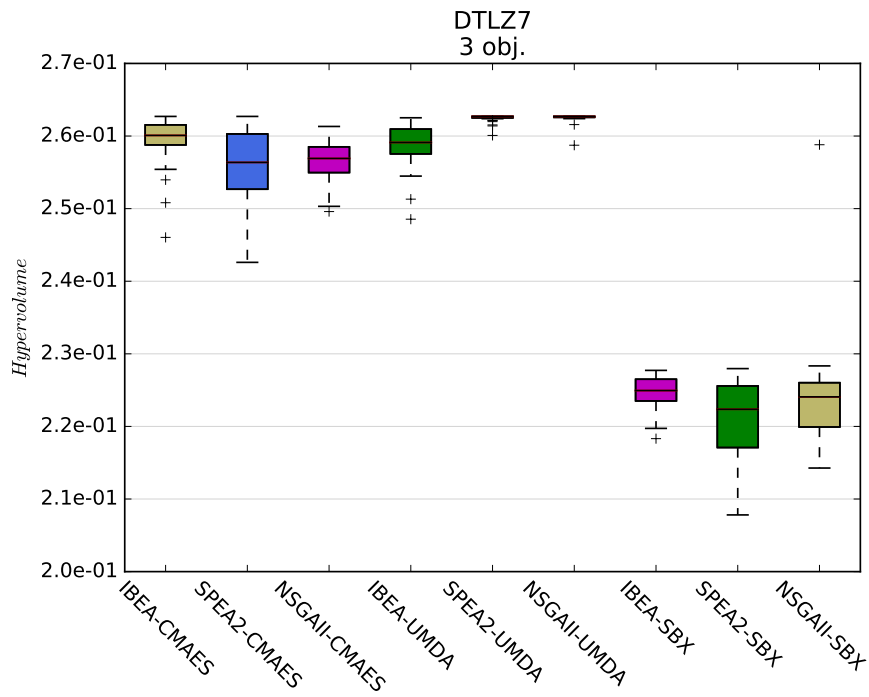


Figura 5.4: Gráfico *boxplot* para métrica Hypervolume com 3 objetivos

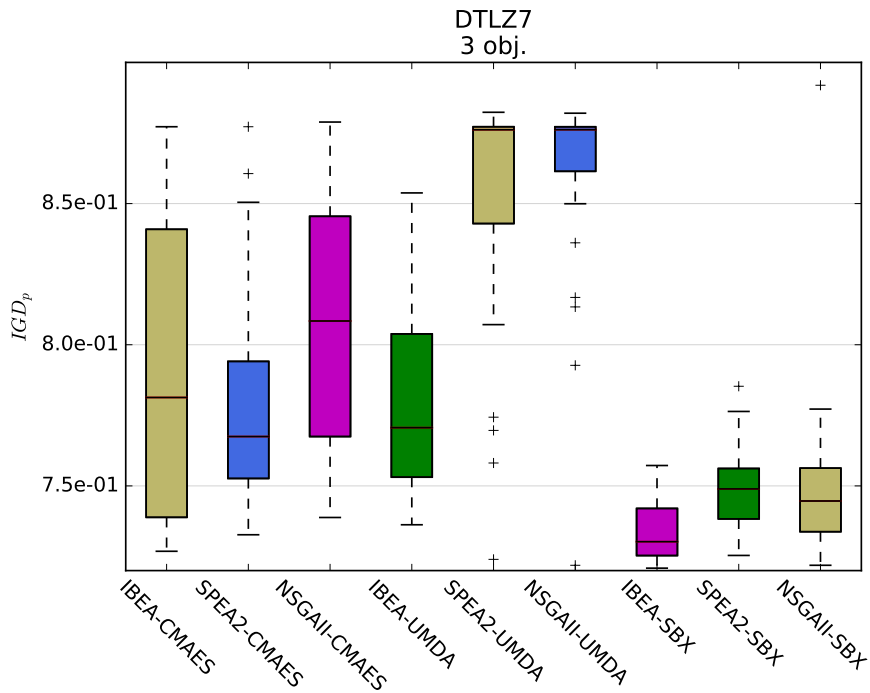


Figura 5.5: Gráfico *boxplot* para métrica  $IGD_p$  com 3 objetivos

avaliações de função é considerado mais eficiente. Figura 5.6 apresenta os resultados obtidos por todos os algoritmos. Cada sub-figura corresponde aos resultados de uma das seis dimensões consideradas.

O primeiro padrão que pode ser considerado a partir da análise das figuras é que o operador SBX tem desempenho pior do que algoritmos que aplicam modelos probabilísticos em

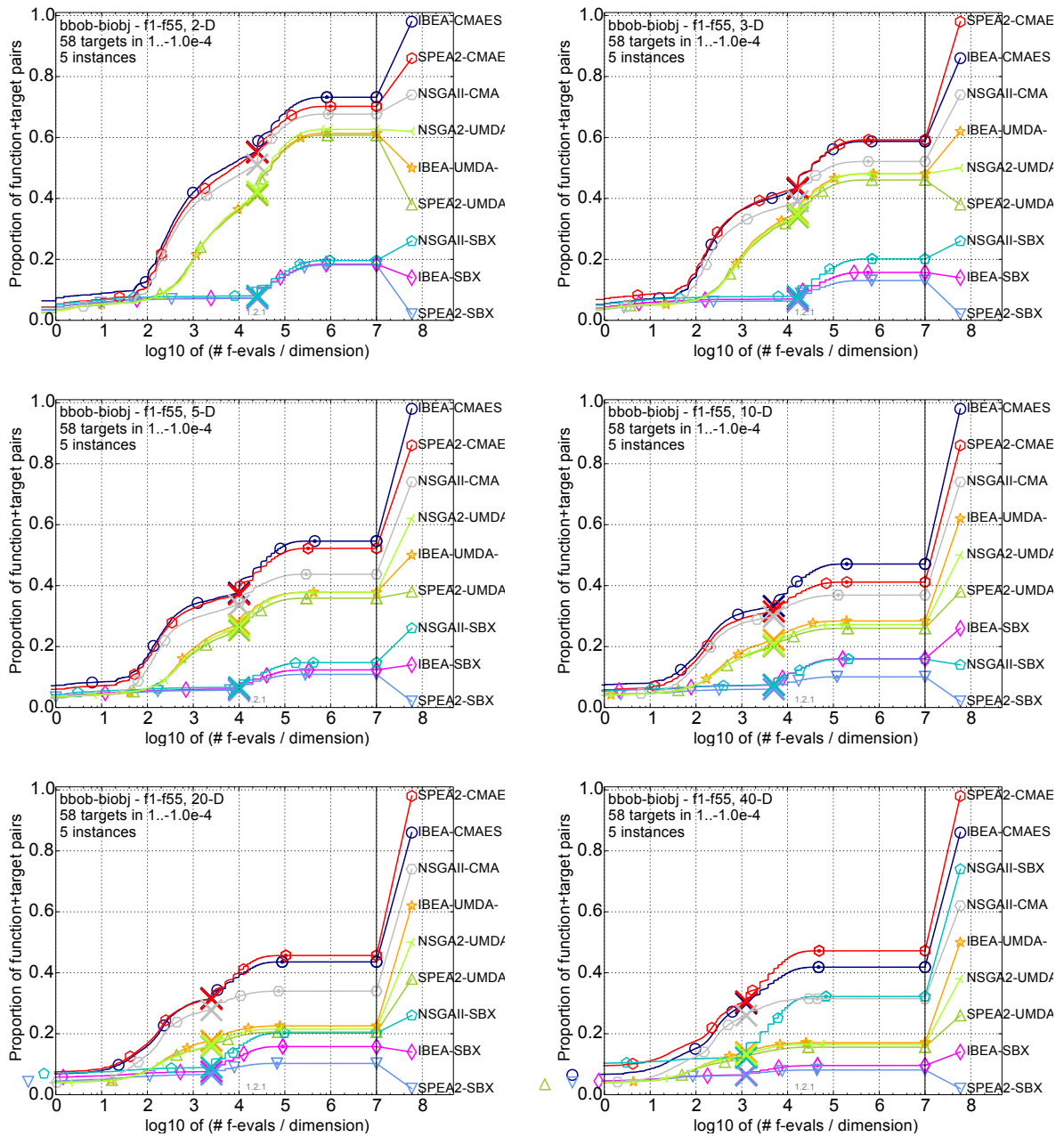


Figura 5.6: Distribuição acumulada empírica (ECD) de valores-alvos alcançados no tempo de execução simulado, medido em número de avaliações de *fitness* pela dimensão do espaço de decisão (Aval/Dim) para os 58 alvos de todos algoritmos.  $\{-10^{-4}, -10^{-4.2}, -10^{-4.4}, -10^{-4.6}, -10^{-4.8}, -10^{-5}, 0, 10^{-5}, 10^{-4.9}, 10^{-4.8}, \dots, 10^{-0.1}, 10^0\}$ . Resultados para todas as 55 funções bi-objetivas e todas as dimensões consideradas.

todas as dimensões. A proporção de valor-alvos que as variantes SBX são capazes de alcançar em nenhum momento é superior a 0.2. Isto é perceptível para dimensões pequenas ( $DIM \leq 5$ ) para as quais a diferença com UMDA é clara. O segundo fato dessa análise é que o CMA-ES supera todos os outros algoritmos, alcançando limiar acima de 0.7 de precisão para  $DIM = 2$  e acima de 0.4 em todas as dimensões.

Como esperado, quando o número de dimensões é incrementado a proporção de valores-alvo atingidos por todos os algoritmos é menor. O terceiro fato que os resultados dos experimentos

revelam é bastante relevante para os tópicos tratados neste artigo. Enquanto IBEA e SPEA são consistentemente as melhores estratégias de seleção para CMAES, e IBEA é também a melhor estratégia para UMDA, em contra partida, os resultados para SBX são diferentes. Para o operador tradicional (SBX), o método de seleção NSGA-II produz melhores resultados do que os outros métodos. A influência do operador de seleção é tão importante que para as dimensões 20 e 40 o NSGA-II-SBX tem resultado competitivo ou supera as variantes UMDA.

Prosseguindo agora com a análise de diferentes classes de funções desta ferramenta (ver Tabela 5.1), iniciando com a menor dimensão  $DIM = 2$  possível. Figura 5.7 apresenta os resultados obtidos por todos os algoritmos. Cada sub-figura corresponde aos resultados onde a mesma classe de funções eram avaliadas, assim 5 classes foram consideradas (por exemplo, separável-separável, moderado-moderado, etc).

Para analisar a escalabilidade do algoritmo, foi verificado como o desempenho dos algoritmos muda para as outras dimensões consideradas nos experimentos. Estes resultados são mostrados nas Figura 5.8 e Figura 5.9 (com  $DIM = 3$  e  $DIM = 5$ , respectivamente). Como esperado, diferentes taxas de alvos alcançados foram encontradas quando analisados os algoritmos para as outras dimensões consideradas, inclusive é perceptível a presença de uma queda nos valores de alvos atingidos comparando os resultados de  $DIM = 2$ ,  $DIM = 3$  e  $DIM = 5$ . Essa piora não é tão clara para os problemas separáveis, porém quando analisados os mais severos (severa, multi-modal e fracamente estruturado), é visível o deterioramento na proporção de alvos atingidos.

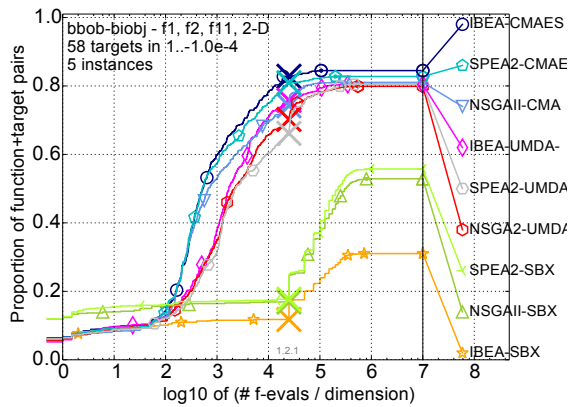
Como um passo final na investigação dos algoritmos, foi estudada a variabilidade dos comportamentos do algoritmo dentro das classes mais fáceis e mais difíceis de problemas. Com o objetivo de determinar se o bom (ou mau) desempenho para uma determinada classe de problemas é devido a uma única função em cada classe, ou se todas as funções dentro de uma classe produzem um impacto semelhante no comportamento do algoritmo. Neste passo, foram utilizados apenas os "melhores" algoritmos de cada estratégia de recombinação (ou modelos probabilísticos) de acordo com a Figura 5.6 quando considerado a experimentação no geral. Portanto, foram escolhidos: IBEA-CMA-ES, IBEA-UMDA e NSGA-II-SBX. A Figura 5.10 apresenta a variabilidade para as classes separável - separável e multi-modal - multi-modal para  $DIM = 2$ . Cada sub-figura esta relacionada a uma classe de funções e a um dos três algoritmos acima mencionados. Nota-se que em ambos os casos existe variabilidade entre as funções, porém é claro que a variabilidade depende também do algoritmo, pois para os algoritmos IBEA-CMA-ES e IBEA-UMDA é notável a pouca diferença entre as funções para os problemas separáveis e multi-modais, respectivamente.

Estes resultados mostram que a escolha do operador de seleção deve considerar as características do método utilizado para gerar novas soluções. Diferentes operadores de recombinação podem exigir diferentes maneiras de selecionar as melhores soluções. Um aprendizado retirado desta análise é que o melhor método de seleção encontrado para MOEAs tradicionais não é necessariamente o melhor algoritmo para abordagens que usam modelos probabilísticos.

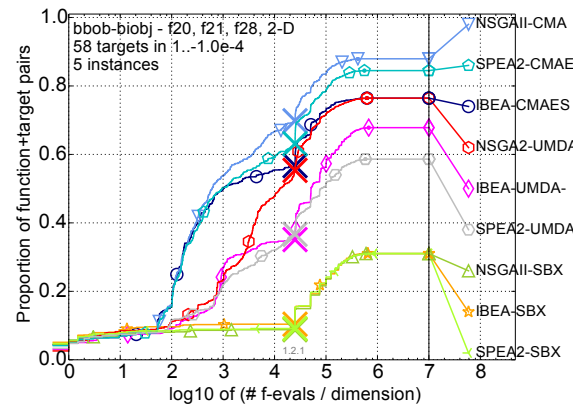
## 5.4 Discussão

Em um resumo dos resultados anteriormente apresentados, pode-se destacar que o método de seleção tem um impacto em todos os operadores de recombinação, porém a influência muda de acordo com os diferentes métodos de geração de soluções, modelos probabilísticos e SBX.

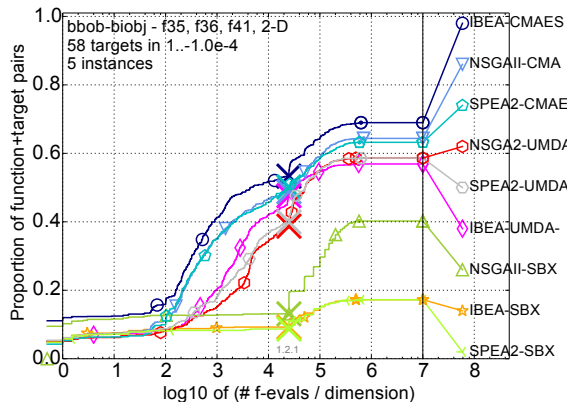
Outro fato interessante é que em problemas mais fáceis com menos dependências entre as variáveis, como na família DTLZ, o UMDA apresenta um desempenho muito competitivo em



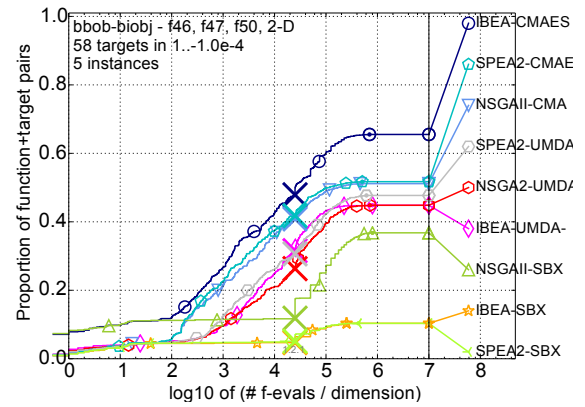
(a): separável - separável



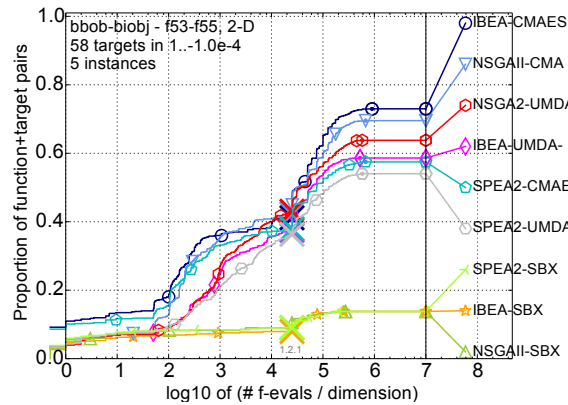
(b): moderado - moderado



(c): severa - severa



(d): multi-modal - multi-modal



(e): fracamente estruturado - fracamente estruturado

Figura 5.7: ECD para todos os algoritmos, como descrito na Figura 5.6. Resultados para as funções bi-objetivas que pertencem as mesmas classes e  $DIM = 2$ .

relação ao outro método que executa cálculos mais avançados, o CMA-ES. No entanto quando consideramos problemas mais difíceis, como a família COCO, a superioridade do algoritmo CMA-ES é perceptível, tal que supera todos os outros algoritmos no geral.

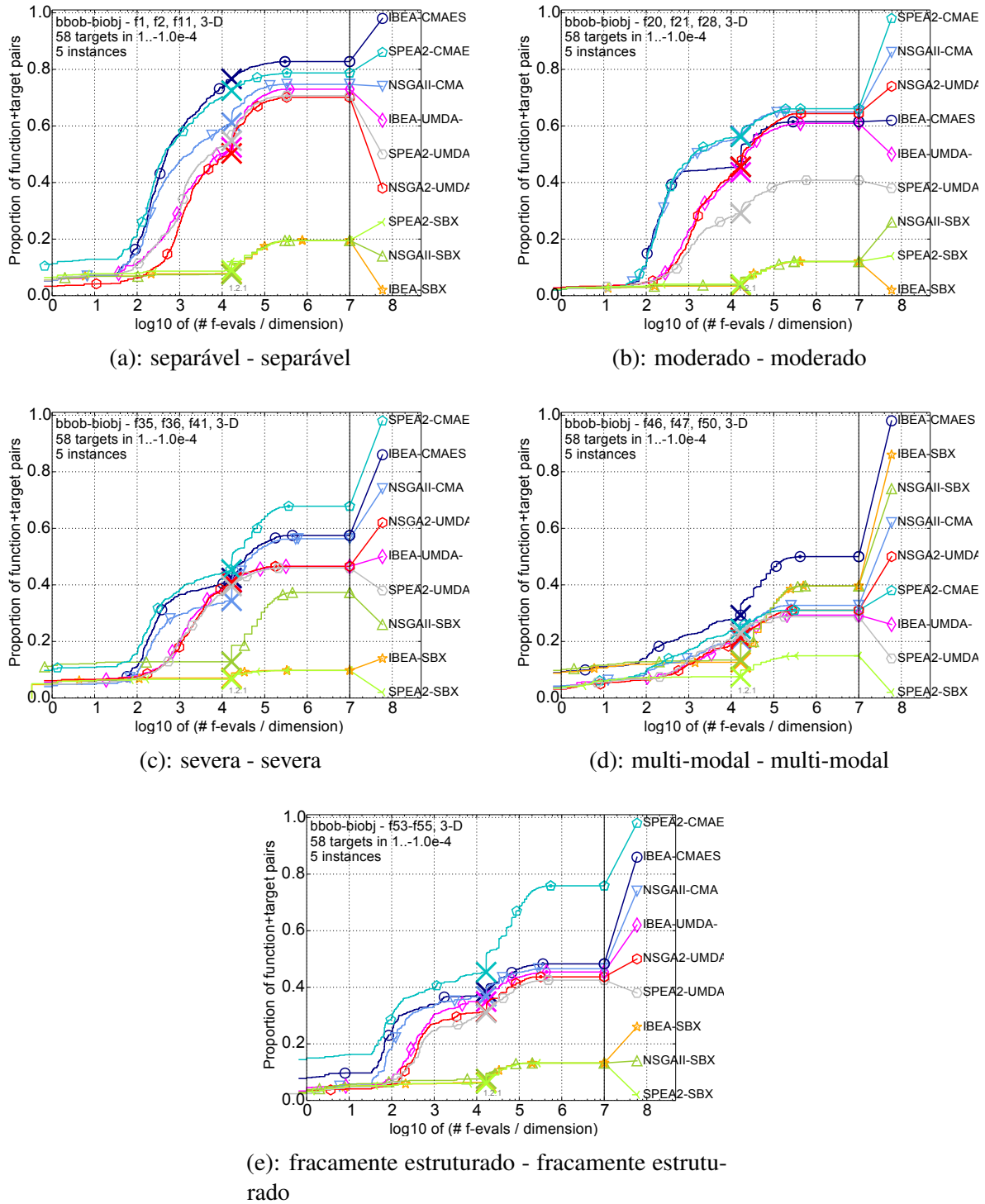
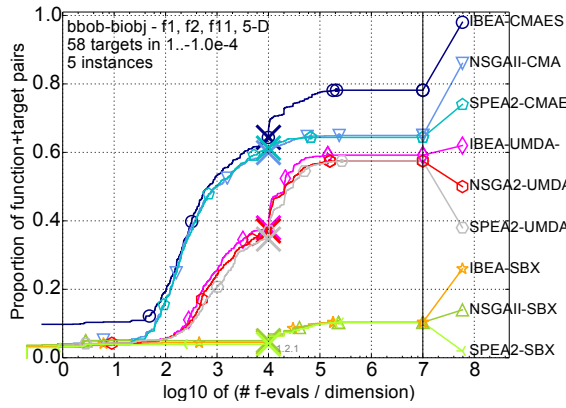
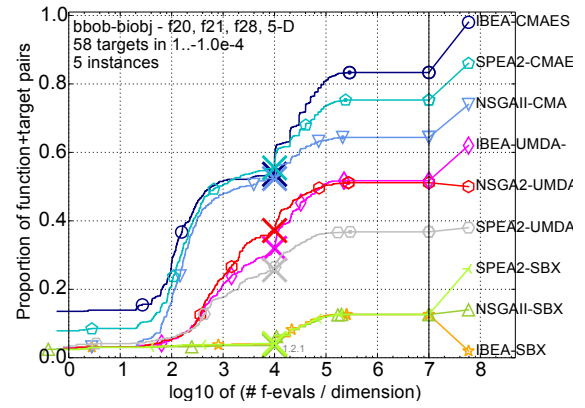


Figura 5.8: ECD para todos os algoritmos, como descrito na Figura 5.6. Resultados para as funções bi-objetivas que pertencem as mesmas classes e  $DIM = 3$ .

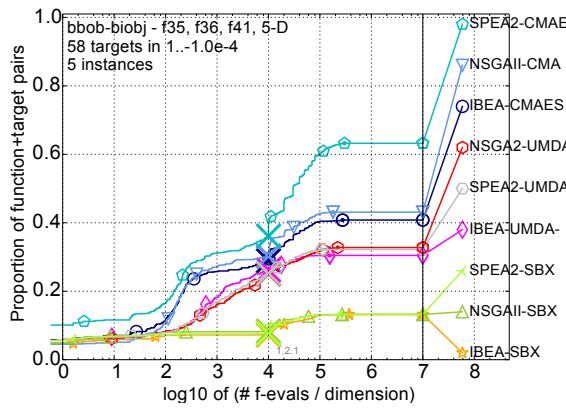
Além disso, pode-se confirmar que tanto nos problemas mais fáceis quanto nos mais difíceis, o uso de EDAs é recomendado, uma vez que eles foram capazes de superar o operador clássico SBX na grande maioria dos estudos envolvendo os dois conjuntos de problemas distintos.



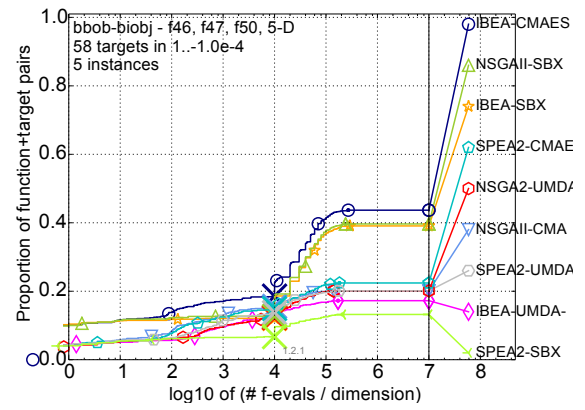
(a): separável - separável



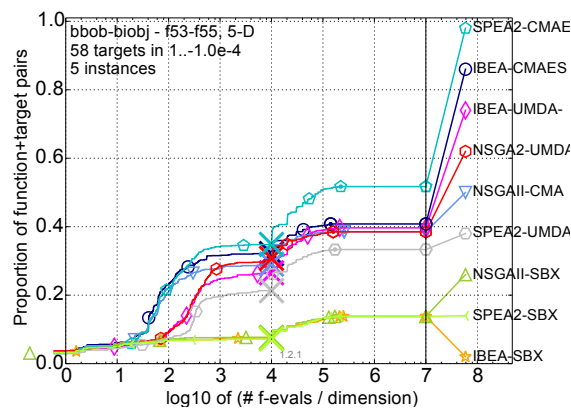
(b): moderado - moderado



(c): severa - severa

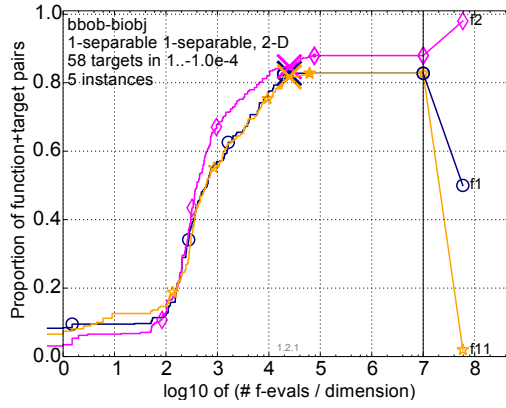


(d): multi-modal - multi-modal

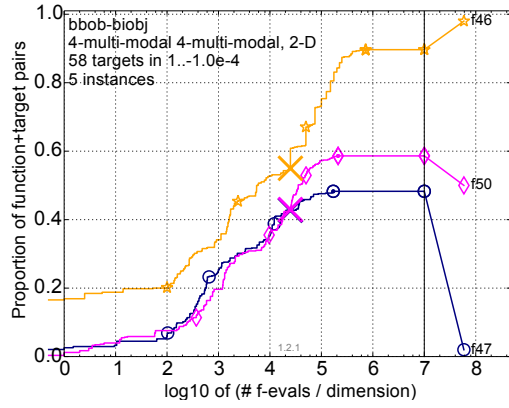


(e): fracamente estruturado - fracamente estruturado

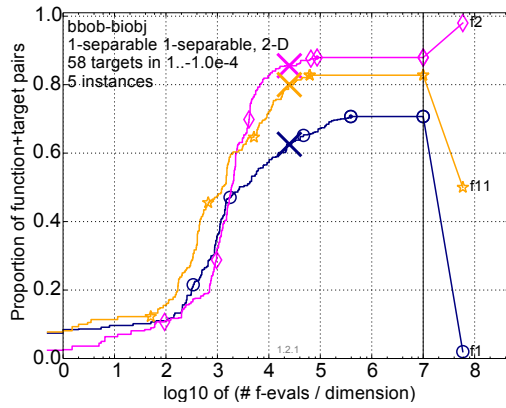
Figura 5.9: ECD para todos os algoritmos, como descrito na Figura 5.6. Resultados para as funções bi-objetivas que pertencem as mesmas classes e  $DIM = 5$ .



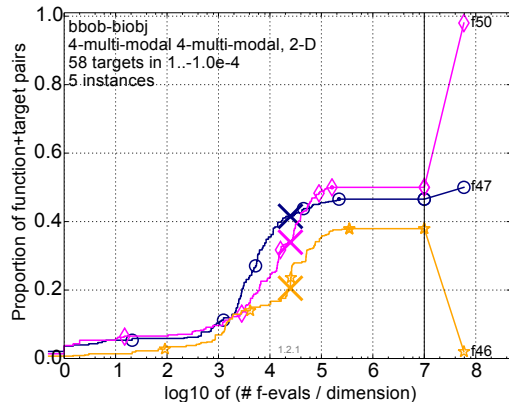
(a): IBEA-CMA-ES separável



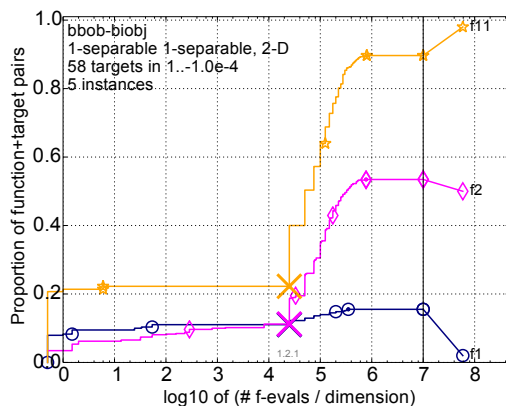
(b): IBEA-CMA-ES multi-modal



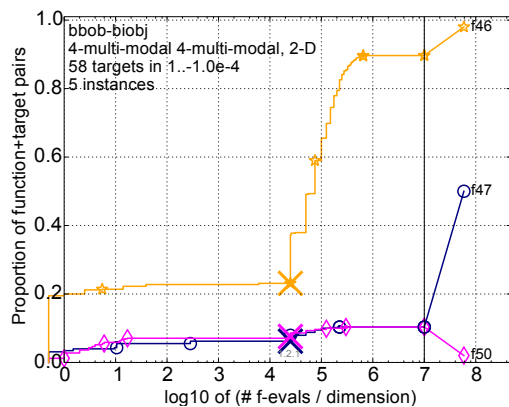
(c): IBEA-UMDA separável



(d): IBEA-UMDA multi-modal



(e): NSGA-II-SBX separável



(f): NSGA-II-SBX multi-modal

Figura 5.10: ECD para os algoritmos IBEA-CMA-ES, IBEA-UMDA e NSGA-II-SBX, respectivamente, como descrito na Figura 5.6. Resultados para as classes de funções separável - separável e multi-modal - multi-modal com  $DIM = 2$ .

## Capítulo 6

### Conclusão e Trabalhos Futuros

Neste trabalho foram apresentados alguns conceitos gerais sobre EDAs, especialmente para o contexto multi-objetivo (MOEDAs). MOEAs que são compostos por diferentes métodos de seleção, substituição e arquivamento foram utilizados com o objetivo de investigar o desempenho de cada um quando aplicados com operadores de recombinação alternativos aos tradicionais para o domínio contínuo.

Dentre os fatores que influenciam o comportamento dos MOEAs existem dois que têm forte impacto no desempenho dos algoritmos: O método utilizado para selecionar as soluções que serão utilizadas para gerar as novas populações, e a forma como essas novas populações são geradas. Embora essas duas questões possam ser estudadas independentemente, elas realmente interagem para determinar a dinâmica de um MOEA. Neste trabalho, investigou-se como essa interação é expressa nos resultados de diferentes variantes de MOEAs. Foi considerado um conjunto de estratégias de seleção de MOEAs clássicos combinado com um conjunto de métodos para gerar novas soluções. As estratégias de seleção incluíram os renomados e amplamente utilizados algoritmos NSGA-II, SPEA2 e IBEA. Os métodos para gerar novas soluções foram o *crossover* SBX, um operador de recombinação usado por MOEAs tradicionais e dois métodos baseados em modelagem probabilística: UMDA<sup>G</sup> e CMA-ES.

Para avaliar empiricamente todas as variantes dos algoritmos, foram empregados os problemas de *benchmark* da família DTLZ e da ferramenta COCO. Dois conjuntos de problemas compostos por MOPs com 2 e 3 objetivos, e um grande conjunto de funções que representam diferentes complexidades de domínios. Como indicadores de qualidade, as métricas IGD<sub>p</sub> e Hypervolume foram aplicadas. Foram realizados testes estatísticos para avaliar a significância das diferenças encontradas entre os algoritmos.

A partir da análise dos resultados, conclui-se que levar em consideração a relação entre o método de seleção e as estratégias para gerar as novas soluções é importante para o desempenho de algoritmos eficientes. Para os problemas de *benchmark* da ferramenta COCO, o método de seleção NSGA-II, muito aplicado pela comunidade de MOEAs, não foi o que produziu os melhores resultados, entretanto o método IBEA, produziu resultados melhores em conjunto com os algoritmos de modelagem probabilística que foram aplicados. Para os problemas DTLZ, as melhores combinações de algoritmos são UMDA<sup>G</sup>-NSGA-II e CMA-ES-IBEA. Isto mostra que esses dois importantes componentes dos MOEAs devem ser considerados em conjunto.

Outra conclusão desta análise, é que os modelos probabilísticos obtiveram uma vantagem sobre os métodos tradicionais de recombinação para a maioria das funções consideradas. Como mencionado anteriormente, considerando a métrica de Hypervolume, os algoritmos com melhor desempenho usam modelos probabilísticos. Para os problemas da ferramenta COCO, o método SBX produz regularmente os piores resultados para todas as dimensões das variáveis de



decisão consideradas. Também para os mesmos problemas, e como outra conclusão importante da análise, pode-se observar que o modelo probabilístico CMA-ES obteve os melhores resultados para todas as dimensões. Embora já tenha sido relatado que o CMA-ES tenha obtido bons resultados para essas funções bi-objetivo, não foram encontrados relatórios anteriores sobre o uso conjunto do algoritmo IBEA e do CMA-ES, ou uma análise detalhada sobre o impacto de diferentes métodos de seleção para o CMA-ES.

Considerando que a abordagem, proposta neste trabalho, para investigar a interação entre seleção e geração de soluções, poderia ser aplicada para investigar como outros componentes de MOEAs influenciam o comportamento dos algoritmos. Embora existam várias estratégias para abordar os subproblemas envolvidos na otimização multi-objetivo (por exemplo, inicialização, arquivamento, etc.), estas estratégias são normalmente avaliadas no contexto da proposta original. Isolando os componentes de alguns dos algoritmos populares e testá-los combinados com outros componentes, permitiu entender melhor o papel de cada método na busca e encontrar variantes mais eficientes de MOEAs.

Por fim, os resultados obtidos inspiram mais pesquisas sobre o uso de modelos probabilísticos para usar a vantagem de suas diferentes capacidades para projetar algoritmos mais adaptáveis, capazes de explorar os diversos cenários que podem ser encontrados em diferentes conjuntos de testes de *benchmark* e em problemas de otimização do mundo real. Trabalhos futuros incluem determinar quando alguma combinação particular de métodos de geração de, seleção e solução, produzem melhorias no comportamento dos algoritmos. Este tipo de explicações pode ajudar a um design mais inteligente de MOEAs. Também considera-se o uso de outros métodos de modelagem probabilística e a análise de MOPs com representação binária e discreta.

## Referências Bibliográficas

- [1] C.W. Ahn, RS Ramakrishna, and D.E. Goldberg. Real-coded Bayesian optimization algorithm: Bringing the strength of BOA into the continuous world. In *Proceedings of the Genetic and Evolutionary Computation Conference GECCO-2004*, pages 840–851. Springer, 2004.
- [2] R. Armañanzas, I. Inza, R. Santana, Y. Saeys, J. L. Flores, J. A. Lozano, Y. Van de Peer, R. Blanco, V. Robles, C. Bielza, and P. Larrañaga. A review of estimation of distribution algorithms in bioinformatics. *BioData Mining*, 1(6):doi:10.1186/1756-0381-1-6, 2008.
- [3] Anne Auger, Dimo Brockhoff, Nikolaus Hansen, Dejan Tušar, Tea Tušar, and Tobias Wagner. Benchmarking RM-MEDA on the Bi-objective BBOB-2016 Test Suite. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion, GECCO '16 Companion*, pages 1241–1247, New York, NY, USA, 2016. ACM.
- [4] Nicola Beume, Boris Naujoks, and Michael Emmerich. SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, 181(3):1653–1669, 2007.
- [5] Hans-Georg Beyer and Hans-Paul Schwefel. Evolution strategies – A comprehensive introduction. *Natural Computing*, 1(1):3–52.
- [6] S. Bleuler, M. Laumanns, L. Thiele, and E. Zitzler. PISA - A Platform and Programming Language Independent Interface for Search Algorithms. In C. M. Fonseca et al., editors, *Conference on Evolutionary Multi-Criterion Optimization (EMO 2003)*, volume 2632 of *LNCS*, pages 494–508, Berlin, 2003. Springer.
- [7] D. Brockhoff, T. Tušar, D. Tušar, T. Wagner, N. Hansen, and A. Auger. Biobjective Performance Assessment with the COCO Platform. *ArXiv e-prints*, may 2016.
- [8] A. E. I. Brownlee, J. McCall, Q. Zhang, and D. Brown. Approaches to selection and their effect on fitness modelling in an Estimation of Distribution Algorithm. In *Proceedings of the 2008 Congress on Evolutionary Computation CEC-2008*, pages 2621–2628, Hong Kong, 2008. IEEE Press.
- [9] Alexander E.I. Brownlee, John A.W. McCall, and Martin Pelikan. Influence of Selection on Structure Learning in Markov Network EDAs: An Empirical Study. In *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation, GECCO '12*, pages 249–256, New York, NY, USA, 2012. ACM.
- [10] Olacir Castro Jr., Aurora Pozo, Jose A. Lozano, and Roberto Santana. Transfer weight functions for injecting problem information in the multi-objective CMA-ES. *Memetic Computing*, pages 1–28, 2016.

- [11] Carlos A. Coello Coello, Gary B. Lamont, and David A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
- [12] Carlos A. Coello Coello, Gary B. Lamont, and David A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.
- [13] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. Wiley-Interscience Series in Systems and Optimization. John Wiley & Sons, Chichester, 2001.
- [14] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *Trans. Evol. Comp.*, 6(2):182–197, apr 2002.
- [15] Kalyanmoy Deb and Ram Bhushan Agrawal. Simulated Binary Crossover for Continuous Search Space. *Complex systems*, 9(2):115–148, 1995.
- [16] Kalyanmoy Deb and Mayank Goyal. A combined genetic adaptive search (geneas) for engineering design. *Computer Science and Informatics*, 26:30–45, 1996.
- [17] Kalyanmoy Deb, Lothar Thiele, Marco Laumanns, and Eckart Zitzler. Scalable multi-objective optimization test problems. In *Proceedings of the 2002 Congress on Evolutionary Computation CEC-2002*, volume 2, pages 825–830. IEEE press, 2002.
- [18] Janez Demsar. Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research*, 7:1–30, 2006.
- [19] Juan J. Durillo and Antonio J. Nebro. jmetal: A java framework for multi-objective optimization. *Advances in Engineering Software*, 42:760–771, 2011.
- [20] C. Echegoyen. *Contributions to the Analysis and Understanding of Estimation of Distribution Algorithms*. PhD thesis, Department of Computer Science and Artificial Intelligence. University of the Basque Country, 2012.
- [21] Carlos M. Fonseca and Peter J. Fleming. Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In *Proceedings of the 5th International Conference on Genetic Algorithms*, pages 416–423, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc.
- [22] Carlos M Fonseca, Joshua D Knowles, Lothar Thiele, and Eckart Zitzler. A tutorial on the performance assessment of stochastic multiobjective optimizers. *Third International Conference on Evolutionary Multi-Criterion Optimization (EMO 2005)*, 216:240, 2005.
- [23] Milton Friedman. The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance. *Journal of the American Statistical Association*, 32(200):675–701, 1937.
- [24] M. Gallagher, I. Wood, J. Keith, and G. Sofronov. Bayesian inference in estimation of distribution algorithms. In *2007 IEEE Congress on Evolutionary Computation*, pages 127–133, Sept 2007.
- [25] D. E. Goldberg. *The Design of innovation: Lessons from and for Competent Genetic Algorithms*. Kluwer Academics Publishers, 2002.

- [26] N. Hansen and A. Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*, pages 312–317, 1996.
- [27] Nikolaus Hansen, Anne Auger, Olaf Mersmann, Tea Tusar, and Dima Brockhoff. COCO: A Platform for Comparing Continuous Optimizers in a Black-Box Setting. *CoRR*, abs/1603.08785, 2016.
- [28] Nikolaus Hansen, Anne Auger, Raymond Ros, Steffen Finck, and Petr Pošík. Comparing Results of 31 Algorithms from the Black-box Optimization Benchmarking BBOB-2009. In *Proceedings of the 12th Annual Conference Companion on Genetic and Evolutionary Computation*, GECCO '10, pages 1689–1696, New York, NY, USA, 2010. ACM.
- [29] Nikolaus Hansen and Andreas Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on*, pages 312–317. IEEE, 1996.
- [30] Georges Harik. Linkage learning via probabilistic modeling in the ECGA. IlliGAL Report 99010, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 1999.
- [31] M. Hauschild and M. Pelikan. An introduction and survey of estimation of distribution algorithms. *Swarm and evolutionary computation*, 1(3):111–128, 2011.
- [32] S. Huband, P. Hingston, L. Barone, and L. While. A Review of Multiobjective Test Problems and a Scalable Test Problem Toolkit. *Transactions on Evolutionary Computation*, 10(5):477–506, oct 2006.
- [33] Christian Igel, Nikolaus Hansen, and Stefan Roth. Covariance Matrix Adaptation for Multi-objective Optimization. *Evol. Comput.*, 15(1):1–28, March 2007.
- [34] H. Karshenas, R. Santana, C. Bielza, and P. Larrañaga. Multiobjective Estimation of Distribution Algorithm Based on Joint Modeling of Objectives and Variables. *Evolutionary Computation, IEEE Transactions on*, 18(4):519–542, 2014.
- [35] J. R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. The MIT Press, Cambridge, MA, 1992.
- [36] William H. Kruskal and W. Allen Wallis. Use of ranks in one-criterion variance analysis. *Journal of the American Statistical Association*, 47(260):pp. 583–621, 1952.
- [37] Pedro Larrañaga and Jose A. Lozano. *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, Norwell, MA, USA, 2001.
- [38] P. Larrañaga, R. Etxeberria, J. A. Lozano, and J. M. Peña. Optimization by learning and simulation of Bayesian and Gaussian networks. Technical Report EHU-KZAA-IK-4/99, Department of Computer Science and Artificial Intelligence, University of the Basque Country, 1999.
- [39] P. Larrañaga, H. Karshenas, C. Bielza, and R. Santana. A review on probabilistic graphical models in evolutionary computation. *Journal of Heuristics*, 18(5):795–819, 2012.

- [40] P. Larrañaga and J. A. Lozano, editors. *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, Boston/Dordrecht/London, 2002.
- [41] Pedro Larrañaga, Ramón Etxeberria, Jose Antonio Lozano, and Jose Manuel Peña. Optimization in Continuous Domains by Learning and Simulation of Gaussian Networks. In *Workshop in Optimization by Building and using Probabilistic Models*, A Workshop withing the 2000 Genetic and Evolutionary Computation Conference, GECCO 2000, pages 201–204, Las Vegas, Nevada, USA, 2000.
- [42] Ilya Loshchilov and Tobias Glasmachers. Anytime Bi-Objective Optimization with a Hybrid Multi-Objective CMA-ES (HMO-CMA-ES). In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion*, GECCO '16 Companion, pages 1169–1176, New York, NY, USA, 2016. ACM.
- [43] J. A. Lozano, P. Larrañaga, I. Inza, and E. Bengoetxea, editors. *Towards a New Evolutionary Computation: Advances on Estimation of Distribution Algorithms*. Springer, 2006.
- [44] Christian Lübben, Benjamín Barán, and Carlos Brizuela. A survey on multi-objective evolutionary algorithms for many-objective problems. *Comput. Optim. Appl.*, 58(3):707–756, July 2014.
- [45] Sean Luke. *Essentials of Metaheuristics*. Lulu, second edition, 2013. Available for free at <http://cs.gmu.edu/~sean/book/metaheuristics/>.
- [46] Anirban Mukhopadhyay, Ujjwal Maulik, and Sanghamitra Bandyopadhyay. A survey of multiobjective evolutionary clustering. *ACM Comput. Surv.*, 47(4):61:1–61:46, May 2015.
- [47] Nadia Nedjah and Luiza de Macedo Mourelle. Evolutionary multi-objective optimisation: A survey. *Int. J. Bio-Inspired Comput.*, 7(1):1–25, March 2015.
- [48] P. Nemenyi. *Distribution-free Multiple Comparisons*. PhD thesis, Princeton University, 1963.
- [49] M. Pelikan and H. Mühlenbein. The bivariate marginal distribution algorithm. In R. Roy, T. Furuhashi, and P.K. Chawdhry, editors, *Advances in Soft Computing - Engineering Design and Manufacturing*, pages 521–535, London, 1999. Springer.
- [50] Martin Pelikan. *Bayesian optimization algorithm: From single level to hierarchy*. PhD thesis, University of Illinois, 2002.
- [51] Martin Pelikan, Mark Hauschild, and Fernando G. Lobo. Introduction to estimation of distribution algorithms. MEDAL Report No. 2012003, Missouri Estimation of Distribution Algorithms Laboratory (MEDAL), 2012.
- [52] Martin Pelikan, Kumara Sastry, and David E. Goldberg. *Scalable Optimization via Probabilistic Modeling*, chapter Multiobjective Estimation of Distribution Algorithms, pages 223–248. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.
- [53] Roberto Santana, Alexander Mendiburu, and Jose A Lozano. Customized Selection in Estimation of Distribution Algorithms. In *Proceedings of the 10th International Conference Simulated Evolution and Learning (SEAL-2014)*, pages 94–105. Springer, 2014.

- [54] H. Sato, H. E. Aguirre, and K. Tanaka. Local dominance using polar coordinates to enhance multiobjective evolutionary algorithms. In *Evolutionary Computation, 2004. CEC2004. Congress on*, volume 1, pages 188–195 Vol.1, June 2004.
- [55] O. Schutze, X. Esquivel, A. Lara, and Carlos A. Coello Coello. Using the Averaged Hausdorff Distance as a Performance Measure in Evolutionary Multiobjective Optimization. *Evolutionary Computation, IEEE Transactions on*, 16(4):504–522, August 2012.
- [56] Michèle Sebag and Antoine Ducoulombier. Extending population-based incremental learning to continuous search spaces. In *Proceedings of the 5th International Conference on Parallel Problem Solving from Nature, PPSN V*, pages 418–427, London, UK, UK, 1998. Springer-Verlag.
- [57] Bernard W Silverman. *Density estimation for statistics and data analysis*, volume 26. CRC press, 1986.
- [58] M. Soto, Y. González-Fernández, and A. Ochoa. Modeling with Copulas and Vines in Estimation of Distribution Algorithms. *ArXiv e-prints*, oct 2012.
- [59] Gilbert Syswerda. Uniform Crossover in Genetic Algorithms. In *Proceedings of the 3rd International Conference on Genetic Algorithms*, pages 2–9, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc.
- [60] Mohammad Tabatabaei, Jussi Hakanen, Markus Hartikainen, Kaisa Miettinen, and Karthik Sindhya. A survey on handling computationally expensive multiobjective optimization problems using surrogates: Non-nature inspired methods. *Struct. Multidiscip. Optim.*, 52(1):1–25, July 2015.
- [61] T. Tusar, D. Brockhoff, N. Hansen, and A. Auger. Coco: The bi-objective black box optimization benchmarking (bbob-biobj) test suite. *ArXiv e-prints*, apr 2016.
- [62] David A Van Veldhuizen and Gary B Lamont. On Measuring Multiobjective Evolutionary Algorithm Performance. In *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on*, volume 1, pages 204–211. IEEE, 2000.
- [63] L. While, L. Bradstreet, and L. Barone. A fast way of calculating exact hypervolumes. *IEEE Transactions on Evolutionary Computation*, 16(1):86–95, February 2012.
- [64] L. While, P. Hingston, L. Barone, and S. Huband. A faster algorithm for calculating hypervolume. *IEEE Transactions on Evolutionary Computation*, 10(1):29–38, Feb 2006.
- [65] Saúl Zapotecas-Martínez, Bilel Derbel, Arnaud Liefooghe, Dimo Brockhoff, Hernán Aguirre, and Kiyoshi Tanaka. Injecting CMA-ES into MOEA/D. In *Genetic and Evolutionary Computation Conference (GECCO 2015)*, 2015.
- [66] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization. In K. C. Giannakoglou and Others, editors, *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001)*, pages 95–100. International Center for Numerical Methods in Engineering (CIMNE), 2002.

- [67] E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, Nov 1999.
- [68] E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca, and V.G. da Fonseca. Performance Assessment of Multiobjective Optimizers: An Analysis and Review. *Evolutionary Computation, IEEE Transactions on*, 7(2):117–132, April 2003.
- [69] Eckart Zitzler. Evolutionary algorithms for multiobjective optimization: Methods and applications, 1999.
- [70] Eckart Zitzler, Dima Brockhoff, and Lothar Thiele. The Hypervolume Indicator Revisited: On the Design of Pareto-compliant Indicators Via Weighted Integration. In Shigeru Obayashi, Kalyanmoy Deb, Carlo Poloni, Tomoyuki Hiroyasu, and Tadahiko Murata, editors, *Evolutionary Multi-Criterion Optimization*, volume 4403 of *Lecture Notes in Computer Science*, pages 862–876. Springer Berlin Heidelberg, 2007.
- [71] Eckart Zitzler and Simon Künzli. Indicator-Based Selection in Multiobjective Search. In *Proc. 8th International Conference on Parallel Problem Solving from Nature (PPSN VIII)*, pages 832–842. Springer, 2004.
- [72] Eckart Zitzler and Simon Künzli. *Parallel Problem Solving from Nature - PPSN VIII: 8th International Conference, Birmingham, UK, September 18-22, 2004. Proceedings*, chapter Indicator-Based Selection in Multiobjective Search, pages 832–842. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.