Um estudo dos parâmetros de controle de um Algoritmo Genético

Ricardo de Brito Damm

DISSERTAÇÃO APRESENTADA
AO
INSTITUTO DE MATEMÁTICA E ESTATÍSTICA
DA
UNIVERSIDADE DE SÃO PAULO
PARA
OBTENÇÃO DO GRAU DE MESTRE
EM
MATEMÁTICA APLICADA

Área de concentração: Análise Numérica
Orientador: Prof. Dr. Luis Carlos de Castro Santos

São Paulo. 06 de outubro de 2005.

Este exemplar corresponde à redação final
da dissertação devidamente corrigida e
defendida por Ricardo de Brito Damm
e aprovada pela comissão julgadora.

São Paulo, 06 de outubro de 2005.

Banca examinadora:

Prof. Dr. Luis Carlos de Castro Santos (Orientador) — MAP-IME-USP

Prof. Dr. Ernesto Julián Goldberg Birgin — DCC-IME-USP

Profa. Dra. Celma de Oliveira Ribeiro — PRO-POLI-USP

Aos meus pais, Heinz e Lili.

Agradecimentos

Meus sinceros agradecimentos ao meu orientador, *Luis Carlos*, pela dedicação de todos esses anos.

À professora Celma, que me acompanhou durante os anos de graduação, os meus agradecimentos por todos os conselhos e incentivos.

Aos meus amigos, pelo incentivo e carinho.

Resumo

Os algoritmos genéticos são algoritmos de busca que utilizam o mecanismo de seleção natural, segundo a teoria da origem das espécies de *Charles Darwin*, e os conceitos da genética moderna. São algoritmos computacionalmente simples; não possuem limitantes como a necessidade de derivadas, continuidade ou a dificuldade em trabalhar com funções multimodais; trabalham com um conjunto de soluções, são probabilísticos e robustos. O objetivo deste trabalho é fazer um estudo amplo sobre o Algoritmo Genético, aplicando-o a diferentes tipos de problemas. Serão estudadas funções com ou sem restrições, com um ou mais objetivos (multi-objetivos). As versões mais conhecidas do Algoritmo Genético para cada tipo de problema serão apresentadas. Também se estudou a influência de alguns parâmetros na busca: a probabilidade de recombinação (*crossover*), de mutação, o número de gerações e o tamanho da população. Procurou-se determinar as faixas de variações que produzem melhores resultados e, conseqüentemente, facilitam o ajuste do algoritmo.

Abstract

Genetic algorithms are search algorithms based on the mechanics of natural selection - on the evolution theory of Charles Darwin - and modern genetic. These algorithms are computationally simple, not limited by restrictive assumptions about the search space (existence of derivatives, continuity and unimodality), search from a population of points (not a single point), use probabilistic transition rules (not deterministic rules) and are robustness. The present work is an overview of Genetic Algorithms and describes its application to unconstrained numerical optimization, constrained optimization problems and multiobjective problems. The most popular Genetic Algorithms are presented. The influence of the control parameters (probabilities of crossover, mutation, population size and the number of generation) was studied.

Índice

Lista de Figuras	xix
Lista de Tabelas	xxiii
1 Introdução	1
1.1 Algoritmos genéticos	1
1.2 Revisão bibliográfica	2
1.3 Objetivo do trabalho	3
1.4 Descrição dos capítulos	4
2 O Algoritmo Genético	5
2.1 Notação utilizada	5
2.2 Alguns métodos de otimização	6
2.3 De Lamarck ao neodarwinismo	7
2.4 O Algoritmo Genético: parâmetros e operadores	8
2.4.1 Tamanho da população	8
2.4.2 Codificação das variáveis	9
2.4.3 População inicial	10
2.4.4 Recombinação ou <i>Crossover</i>	11
2.4.5 Mutação	12
2.4.6 Seleção	14

2.4.7	Elitismo	14
2.4.8	Cálculo da aptidão	14
2.4.9	Critério de parada	15
2.5	Aplicação do Algoritmo Genético	16
2.6	Teorema fundamental do Algoritmo Genético	21
2.6.1	Alguns conceitos	21
2.6.2	Teorema fundamental do Algoritmo Genético	22
2.6.3	Exemplos	24
2.7	Convergência do algoritmo	26
2.8	Outras técnicas	27
2.9	Vantagens e desvantagens do Algoritmo Genético	29
3	Problemas de minimização em caixas	31
3.1	Conjunto de problemas teste	31
3.2	Função penalidade	34
3.3	Método de estudo dos parâmetros	35
3.4	Probabilidade de recombinação	40
3.5	Probabilidade de mutação	43
3.6	Tamanho da população	46
3.7	Conclusão	48
4	Problemas com restrições	49
4.1	Métodos que preservam a factibilidade das soluções	50
4.1.1	Genocop	50
4.1.2	<i>Searching the boundary of feasible region</i> (Busca da fronteira da região factível)	50
4.2	Métodos baseados em penalizações	51
4.2.1	Método de penalidades estáticas	52
4.2.2	Método de penalidades dinâmicas	52
4.2.3	Genocop II	53
4.2.4	Método de penalidades adaptativas	54

4.2.5	Pena de morte	55
4.3	Outras maneiras de abordar as restrições	55
4.3.1	Memória de comportamento	55
4.3.2	Método da superioridade dos pontos factíveis	56
4.3.3	Método de penalidade livre de parâmetros	57
4.3.4	Genocop III	58
4.4	Métodos híbridos	59
4.4.1	Multiplicadores de Lagrange	59
4.4.2	Otimização aleatória	60
4.4.3	Sistema imune	61
4.5	Escolha do método e problemas teste	62
4.6	Probabilidade de recombinação	64
4.7	Probabilidade de mutação	67
4.8	Tamanho da população	70
4.9	Conclusão	72
5	Problemas Multi-objetivos	73
5.1	Métodos que agregam os objetivos em uma única função	74
5.1.1	Agregação de objetivos por pesos variáveis	74
5.1.2	Goal programming	75
5.1.3	Goal attainment	75
5.1.4	The ε -Constraint Method	76
5.2	Outros métodos não baseados no conceito de Pareto	77
5.2.1	Vector Evaluated Genetic Algorithms (<i>VEGA</i>)	77
5.2.2	Ordem Lexicográfica	77
5.3	Métodos que utilizam o conjunto ótimo de Pareto	78
5.3.1	Multi-objective Optimization Genetic Algorithm (<i>MOGA</i>)	80
5.3.2	Niched Pareto Genetic Algorithm (<i>NPGA</i>)	81
5.3.3	Nondominated Sorting Genetic Algorithm (<i>NSGA</i>)	81
5.3.4	Strength Pareto Evolutionary Algorithm (<i>SPEA</i>)	82

5.4	Nondominated Sorting Genetic Algorithm II	82
5.5	Aplicação do NSGA II	86
5.6	Probabilidade de recombinação	89
5.7	Probabilidade de mutação	92
5.8	Tamanho da população	95
5.9	Conclusão	98
6	Conclusões	99
6.1	Resultados globais	99
6.1.1	Probabilidade de recombinação	99
6.1.2	Probabilidade de mutação	103
6.1.3	Tamanho da população	106
6.1.4	Resultados globais	109
6.2	Análise de sensibilidade	110
6.3	Novas medidas	113
6.4	Conclusão final	115
A	Problemas de minimização em caixas	117
A.1	Conjunto de problemas teste	117
A.2	Resultados	123
B	Problemas com restrições	129
B.1	Conjunto de problemas teste	129
B.2	Resultados	136
C	Problemas Multi-objetivos	143
C.1	Conjunto de problemas teste	143
C.2	Resultados	148
D	Resultados Globais	155
E	Novos resultados	161

E.1	Nova função de minimização em caixas	161
E.2	Nova função com restrições	166
E.3	Nova função multi-objetivos	171

Lista de Figuras

2.1	Gráfico da função de Branin	16
2.2	Gráfico da 1ª geração da função de Branin	17
2.3	Gráfico da 5ª geração da função de Branin	17
2.4	Gráfico da 10ª geração da função de Branin	18
2.5	Gráfico da função de Easom	19
2.6	Gráficos da função de Easom (gerações)	20
2.7	Gráfico do número de <i>schemata</i> H_1 para cada geração da função Easom . .	25
2.8	Gráfico do número de <i>schemata</i> H_2 para cada geração da função Easom . .	25
2.9	Gráfico da função de Ackley's Path	30
3.1	Penalidade normalizada da função Michalewicz — no cenário A	37
3.2	Penalidade normalizada média da função Michalewicz	39
3.3	Penalidade normalizada média <i>versus</i> probabilidade de recombinação . . .	40
3.4	Penalidade total normalizada <i>versus</i> probabilidade de recombinação	40
3.5	Número de problemas com baixas penalidades <i>versus</i> probabilidade de re- combinação	41
3.6	Número médio de gerações da melhor solução <i>versus</i> probabilidade de re- combinação	42
3.7	Penalidade normalizada média <i>versus</i> probabilidade de mutação	43
3.8	Penalidade total normalizada <i>versus</i> probabilidade de mutação	43
3.9	Número de problemas com baixas penalidades <i>versus</i> probabilidade de mutação	44

3.10	Número médio de gerações da melhor solução <i>versus</i> probabilidade de mutação	45
3.11	Penalidade normalizada média <i>versus</i> tamanho da população	46
3.12	Penalidade total normalizada <i>versus</i> tamanho da população	46
3.13	Número de problemas com baixas penalidades <i>versus</i> tamanho da população	47
3.14	Número médio de gerações da melhor solução <i>versus</i> tamanho da população	47
4.1	Penalidade normalizada média <i>versus</i> probabilidade de recombinação . . .	64
4.2	Penalidade total normalizada <i>versus</i> probabilidade de recombinação . . .	64
4.3	Número de problemas com baixas penalidades <i>versus</i> probabilidade de recombinação	65
4.4	Número médio de gerações da melhor solução <i>versus</i> probabilidade de recombinação	66
4.5	Penalidade normalizada média <i>versus</i> probabilidade de mutação	67
4.6	Penalidade total normalizada <i>versus</i> probabilidade de mutação	67
4.7	Número de problemas com baixas penalidades <i>versus</i> probabilidade de mutação	68
4.8	Número médio de gerações da melhor solução <i>versus</i> probabilidade de mutação	68
4.9	Penalidade normalizada média <i>versus</i> tamanho da população	70
4.10	Penalidade total normalizada <i>versus</i> tamanho da população	70
4.11	Número de problemas com baixas penalidades <i>versus</i> tamanho da população	71
4.12	Número médio de gerações da melhor solução <i>versus</i> tamanho da população	71
5.1	Exemplo de um problema multi-objetivo	87
5.2	Penalidade normalizada média <i>versus</i> probabilidade de recombinação . . .	89
5.3	Penalidade total normalizada <i>versus</i> probabilidade de recombinação . . .	89
5.4	Número de problemas com baixas penalidades <i>versus</i> probabilidade de recombinação	90
5.5	Número médio de gerações da melhor solução <i>versus</i> probabilidade de recombinação	90
5.6	Penalidade normalizada média <i>versus</i> probabilidade de mutação	92
5.7	Penalidade total normalizada <i>versus</i> probabilidade de mutação	92

5.8	Número de problemas com baixas penalidades <i>versus</i> probabilidade de mutação	93
5.9	Número médio de gerações da melhor solução <i>versus</i> probabilidade de mutação	93
5.10	Penalidade normalizada média <i>versus</i> tamanho da população	95
5.11	Penalidade total normalizada <i>versus</i> tamanho da população	95
5.12	Penalidade normalizada média (exceto o problema 4) <i>versus</i> tamanho da população	96
5.13	Penalidade total normalizada (exceto o problema 4) <i>versus</i> tamanho da população	96
5.14	Número de problemas com baixas penalidades <i>versus</i> tamanho da população	97
5.15	Número médio de gerações da melhor solução <i>versus</i> tamanho da população	97
6.1	Penalidade normalizada média <i>versus</i> probabilidade de recombinação . . .	99
6.2	Penalidade total normalizada <i>versus</i> probabilidade de recombinação . . .	100
6.3	Porcentagem de problemas com baixas penalidades <i>versus</i> probabilidade de recombinação	101
6.4	Número médio de gerações da melhor solução <i>versus</i> probabilidade de recombinação	101
6.5	Penalidade normalizada média <i>versus</i> probabilidade de mutação	103
6.6	Penalidade total normalizada <i>versus</i> probabilidade de mutação	103
6.7	Porcentagem de problemas com baixas penalidades <i>versus</i> probabilidade de mutação	104
6.8	Número médio de gerações da melhor solução <i>versus</i> probabilidade de mutação	104
6.9	Penalidade normalizada média <i>versus</i> tamanho da população	106
6.10	Penalidade total normalizada <i>versus</i> tamanho da população	106
6.11	Porcentagem de problemas com baixas penalidades <i>versus</i> tamanho da população	107
6.12	Número médio de gerações da melhor solução <i>versus</i> tamanho da população	107
6.13	Coeficiente α_1 <i>versus</i> tamanho da população	110
6.14	Coeficiente α_2 <i>versus</i> probabilidade de recombinação	111
6.15	Coeficiente α_2 <i>versus</i> probabilidade de mutação	111

6.16	Coeficiente α_2 <i>versus</i> tamanho da população	112
6.17	Novas medidas para a probabilidade de recombinação	113
6.18	Novas medidas para a probabilidade de mutação	114
6.19	Novas medidas para o tamanho da população	115

Lista de Tabelas

2.1	Transformação do código binário para a forma real	10
2.2	Comparação entre os números binários e o código <i>Gray</i>	10
3.1	Cenários A, B e C	32
3.2	Função de <i>Michalewicz</i> no cenário A — resultados	33
3.3	Função de <i>Michalewicz</i> no cenário A — penalidades	36
3.4	Função de <i>Michalewicz</i> nos três cenários	38
3.5	Melhores resultados para probabilidade de recombinação	42
3.6	Melhores resultados para probabilidade de mutação	45
3.7	Melhores resultados para o tamanho da população	48
3.8	Principais resultados do capítulo	48
4.1	Conjunto de problemas teste com restrições	63
4.2	Melhores resultados para probabilidade de recombinação	66
4.3	Melhores resultados para probabilidade de mutação	69
4.4	Melhores resultados para o tamanho da população	72
4.5	Principais resultados do capítulo	72
5.1	Conjunto de problemas teste Multi-objetivos	86
5.2	Melhores resultados para probabilidade de recombinação	91
5.3	Melhores resultados para probabilidade de mutação	94
5.4	Melhores resultados para o tamanho da população	98

5.5	Principais resultados do capítulo	98
6.1	Melhores resultados para probabilidade de recombinação	102
6.2	Melhores resultados para probabilidade de mutação	105
6.3	Melhores resultados para o tamanho da população	108
6.4	Resultados globais	109
6.5	Principais resultados do trabalho	113
A.1	Penalidades <i>versus</i> probabilidade de recombinação	123
A.2	Número de problemas e de gerações <i>versus</i> probabilidade de recombinação	124
A.3	Penalidades <i>versus</i> probabilidade de mutação	125
A.4	Número de problemas e de gerações <i>versus</i> probabilidade de mutação . . .	126
A.5	Penalidades <i>versus</i> tamanho da população	127
A.6	Número de problemas e de gerações <i>versus</i> tamanho da população	128
B.1	Penalidades <i>versus</i> probabilidade de recombinação	136
B.2	Número de problemas e de gerações <i>versus</i> probabilidade de recombinação	137
B.3	Penalidades <i>versus</i> probabilidade de mutação	138
B.4	Número de problemas e de gerações <i>versus</i> probabilidade de mutação . . .	139
B.5	Penalidades <i>versus</i> tamanho da população	140
B.6	Número de problemas e de gerações <i>versus</i> tamanho da população	141
C.1	Penalidades <i>versus</i> probabilidade de recombinação	148
C.2	Número de problemas e de gerações <i>versus</i> probabilidade de recombinação	149
C.3	Penalidades <i>versus</i> probabilidade de mutação	150
C.4	Número de problemas e de gerações <i>versus</i> probabilidade de mutação . . .	151
C.5	Penalidades <i>versus</i> tamanho da população	152
C.6	Penalidades (todos os problemas, exceto o 4) <i>versus</i> tamanho da população	153
C.7	Número de problemas e de gerações <i>versus</i> tamanho da população	154
D.1	Penalidades <i>versus</i> probabilidade de recombinação	155
D.2	Número de problemas e de gerações <i>versus</i> probabilidade de recombinação	156

D.3	Penalidades <i>versus</i> probabilidade de mutação	157
D.4	Número de problemas e de gerações <i>versus</i> probabilidade de mutação . . .	158
D.5	Penalidades <i>versus</i> tamanho da população	159
D.6	Número de problemas e de gerações <i>versus</i> tamanho da população	160
E.1	Penalidade <i>versus</i> probabilidade de recombinação - novos resultados em um problema de minimização em caixas	162
E.2	Penalidade <i>versus</i> probabilidade de mutação - novos resultados em um problema de minimização em caixas	164
E.3	Penalidade <i>versus</i> tamanho da população - novos resultados em um problema de minimização em caixas	165
E.4	Penalidade <i>versus</i> probabilidade de recombinação - novos resultados em uma função com restrições	167
E.5	Penalidade <i>versus</i> probabilidade de mutação - novos resultados em uma função com restrições	169
E.6	Penalidade <i>versus</i> tamanho da população - novos resultados em uma função com restrições	170
E.7	Penalidade <i>versus</i> probabilidade de recombinação - novos resultados em uma função multi-objetivos	172
E.8	Penalidade <i>versus</i> probabilidade de mutação - novos resultados em uma função multi-objetivos	173
E.9	Penalidade <i>versus</i> tamanho da população - novos resultados em uma função multi-objetivos	174

Introdução

1.1 Algoritmos genéticos

Os algoritmos genéticos são algoritmos de busca que utilizam, principalmente, o mecanismo de seleção natural — segundo a teoria da origem das espécies, de Charles Darwin — e os conceitos da genética moderna. A partir de uma população artificial inicial, novas gerações são formadas e evoluídas, através de uma competição no momento da reprodução. Cada indivíduo da população é uma solução potencial para um problema de otimização qualquer e, quanto melhor é a sua qualidade, quanto maior é a sua aptidão ao meio ambiente, maior a probabilidade de ser escolhido para gerar descendentes. Após o sorteio dos futuros pais, geram-se os filhos mediante os operadores de recombinação cromossômica (mistura do material genético de dois progenitores) e de mutação (alteração aleatória de um gene). Depois de um número fixo de gerações, ou de algum outro critério de parada, a população final deverá ter convergido para os pontos de máximo local e global.

As principais características dos Algoritmos Genéticos são:

- robustez: pode ser aplicado a uma diversidade de tipos de problemas;
- não necessita da continuidade de funções: utiliza diretamente a função objetivo do problema, não sendo necessário o cálculo de derivadas;
- trabalha com um conjunto de soluções e não apenas uma única;
- é computacionalmente simples;
- trabalha com qualquer tipo de variável (discretas, contínuas ou binárias);
- é um método de otimização global;
- pode ser combinado com outros métodos ou algoritmos locais.

Talvez a principal vantagem dos Algoritmos Genéticos seja a de trabalhar com um conjunto de soluções. Algumas vezes a melhor solução gerada por um algoritmo pode não ser factível na realidade, devido às inevitáveis simplificações feitas durante a modelagem. Conseqüentemente, quando há diversas boas soluções para um problema, a probabilidade de existir alguma realmente factível é significativamente maior. Como o Algoritmo Genético pode facilmente se adaptar a diferentes tipos de problemas, inúmeras pesquisas científicas foram realizadas, nos diversos campos da engenharia, da física, da química e no mercado financeiro.

Porém, como trabalha com um conjunto de populações, o algoritmo tende a ser mais caro do que outros métodos. Outra desvantagem é a grande quantidade de parâmetros que influenciam na qualidade das soluções.

1.2 Revisão bibliográfica

Os trabalhos precursores do Algoritmo Genético surgiram a partir do final dos anos 50, quando alguns biólogos utilizaram computadores digitais para, através de simulações de modelos genéticos, compreender os fenômenos evolutivos naturais (Baricelli, 1957, 1962; Fraser, 1960, 1962; Martin e Cockerham, 1960). De todos esses estudos, Fraser foi o que mais se aproximou do conceito atual do algoritmo [13].

Nos anos 60 surgiram algumas técnicas evolutivas. Rechenberg (1965) desenvolveu a Estratégia Evolutiva, onde um ou mais indivíduos geram um ou muitos descendentes por mutação. Outra técnica, que se assemelha ainda mais com o Algoritmo Genético, é a Programação Evolutiva; criada com Fogel, Owens e Walsh (1966), trabalha com uma população onde cada indivíduo gera por mutação um único descendente.

O responsável, propriamente dito, pelo desenvolvimento do Algoritmo Genético foi John Holland (Universidade de Michigan) e seu grupo de pesquisa. Em 1962, buscando projetar os sistemas artificiais com os mecanismos evolutivos, Holland cria o conceito de operadores genéticos nesses sistemas. Em 1968, desenvolve o teorema fundamental do Algoritmo Genético (*theory of schemata*). Alguns anos depois, publica uma monografia — *Adaptation in Natural and Artificial Systems* (1975) — abordando a eficácia do algoritmo em espaços de busca complexos.

Também em 1975, De Jong, pesquisador da Universidade de Michigan, completou sua dissertação: *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. De Jong compara o Teorema Fundamental do Algoritmo Genético a vários experimentos, resultando em um importante trabalho sobre o funcionamento dos operadores do algoritmo.

Muitos trabalhos surgiram após as publicações de Holland e De Jong. Brindle (1981) pesquisou diversos métodos de seleção. Baker (1985) propôs o sistema de ranqueamento

para normalizar a aptidão dos indivíduos de uma população; Forrest (1985) apresentou outros métodos, como a escala linear, baseados em estudos da década de 60. Grefenstette e Fitzpatrick (1985) estudaram o comportamento do Algoritmo Genético em funções com muitos mínimos locais (*noisy function*). Schaffer (1984) foi um dos primeiros a estudar problemas multi-objetivos usando o Algoritmo Genético [13].

Outra grande contribuição foi a do pesquisador da Universidade do Alabama, David E. Goldberg. Em seu livro *Genetic algorithms in search, optimization and machine learning* (1989) apresenta uma visão detalhada dos fundamentos do Algoritmo Genético e da sua implementação. Goldberg também aborda algumas técnicas específicas, frutos de seus trabalhos e outros estudos, como Algoritmos Genéticos com processos paralelos de busca, formação de nichos e medidas de similaridade, inserção do conceito de gene dominante e recessivo, cromossomos diplóides, inversão de um cromossomo, taxa de natalidade e idade de um cromossomo, diferenciação sexual, etc. [13].

Na década de 90 surgiram vários métodos para adaptar o Algoritmo Genético à problemas com restrições. Michalewicz e Janikow (1991) apresentaram um método que preserva a factibilidade das soluções. Bean e Hadja-Alouane (1992), Homaifar, Lai, e Qi (1994), Joines e Houck (1994) e Michalewicz (1996) desenvolveram métodos de penalizações. Schoenauer e Xanthakis (1993), Powel e Skolnick (1993), Deb K. (2000) e Michalewicz (1996) sugeriram métodos que distinguem as soluções factíveis das infactíveis durante a evolução do algoritmo [19] [2] [21].

Com relação aos métodos multi-objetivos, vários pesquisadores sugeriram diferentes versões do Algoritmo Genético, principalmente entre os anos de 1992 e 1994. Hajela e Lin (1992) agregaram os objetivos em uma única função. Fonseca e Fleming (1993), Horn e Nafpliotis (1993) e Srinivas e Deb (1994) propuseram métodos baseados nas idéias expostas em Goldberg (1989). Já Zitzler e Theile (1999) desenvolveram um método que trabalha com uma população externa [3] [29].

Nos últimos anos, os principais artigos abordaram novas versões dos métodos desenvolvidos na década 90 [4] [6] [11] ou aplicações do algoritmo para diversos tipos de problemas: projetos de aviões [22] [23] [25], projetos de carros de fórmula 1 [27], problemas de seqüenciamento de produção [16] [24], estudo de esforços em estruturas [1] e problemas de logística [26] [10].

1.3 Objetivo do trabalho

O objetivo deste trabalho é estudar a influência de alguns dos parâmetros do Algoritmo Genético em problemas de otimização: a probabilidade de recombinação, a probabilidade de mutação, o número de gerações e o tamanho da população. Procurou-se determinar os intervalos em que esses parâmetros produzem melhores resultados e, conseqüentemente,

facilitam o desempenho do algoritmo.

Três classes de problemas foram estudadas: problemas com restrições de caixas (apenas restrições com os limites superiores e inferiores das variáveis), funções com restrições (igualdades e desigualdades) e também problemas multi-objetivos (com dois ou mais objetivos).

Para que fosse possível uma comparação entre os resultados dos diversos problemas, escolheu-se uma medida de comparação simples, que pondera o esforço do algoritmo (o número de pontos avaliados ao longo de todas as gerações), a velocidade com que a melhor solução é encontrada, a diferença de valor da função objetivo e a distância entre a melhor solução encontrada e a melhor conhecida. Essa proposta de medida de comparação está descrita detalhadamente na seção 3.2.

1.4 Descrição dos capítulos

O texto se divide da seguinte forma:

O capítulo II contém a descrição de um Algoritmo Genético básico, abordando os seus parâmetros e operadores, o teorema fundamental do Algoritmo Genético e um estudo da sua convergência. Também são abordadas algumas técnicas opcionais, sugeridas por Goldberg [13], e as principais vantagens do algoritmo.

Os problemas de minimização em caixas são estudados no capítulo III, que apresenta também a medida de comparação dos resultados obtidos e de análise dos parâmetros do algoritmo. O final do capítulo mostra as melhores faixas de probabilidade de recombinação cromossômica, de mutação, número de gerações e tamanho da população para esta classe de problemas.

O capítulo IV aborda as funções com restrições. Na primeira parte, é apresentada uma classificação dos Algoritmos Genéticos para problemas com restrições e uma descrição dos principais modelos. A segunda, apresenta o conjunto de problemas teste, o Algoritmo Genético escolhido e os resultados dos parâmetros estudados.

As funções multi-objetivos são abordadas no capítulo V. Inicialmente são apresentados os conceitos básicos para a otimização multi-objetivos e os principais Algoritmos Genéticos. Segue-se a descrição do método utilizado, o conjunto de problemas teste e os resultados para esse tipo de função.

O último capítulo apresenta uma comparação dos resultados dos três anteriores e uma análise de sensibilidade dos valores obtidos. Também foram feitos novos testes, a partir dos resultados obtidos, para confirmá-los. No final, são abordadas as conclusões mais significativas e algumas perspectivas de trabalhos futuros.

O Algoritmo Genético

2.1 Notação utilizada

Um problema de otimização pode ser representado por [17]:

$$\begin{aligned} & \text{Min} f(x) \\ & \text{sujeito a} \\ & x \in \Omega \subset \mathbf{R}^n \end{aligned} \tag{2.1}$$

onde $f(x)$ é a função objetivo e Ω , o espaço factível, que é determinado por todas as restrições do problema.

Uma vez que maximizar uma função equivale a minimizar o seu inverso, ao longo deste trabalho falaremos somente em problemas de minimização.

Se uma solução x^* de (2.1) é tal que $f(x^*) \leq f(x)$ para todo $x \in \Omega$, então dizemos que x^* é o *minimizador global* do problema e $f(x^*)$, o *mínimo* de f em Ω .

Por outro lado, dizemos que uma solução x^* é um *minimizador local* de (2.1) se existe $\varepsilon > 0$ tal que $f(x^*) \leq f(x)$ para todo $x \in \Omega$ tal que $\|x - x^*\| \leq \varepsilon$. Os pontos de mínimos locais podem ser boas soluções, mas não são necessariamente as melhores. Portanto, os métodos de buscas mais eficientes são aqueles que conseguem fugir dos pontos de mínimos locais e alcançar os globais.

Com relação ao espaço factível Ω , distinguiram-se dois tipos diferentes de problemas:

$\Omega = \{x \in \mathbf{R}^n | l \leq x \leq u\}$: otimização em caixas

$\Omega = \{x \in \mathbf{R}^n | h(x) = 0, h : \mathbf{R}^n \rightarrow \mathbf{R}^m; g(x) \leq 0, g : \mathbf{R}^n \rightarrow \mathbf{R}^p; l \leq x \leq u\}$:

problema geral de otimização com restrições

onde a notação $l \leq x \leq u$ significará o conjunto dos $x \in \mathbf{R}^n$, tais que $l_i \leq x_i \leq u_i$ para todo $i = 1, \dots, n$.

2.2 Alguns métodos de otimização

A função objetivo do problema (2.1) pode ser uni ou multimodal, se possuir um ou mais mínimos, respectivamente.

Os métodos de busca tradicionais (Newton, Gradiente ou Lagrange, e suas variações) utilizam derivadas para caminhar em direção dos pontos de mínimos locais; por isso são conhecidos como algoritmos locais. Quando trabalhamos com funções unimodais, esses métodos possuem grande eficácia, pois, há somente um mínimo local que coincide com o global. No entanto, quando se tratam de funções multimodais, esses métodos caminham sempre para o mínimo local mais próximo e, com uma incerteza maior ou menor, conseguem atingir o mínimo global somente após inúmeras tentativas.

Outros métodos (a busca tabu, o *simulated annealing* e os processos evolutivos) não utilizam derivadas e, ao percorrerem a região factível, não finalizam necessariamente a busca nos primeiros mínimos locais encontrados. Em princípio, esses métodos possuem uma maior probabilidade de encontrarem o mínimo global; por isso, são conhecidos como algoritmos globais.

A busca tabu faz uma analogia a um alpinista escalando uma montanha e o *simulated annealing*, ao processo de resfriamento de um material em alta temperatura. Por outro lado, os algoritmos evolutivos estão baseados no processo de evolução natural e simulam o mecanismo de adaptação dos seres vivos ao meio ambiente ao longo de gerações. A partir de uma população de indivíduos (possíveis soluções para um problema concreto) um processo de competição é simulado, onde os mais aptos possuem maiores chances de sobreviver e se reproduzir.

A estratégia evolutiva é um dos processos evolutivos: através de mutações, uma população de um ou mais ancestrais geram um ou mais filhos que competem entre si ou substituem os pais. Outro é a programação evolutiva: cada ancestral gera um filho através de mutações; a melhor metade dos ancestrais e dos descendentes formam uma

nova população. Porém, o método evolutivo mais conhecido é o algoritmo genético, que será descrito a seguir.

Os mecanismos naturais, que promovem a evolução dos seres vivos, podem ser considerados como inteligentes, pois, permitem, através de alguns mecanismos a auto-organização de uma coletividade para atingir um objetivo.

2.3 De Lamarck ao neodarwinismo

A primeira teoria sobre a origem da variabilidade das formas de vida foi desenvolvida por Lamarck (1744 - 1829). Após uma observação meticulosa, o pesquisador concluiu que através do *uso* dos órgãos, os seres vivos os desenvolvem de acordo com as condições do meio ambiente; por outro lado, o *desuso* de algumas partes do organismo, provoca o seu atrofiamento e todas essas modificações são transmitidas hereditariamente aos seus descendentes. Ou seja, a sua teoria apóia-se no princípio do *uso e desuso* para justificar as mudanças ocorridas nos seres vivos ao longo dos séculos.

Algumas décadas depois, Darwin (1809 - 1882) desenvolveria a teoria da Seleção Natural. A partir de observações em uma viagem às ilhas Galápagos e à América do Sul e da teoria de Malthus sobre o crescimento populacional, percebeu que as variações entre os indivíduos de uma mesma espécie podem torná-los mais ou menos adaptados ao meio ambiente e as mudanças favoráveis tendem a predominar nas futuras gerações diante das desfavoráveis.

Dessa forma, os indivíduos que preenchem melhor as funções da natureza — por sua força, velocidade, coragem, resistência, saúde ou beleza — tendem a deixar uma descendência mais numerosa. Em 1859, Darwin publicou a sua grande obra: *A origem das Espécies, por Via de Seleção Natural*.

Porém Darwin não conseguiu explicar de onde vinham as variações das espécies e como eram herdadas. Em 1930, um grupo de cientistas formulou uma teoria entrelaçando o darwinismo com as leis de hereditariedade de Mendel e o conceito de mutação, que ficou conhecida como Teoria Sintética da Evolução ou Neodarwinismo. Assim foi possível responder algumas questões que Darwin deixou em aberto.

No Neodarwinismo, as causas que permitem o desenvolvimento evolutivo das espécies são as mutações ao acaso e a seleção natural. Como as mutações podem ser bem sucedidas, indiferentes ou prejudiciais a um organismo (dependendo dos seus efeitos e do meio ambiente), alguns indivíduos poderão adaptar-se melhor e sobreviver com mais probabilidade, deixando um número maior de descendentes até que, pela seleção natural, chegarão a predominar em toda população.

Assim, a mutação é considerada a matéria-prima da evolução e a seleção natural, o mecanismo de escolha das características mais vantajosas ou de redução das menos aptas.

2.4 O Algoritmo Genético: parâmetros e operadores

O Algoritmo Genético trabalha essencialmente com os mecanismos evolutivos do Neodarwinismo: a mutação e a seleção natural. A partir de uma população artificial inicial, onde cada indivíduo é uma solução potencial para um problema de otimização, novas gerações são formadas através da seleção ponderada dos pais (semelhante à seleção natural), onde os melhores indivíduos possuem maior probabilidade de serem escolhidos para reproduzirem. Os pais escolhidos sofrerão cruzamentos (recombinação do material genético entre os indivíduos) e mutações aleatórias. Quando o critério de parada for satisfeito, a última geração deverá possuir um conjunto de boas soluções para o problema.

Abaixo podemos observar um pseudocódigo do Algoritmo Genético.

```
t = 0
Gere uma população inicial
Avalie a aptidão da população inicial
Enquanto (não - critério de parada)
    t = t + 1
    selecione os cromossomos reprodutores
    faça o cruzamento dos reprodutores
    gere as mutações nos novos indivíduos
    avalie a aptidão de cada indivíduo
    atualize a população
Fim - enquanto
```

Os tópicos a seguir apresentam uma descrição detalhada de cada parâmetro ou operador do Algoritmo Genético.

2.4.1 Tamanho da população

O primeiro parâmetro que deve ser determinado é o tamanho da população (N), que pode ser fixo ou variável. Na natureza certamente o tamanho de uma população é variável,

havendo eventualmente algumas restrições limitantes: a escassez de alimentos, os predadores, o espaço físico ou as condições climáticas. Caso o tamanho seja variável, deve-se estipular um número máximo e, se conveniente, um mínimo; a população oscilará ao longo das gerações entre esses valores. Logicamente, o número de filhos a cada geração deverá acompanhar essas oscilações.

2.4.2 Codificação das variáveis

O algoritmo genético tradicional, para problemas de otimização contínua, trabalha geralmente com variáveis binárias. Assim, cada indivíduo da população é representado por um cromossomo formado por uma sequência de zeros e uns. Para converter uma variável binária em real, o algoritmo utilizará as seguintes equações:

$$x = a + yp$$

$$p = \frac{(b - a)}{2^l - 1}$$

onde:

$$a \leq x \leq b$$

p é a precisão utilizada

l é o comprimento da parte do cromossomo que corresponde à variável x

y é a representação real da sequência binária

Podemos ver claramente essa codificação em um dos problemas de minimização em caixas estudados no capítulo III: a função de *Rosenbrock*.

$$f(x) = \sum_{i=1}^{n-1} (100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2)$$

onde:

$$-2,048 \leq x_i \leq 2,048$$

Para cinco variáveis ($i = 5$) e com precisão de 10^{-3} , o cromossomo de cada indivíduo possui 60 genes (12 para cada variável). A tabela 2.1 apresenta a transformação do código binário para a forma real de uma variável.

Tabela 2.1: Transformação do código binário para a forma real

<i>Binário</i>	<i>Inteiro (y)</i>	<i>Real (x)</i>
000.000.000.001	1	-2,047
000.000.000.010	2	-2,046
000.000.000.011	3	-2,045
100.110.010.010	2450	0,402
111.111.111.000	4088	2,040

Uma variação do código binário é o código *Gray*, que normaliza as distâncias de Hamming (número de bits distintos entre cromossomos) de modo que os números vizinhos possuem a distância de uma unidade. A tabela 2.2 apresenta uma comparação de números de 3 bits binários e com o Código *Gray* e as distâncias de Hamming com relação ao seus antecessores.

Tabela 2.2: Comparação entre os números binários e o código *Gray*

<i>Decimal</i>	<i>Binário</i>	<i>dH</i>	<i>Gray</i>	<i>dH</i>
0	000	-	000	-
1	001	1	001	1
2	010	2	011	1
3	011	1	010	1
4	100	3	110	1
5	101	1	111	1

Outros algoritmos utilizam diretamente variáveis reais para representar os indivíduos de uma população. Cada variável representa um gene do cromossomo e são utilizados operadores específicos de mutação e recombinação, que serão descritos nos próximos itens.

2.4.3 População inicial

Geralmente é aleatória. Para a codificação binária ou o código *Gray*, gera-se uma sequência de zeros e uns do tamanho do cromossomo; para variáveis reais, geram-se aleatoriamente números reais entre os limites de cada variável.

Eventualmente, a população inicial pode não ser aleatória: a partir do conhecimento de um problema específico, pode-se gerar alguns indivíduos através de fórmulas ou de algum outro procedimento. Outra possibilidade é adotar como população inicial a última geração de outro processo do mesmo problema ou de outro semelhante.

2.4.4 Recombinação ou *Crossover*

Um dos principais operadores do Algoritmo Genético, a recombinação cromossômica (ou *Crossover*) consiste na troca de genes entre dois pais para gerar um ou mais filhos. Geralmente está associada a uma probabilidade de recombinação p_c , fixa ou variável. Assim, parte de uma nova população será formada por filhos provenientes de recombinações (semelhante a uma reprodução sexuada, com mistura de material genético de dois reprodutores) e a outra parte, por cópias idênticas aos pais (reprodução assexuada).

Quando se utiliza a codificação binária, a recombinação consiste na troca de partes dos cromossomos dos pais para gerar os filhos. Essa troca pode ser definida por um, dois ou mais pontos sorteados aleatoriamente, como se pode observar nos exemplos a seguir:

Recombinação em um ponto

$$\begin{aligned} \text{Pai1} &= (1000|0111) \\ \text{Pai2} &= (\mathbf{1110}|\mathbf{1001}) \\ &\quad \Downarrow \\ \text{Filho1} &= (1000|\mathbf{1001}) \\ \text{Filho2} &= (\mathbf{1110}|0111) \end{aligned}$$

Recombinação em dois pontos

$$\begin{aligned} \text{Pai1} &= (10|0001|11) \\ \text{Pai2} &= (\mathbf{11}|\mathbf{1010}|\mathbf{01}) \\ &\quad \Downarrow \\ \text{Filho1} &= (10|\mathbf{1010}|11) \\ \text{Filho2} &= (\mathbf{11}|0001|\mathbf{01}) \end{aligned}$$

Se o algoritmo trabalha com variáveis reais, a troca de informações entre os pais ocorre através de operações aritméticas, como a média aritmética:

$$Filho = \left(\frac{Pai1 + Pai2}{2} \right)$$

a média geométrica:

$$Filho = \sqrt{Pai1 \cdot Pai2}$$

ou a combinação convexa:

$$Filho1 = a \cdot Pai1 + (1 - a) \cdot Pai2$$

$$Filho2 = (1 - a) \cdot Pai1 + a \cdot Pai2$$

onde:

$$0 < a < 1$$

a pode ser aleatório ou depender do número de gerações

Outra recombinação possível é a *blend* - α , que permite gerar filhos fora dos pontos intermediários entre os pais:

$$Filho = Pai1 + a(Pai1 - Pai2)$$

onde:

a é um número aleatório no intervalo $(-\alpha, 1 + \alpha)$

A recombinação *blend* - α tenta evitar a tendência de a população convergir para uma espécie de média aritmética entre todos os indivíduos; no entanto, a probabilidade dos descendentes gerados serem ineficazes é maior.

2.4.5 Mutação

Outro importante operador do Algoritmo Genético é a mutação, que consiste na alteração aleatória de um gene. A probabilidade p_m de ocorrer uma mutação pode ser fixa ou variável.

Para uma variável binária, a mutação é imediata: um gene igual à *zero* será alterado para *um* e vice-versa, como pode ser observado no exemplo a seguir.

$$\begin{array}{c} (10000111) \\ \Downarrow \\ (10010111) \end{array}$$

Já para variáveis contínuas, a mutação poderá ocorrer de diversas formas. Uma possibilidade é a mutação aleatória (ou uniforme), que substitui um gene por qualquer outro valor no intervalo factível de uma variável. Esse método tende a introduzir uma diversidade maior na população do que a mutação binária. A seguir observa-se uma mutação aleatória para um cromossomo com cinco genes, todos no intervalo (0, 1).

$$\begin{array}{c} (0,21 \ 0,42 \ \mathbf{0,86} \ 0,09 \ 0,83) \\ \Downarrow \\ (0,21 \ 0,42 \ \mathbf{0,17} \ 0,09 \ 0,83) \end{array}$$

Outro procedimento possível é a mutação não-uniforme, que incrementa ou decrementa um gene, com igual probabilidade, de acordo com as equações:

$$x_i = \begin{cases} x_i + \theta(t, b_i - x_i) \\ x_i - \theta(t, a_i - x_i) \end{cases}$$

$$\theta(t, y) = yr \left(1 - \frac{t}{T}\right)^\alpha$$

onde:

r é um número aleatório no intervalo (0, 1)

t é a geração atual

T é o número máximo de gerações

α é o grau de dependência do número de gerações

a_i, b_i são os limites inferior e superior do gene x_i

Apesar de ser um método mais complexo do que as mutações aleatórias, a mutação não-uniforme permite um controle do grau de diversidade que convém dar a um problema.

Como será observado nos capítulos seguintes, altos graus de diversidade dificultam a convergência da algoritmo. A mutação não-uniforme permite que, com o decorrer das gerações, a diversidade possa ser menor, o que facilita a convergência da população.

2.4.6 Seleção

Outro mecanismo importante é a escolha dos pais que irão formar a futura geração.

O método mais comum é o da roleta, onde a probabilidade de um indivíduo ser escolhido é diretamente proporcional à sua aptidão (*fitness*). Assim, o sorteio de um indivíduo seria semelhante ao ato de girar uma roleta, onde a chance de parar em cada um é proporcional à sua adaptação ao meio e, conseqüentemente, os mais aptos tenderão a gerar mais descendentes.

Outro método possível de seleção baseia-se em sorteios. Com igual probabilidade para todos os indivíduos, pode-se sortear dois e selecionar o mais apto para reproduzir. Outro método, entre muitos outros, é a comparação de cada indivíduo com outros n sorteados aleatoriamente; a metade que possuir maior número de vitórias seria escolhida.

Os Algoritmos Genéticos também podem, a cada geração, formar uma população totalmente nova (apenas por filhos) ou manter uma porcentagem da anterior na nova.

2.4.7 Elitismo

Com os operadores e parâmetros apresentados até o momento, percebe-se que os melhores indivíduos tendem a gerar mais descendentes, porém não há uma garantia de que as melhores soluções permaneçam na população. Para não perdê-las durante a reprodução, através de uma recombinação ou uma mutação, o algoritmo possui um mecanismo conhecido como elitismo, que preserva intactos os melhores indivíduos de uma geração para outra. Alguns algoritmos substituem parte dos filhos (geralmente, os menos aptos) pelos melhores pais da geração anterior; outros simplesmente os acrescentam à nova população.

2.4.8 Cálculo da aptidão

A aptidão de um indivíduo (ou a função utilidade de uma solução), para um problema de minimização em caixas, é muitas vezes determinada pela equação a seguir.

$$\hat{f}(x) = f_{max} - f(x)$$

onde:

$f(x)$ é a função objetivo do problema

f_{max} é o valor máximo que $f(x)$ pode assumir no espaço factível ou nas últimas gerações

$\hat{f}(x)$ é o valor da aptidão (*fitness*) de cada indivíduo

A função aptidão assim definida sempre será positiva e os indivíduos mais aptos, ou seja, aqueles que possuírem menores valores para $f(x)$, terão os maiores para $\hat{f}(x)$.

O valor de f_{max} pode ser definido como o maior valor de cada geração (ou das últimas) ou do espaço factível. Como f_{max} influencia diretamente na aptidão dos indivíduos de uma geração (e, conseqüentemente, na seleção), convém que a ordem de grandeza deste parâmetro seja semelhante ao valor de $f(x)$. Por exemplo: se o valor de f_{max} é 1.000 e a função objetivo da população varia entre 20 e 50, a aptidão dos indivíduos será muito próxima, entre 950 e 980, e quase não haverá diferenças entre os melhores e os piores. No entanto, se f_{max} é igual a 60, os melhores indivíduos serão quatro vezes mais aptos que os piores e então, de fato, o algoritmo poderá selecionar com maior probabilidade os melhores indivíduos.

Em alguns algoritmos as aptidões dos pais influenciam os filhos; em outros a proximidade de indivíduos podem reduzir a avaliação de todos dentro de uma vizinhança, com o objetivo de aumentar a diversidade na população.

2.4.9 Critério de parada

Muitos algoritmos trabalham com um número máximo de gerações. Outros critérios de parada são: número máximo de gerações sem a alteração da melhor solução (ou com uma variação abaixo de um percentual definido) e um tempo máximo de processamento.

2.5 Aplicação do Algoritmo Genético

Podemos observar o funcionamento do Algoritmo Genético através de um problema com restrições de caixa, que será estudado no Capítulo III: a **função de Branin**.

$$f(x) = a(x_2 - bx_1^2 + cx_1 - d)^2 + e(1 - f) \cos(x_1) + e$$

onde:

$$a = 1, b = \frac{5,1}{4\pi^2}, c = \frac{5}{\pi}, d = 6, e = 10, f = \frac{1}{8\pi}$$

$$-5 \leq x_1 \leq 10 \text{ e } 0 \leq x_2 \leq 15$$

Figura 2.1: Gráfico da função de Branin

Trata-se de um problema de otimização de duas variáveis e que possui o mínimo global de $f(x) = 0,397887$ em 3 pontos ótimos:

$$(x_1, x_2) = (-\pi; 12, 275); (\pi; 2, 275) \text{ e } (9, 42478; 2, 475)$$

Inicialmente, criou-se uma população aleatória de 100 indivíduos, como podemos observar a seguir.

Figura 2.2: Gráfico da 1ª geração da função de Branin

Foi utilizado o código *Gray* com uma precisão de 10^{-3} para cada variável. A recombinação, definida em dois pontos, teve probabilidade de 0,8 e a mutação, de 0,001.

A seleção foi feita através do método da roleta e utilizada a estratégia elitista. A figura 2.2 mostra a distribuição da quinta geração. Observe que a maioria dos pontos já se encontra na região próxima aos pontos ótimos, havendo uma pequena concentração na vizinhança dos pontos $(\pi, 2, 2750)$ e $(9, 42478, 2, 475)$.

Figura 2.3: Gráfico da 5ª geração da função de Branin

Figura 2.4: Gráfico da 10^a geração da função de Branin

Na décima geração praticamente todos os pontos estão na vizinhança de $(9, 42478, 2, 475)$, alguns poucos na de $(\pi, 2, 2750)$ e verificamos claramente que o algoritmo convergiu com apenas dez gerações.

Outro exemplo é a **função Easom**, que possui duas variáveis e um ótimo global: (π, π) .

$$f(x) = -\cos x_1 \cdot \cos x_2 \cdot e^{-((x_1-\pi)^2+(x_2-\pi)^2)}$$

onde:

$$-100 \leq x_i \leq 100, i = 1, 2$$

mínimo global: $f(x^*) = -1$ e $x_i^* = \pi, i = 1, 2$

Figura 2.5: Gráfico da função de Easom

Utilizando os mesmos parâmetros do problema anterior, pode-se observar que em poucas gerações o algoritmo converge.

Figura 2.6: Gráficos da função de Easom (gerações)

Note que na décima geração a população não está tão dispersa quanto a inicial e os pontos estão “caminhando” em direção à cavidade do ótimo global. Na geração quinze alguns pontos já o atingiram e na vigésima, o algoritmo convergiu.

2.6 Teorema fundamental do Algoritmo Genético

2.6.1 Alguns conceitos

O fundamento teórico do Algoritmo Genético com variáveis binárias baseia-se na noção de *schemata*: uma seqüência de genes definidos que formam uma classe ou família de cromossomos. Um *schemata* é construído introduzindo alguns genes indefinidos em um cromossomo, simbolizados por um caractere ($\#$), e representa toda a família dos indivíduos que possuem qualquer gene nessas posições.

Seja um cromossomo de dez genes, com dois indefinidos: 011001#01#. Este *schemata* representa os quatro cromossomos a seguir: 0110011011, 0110011010, 0110010011 e 0110010010. Também podemos dizer que o *schemata* 011001#01# representa o espaço desses cromossomos. Logicamente, um *schemata* formado apenas por números (0111110010) representa apenas um cromossomo e um *schemata* formado apenas por caracteres (#####) simboliza todos os cromossomos possíveis.

Os *schemata* diferenciam-se devido ao número de genes definidos e à posição destes. Assim, podemos definir duas propriedades para estudá-los: ordem de um *schemata* e comprimento definido [13].

A ordem é o número de genes definidos, fixos, que um cromossomo possui. Nos exemplos anteriores, podemos verificar a ordem de alguns cromossomos:

$$\begin{aligned} o(011001\#01\#) &= 8 \\ o(0111110010) &= 10 \\ o(\#####) &= 0 \end{aligned}$$

Comprimento definido é a distância entre o primeiro e o último cromossomo fixo. Assim, os comprimentos definidos dos cromossomos acima são:

$$\begin{aligned} \delta(011001\#01\#) &= 9 - 1 = 8 \\ \delta(0111110010) &= 10 - 1 = 9 \\ \delta(\#####) &= 0 - 0 = 0 \end{aligned}$$

Estas propriedades permitem-nos observar os efeitos dos mecanismos de seleção, recombinação e mutação em um *schemata*.

2.6.2 Teorema fundamental do Algoritmo Genético

Seja uma população $P(t)$ de n indivíduos na t -ésima geração. Seja também um *schemata* H com $m(H, t)$ representantes na população $P(t)$.

Seleção: A probabilidade de um indivíduo representante de um *schemata* H ser escolhido é proporcional à aptidão média $f(H)$ da família H . Assim, o número de descendentes que a família H deve possuir na geração $t + 1$ será:

$$m(H, t + 1) = m(H, t) \cdot n \cdot \frac{f(H)}{\sum_{j=1}^n f_j}$$

sendo $\bar{f} = \frac{\sum_{j=1}^n f_j}{n}$ a aptidão média da geração t , então:

$$m(H, t + 1) = m(H, t) \cdot \frac{f(H)}{\bar{f}}$$

Note que o número dos representantes do *schemata* H na geração $t + 1$ tende a ser proporcional à razão da média da aptidão da família do *schemata* pela média da aptidão de toda geração. Assim, se a média da aptidão do *schemata* for superior à média geral, haverá mais representantes na próxima geração e, no entanto, se a média for inferior, o número de indivíduos da família H tenderá a diminuir.

Cruzamento (ou Crossover): Uma vez selecionado, um membro da família do *schemata* H poderá ser destruído pelo processo de cruzamento. A probabilidade de o *schemata* sobreviver dependerá da probabilidade de ocorrer um cruzamento (p_c) e, caso ocorra, da chance de acontecer fora do comprimento definido do *schemata*.

Supondo que o cruzamento seja definido por apenas um ponto, a probabilidade de ocorrer dentro do comprimento definido será de $\frac{\delta(H)}{l-1}$, onde l é o número de genes do cromossomo.

Assim, a probabilidade de sobrevivência p_s de um *schemata* H será:

$$p_s \geq 1 - p_c \cdot \frac{\delta(H)}{l-1}$$

Uma vez que o número de genes é fixo e considerando que a probabilidade de recombinação também seja, concluímos que os *schemata* de menor comprimento definido possuirão maior probabilidade de sobreviverem. Abaixo, podemos observar o

número de descendentes que um *schemata* H deve possuir após o processo de seleção e recombinação:

$$m(H, t + 1) = m(H, t) \cdot \frac{f(H)}{\bar{f}} \cdot \left[1 - p_c \cdot \frac{\delta(H)}{l - 1} \right]$$

É imediato observar que os efeitos da seleção e recombinação somam-se: os *schemata* mais aptos (com aptidão maior que a média da população) e com comprimento definido pequeno, tenderão a deixar mais descendentes.

Mutação: O último operador que influencia o número de descendentes do *schemata* da família H é a mutação. Para um *schemata* sobreviver, nenhum dos genes definidos poderá ser alterado. Assim, a probabilidade de um gene não mudar será $(1 - p_m)$. Como há $o(H)$ genes fixos e a probabilidade de ocorrer uma mutação em cada um dos genes é independente, a probabilidade de o *schemata* sobreviver será:

$$p_s = (1 - p_m)^{o(H)}$$

Considerando que a probabilidade de mutação é pequena ($p_m \ll 1$), podemos aproximar a expressão anterior:

$$p_s = (1 - o(H) \cdot p_m)$$

Assumindo que a probabilidade de mutação seja fixa, a probabilidade de sobrevivência de um *schemata* será maior quanto menor for a sua ordem.

Somando o efeito dos três operadores, podemos concluir que o número de descendentes de um *schemata* H será:

$$m(H, t + 1) = m(H, t) \cdot \frac{f(H)}{\bar{f}} \cdot \left[1 - p_c \cdot \frac{\delta(H)}{l - 1} - o(H) \cdot p_m \right]$$

Portanto, os *schematas* que possuem pequeno comprimento definido, baixa ordem e alta aptidão, tendem a predominar na população, ou seja, pequenas seqüências de genes

muito bem adaptados tendem a espalhar-se rapidamente pelos indivíduos da população. Isto se adapta perfeitamente à teoria de Darwin: seu livro *A Origem das Espécies* termina de forma quase poética:

“Há uma grandeza simples no fato de considerar a vida, com as suas capacidades de desenvolvimento, assimilação e reprodução (...), através do processo de seleção gradual de mudanças infinitesimais, até chegarem a uma quantidade infinita de formas belíssimas e admiráveis”¹.

Darwin considerava que as pequenas mudanças — que se devem às mutações, como mais tarde se descobriu — podem produzir indivíduos muito bem adaptados. O Teorema Fundamental do Algoritmo Genético conclui o mesmo: as pequenas estruturas com alta aptidão tendem a espalhar-se por toda a população, levando o algoritmo às melhores soluções.

2.6.3 Exemplos

Podemos observar isto claramente na função Easom. A melhor solução (π, π) , possui a seguinte seqüência binária, com precisão de 10^{-3} :

110001100000011111110001100000011111

Para as mesmas condições descritas no item 2.5, observou-se o crescimento do número de indivíduos da família de um *schemata* (H_1):

$H_1 = #####1100#####$

¹Darwin, Charles. *A Origem das Espécies*. Esboço de 1842. Clássicos Econômicos Newton, Rio de Janeiro, 1996

Figura 2.7: Gráfico do número de *schemata* H_1 para cada geração da função Easom

Podemos notar que, por ser uma seqüência bem adaptada, o *schemata* possui um grande crescimento: em 25 gerações, a seqüência espalha-se por quase todos os indivíduos da população. Entre as gerações quinze e vinte e um, por exemplo, a média da aptidão das soluções da família H_1 foi de 23% a 53% maior que a média de toda população.

Outro *schemata* observado foi o H_2 :

$$H_2 = \#0110\#\#\#\#\#\#\#\#\#\#$$

Figura 2.8: Gráfico do número de *schemata* H_2 para cada geração da função Easom

Note que da décima para a trigésima geração o número de indivíduos da família H_2 sobe de 5 para mais de 90 representantes. Na décima quarta geração, por exemplo, a média dos *schemata* H_2 foi 235% maior que a média de toda população.

2.7 Convergência do algoritmo

Algumas vezes o algoritmo genético pode encontrar rapidamente boas soluções, porém distantes do ótimo global, e a busca pode convergir para esses pontos. Este fenômeno é conhecido como convergência prematura: em poucas gerações localizam-se ótimos locais e o algoritmo converge, podendo desviar-se das regiões mais promissoras, que possuem as melhores soluções.

Para que o algoritmo tenha uma maior chance de encontrar o ótimo global, convém que haja uma variedade grande numa população e que se gaste um número razoável de gerações explorando de modo significativo a região factível. Em outras palavras, o Algoritmo Genético requer um tempo mínimo para “conhecer a região factível” e, com maior probabilidade, encontrar as melhores soluções.

Assim, muitas vezes é prudente adotar algum mecanismo para evitar a convergência pré-matura. Um deles é aumentar a diversidade de soluções, substituindo as mais semelhantes por outras geradas aleatoriamente ou diminuindo a avaliação dos pontos muito próximos.

Outras vezes, a convergência prematura deve-se à existência de um superindivíduo, que possui uma aptidão muito superior à dos outros e gera muitos descendentes, ou muitos infraindivíduos, que pouco contribuem para a busca. Nesses casos, pode-se estipular alguma avaliação mínima ou normalizar a aptidão “bruta” de cada indivíduo da população.

Há inúmeras maneiras de normalizar a aptidão de uma população:

Escala linear: A aptidão de cada indivíduo é regulada por uma função linear:

$$\hat{f}(x) = a \cdot f(x) - b$$

onde:

x é um indivíduo da população

$f(x)$ é a avaliação bruta da solução x

a e b são constantes que regulam o nível de competição entre os indivíduos

Escala sigma: A aptidão é ajustada por parâmetros provenientes da própria população:

$$\hat{f}(x) = f(x) - (\bar{f} - c \cdot \sigma)$$

onde:

x é um indivíduo da população

$f(x)$ é a avaliação bruta

$\bar{f}$ é a avaliação média da população

σ é o desvio padrão da população

c é uma constante, geralmente entre 1 e 5

Escala de potência: A aptidão é transformada através de uma função exponencial:

$$\hat{f}(x) = f(x)^k$$

onde:

x é um indivíduo da população

$f(x)$ é a avaliação bruta

k é uma constante, geralmente próxima a 1,005

Ranking A população é ordenada de acordo com a aptidão; o melhor indivíduo possui uma avaliação máxima e o pior, uma mínima. Todos os outros indivíduos possuem uma aptidão intermediária, proporcional a sua posição no ranking.

2.8 Outras técnicas

Existem inúmeras técnicas que podem adaptar-se perfeitamente ao Algoritmo Genético e potencializar a sua busca. Não se fará aqui uma lista exaustiva das variações possíveis: serão citados apenas alguns exemplos, não utilizados nesse trabalho, para que o leitor possa contemplar a grande gama de opções que o algoritmo proporciona.

Medidas de similaridade: O algoritmo pode trabalhar com medidas de similaridade para manter a diversidade de uma população. Para isso, preservam-se nichos de indivíduos ao longo da busca, criando subpopulações dentro de um ecossistema. A aptidão de cada indivíduo será influenciada pelo número de soluções dentro de seu nicho. os filhos poderão substituir o pior indivíduo, caso seja mais apto do que este.

Algoritmo Genético híbrido: Facilmente podemos combinar o algoritmo genético com algoritmos de busca local ou outras heurísticas e encontrar um equilíbrio entre a robustez do Genético e a rapidez dos outros métodos: somam-se as qualidades e pode-se obter um algoritmo mais eficaz. Os melhores indivíduos de cada geração, ou somente os da última, serão submetidos à busca local. Os métodos exatos podem ser utilizados neste processo.

Controle adaptativo de parâmetros: Para aumentar a eficiência da busca, os parâmetros do algoritmo podem ser ajustados de acordo com o estado evolutivo do sistema. Essa adaptação pode depender do número de gerações ou do tempo (variação determinística) ou a partir de informações capturadas do próprio sistema evolutivo. Por exemplo, no início da busca, pode-se ter uma probabilidade de mutação grande, para estimular a diversidade, e reduzi-la ao longo das gerações para permitir a convergência do método.

Número de populações: O algoritmo também pode trabalhar com várias populações, evoluindo-as paralelamente. Diversos modelos já foram implementados:

- *mestre-escravo*: um processo mestre controla todos os operadores de diversas buscas independentes (escravos);
- *memória comum*: diversas buscas simultâneas são realizadas, formando uma memória comum com as melhores soluções de cada processo;
- *processos em rede*: diversas populações são criadas e evoluídas. A cada geração, o melhor indivíduo de cada população é copiado para as outras;
- *modelo de polinização de plantas*: um conjunto de populações é criado, como um jardim de flores, e cada população possui uma probabilidade de reproduzir com indivíduos das outras, como um pólen que pode cair em uma mesma planta ou em outras distintas.

Tamanho da população: Geralmente os Algoritmos Genéticos trabalham com populações de tamanhos fixos. Porém pode-se associar a cada indivíduo uma estimativa de vida e também uma taxa de natalidade a cada geração, de modo que o tamanho da população possa oscilar ao longo da evolução.

Genes dominantes e recessivos: A natureza possui indivíduos com apenas uma sequência de genes (haplóides), geralmente seres mais primitivos, e com dois cromossomos (diplóides), formas de vida mais complexas. O algoritmo genético também pode trabalhar com cromossomos diplóides, genes dominantes e recessivos, genótipos e fenótipos ou outros conceitos da genética moderna.

Explorar conhecimento do problema: Outra possibilidade é desenvolver mecanismos ou operadores específicos para um problema determinado. A partir das carac-

terísticas de uma região factível ou das variáveis, é possível determinar recombinações ou mutações que ajudem a explorar melhor os pontos viáveis. O capítulo IV aborda alguns métodos que adaptam os operadores às características de um problema.

2.9 Vantagens e desvantagens do Algoritmo Genético

Após o estudo do Algoritmo Genético e do seu teorema fundamental, podemos abordar as suas qualidades e desvantagens.

Primeiramente, como se trabalha com uma população de soluções, além do ótimo global, o algoritmo pode gerar muitas outras boas soluções (semelhantes ou distintas da melhor). Além disso, é um algoritmo de otimização global, podendo também encontrar vários ótimos locais. Em problemas de engenharia, muitas vezes, a melhor solução pode ser inviável por outros motivos que não foram considerados na modelagem matemática (uma peça de difícil fabricação, um carro de estética não privilegiada, etc.). Assim, a existência de inúmeras boas soluções torna o algoritmo atrativo para problemas reais.

Outra vantagem significativa é que o algoritmo consegue explorar a região de busca sem ficar preso em pontos de mínimos locais, o que é comum nos algoritmos que trabalham com derivadas. Um exemplo é uma das funções que será estudada no próximo capítulo: a função *Ackley's Path*. Como podemos observar no gráfico 2.9, a sua superfície possui o perfil de uma “caixa de ovo”, com inúmeros mínimos locais. O Algoritmo Genético em poucas gerações encontrou o mínimo global ($x_i = 0$).

Função Ackley's Path:

$$f(x) = -a \cdot e^{-b \cdot \sqrt{\frac{1}{n} \cdot \sum_{i=1}^n x_i^2}} - e^{\frac{1}{n} \cdot \sum_{i=1}^n \cos(c \cdot x_i)} + a + e^1$$

onde:

$$a = 20 \quad b = 0,2 \quad c = 2\pi$$

$$-32,768 \leq x_i \leq 32,768$$

Por outro lado, o Algoritmo Genético é robusto. Por ser uma analogia ao processo de evolução natural, o algoritmo possui a “vitalidade” de adaptar-se facilmente aos diversos tipos de problemas, tendendo a encontrar boas soluções. Não trabalha com derivadas e sim com a função de custo diretamente. Deste modo, não possui problemas de descontinuidade de funções, de inexistência de derivadas ou outras dificuldades apresentadas por outros métodos de busca.

Figura 2.9: Gráfico da função de Ackley's Path

Outra vantagem é que o Algoritmo Genético trabalha com variáveis discretas e/ou contínuas, enquanto que a maioria dos métodos trabalham apenas com um dos tipos.

Por último, o Algoritmo Genético, como já foi abordado no item anterior, pode facilmente ser combinado como outros processos de busca, o que aumenta a sua eficácia.

No entanto o Genético possui duas desvantagens. Geralmente, é mais caro que outras técnicas, pois, trabalha com conjuntos de populações, e possui muitos parâmetros que precisam ser bem ajustados para aumentar a sua eficácia.

Esse estudo possui o objetivo de reduzir essa segunda desvantagem: procurou-se determinar as faixas dos parâmetros do algoritmo em que são obtidos os melhores resultados para algumas classes de problemas, o que ajuda a reduzir o tempo de ajuste desses parâmetros.

Problemas de minimização em caixas

Para o estudo dos problemas de minimização em caixas, utilizamos o popular *GENetic Search Implementation System* (GENESIS, Versão 5.0), idealizada por Jonh J. Grefenstette [14] em linguagem C. Trata-se de uma versão clássica do Algoritmo Genético que, com uma interface simplificada, permite o ajuste de muitos parâmetros e a escolha de diversas opções do algoritmo: probabilidade de mutação, de recombinação, tamanho da população, número de gerações, tamanho de um *string*, estratégia elitista, uso do código binário ou do código *Gray*, maximizar ou minimizar a função objetivo, utilizar um *ranking* dos indivíduos ao selecioná-los, imprimir arquivos com a população final ou a estatística da geração atual, iniciar o algoritmo a partir de um conjunto inicial, etc.

3.1 Conjunto de problemas teste

Para estudar os problemas com restrições de caixa, escolhemos 13 funções clássicas [30], algumas difíceis de resolver, devido ao grande número de mínimos locais, e que são especialmente interessantes para o Algoritmo Genético. O apêndice A contém uma descrição detalhada de todas as funções. Cada uma foi estudada em diferentes dimensões: cinco e dez, duas e quatro ou duas e seis dimensões. Todos os resultados foram gerados com as seguintes especificações:

- estratégia elitista: o melhor indivíduo, o mais apto, sempre permanece na população;
- uso do código *Gray*;
- função aptidão: consideramos que f_{max} assume o valor máximo da função objetivo nas últimas 5 gerações;
- normalização da função aptidão por ranqueamento, onde o menos apto possui uma

aptidão x ($0 < x < 1$) e o melhor indivíduo, $(2 - x)$; todos os demais possuem valores intermediários, proporcionais a sua posição no ranking;

- recombinação em dois pontos.

Com relação aos outros parâmetros do Algoritmo Genético, para evitar a abordagem de situações particulares, os problemas testes foram resolvidos em três cenários:

<i>Cenário</i>	<i>A</i>	<i>B</i>	<i>C</i>
Tamanho da população (N)	50	100	50
Número máximo de gerações	500	500	1000
Probabilidade de recombinação (p_c)	0,7	0,8	0,9
Probabilidade de mutação (p_m)	0,001	0,005	0,010
Variável de ranqueamento	0,3	0,1	0,5
Semente	123456789	314159265	271828182

Tabela 3.1: Cenários A, B e C

Com o objetivo de estudar a influência da probabilidade de recombinação (e dos outros parâmetros nos resultados do Algoritmo Genético), para cada um dos cenários, todos os parâmetros foram fixados, exceto a probabilidade de recombinação, que variou de 0,5 e 1,0.

A tabela 3.2 apresenta os resultados do primeiro problema, a função de Michalewicz com dimensão cinco, para o cenário A.

$$f(x) = - \sum_{i=1}^n \left(\sin(x_i) \cdot \sin\left(\frac{i \cdot x_i^2}{\pi}\right)^{2m} \right)$$

onde:

$$m = 10, n = 5$$

$$0 \leq x_i \leq \pi$$

Nota-se que, apesar de todos os resultados terem se aproximado da melhor solução conhecida, somente algumas faixas de variação a atingiram.

p_c	Melhor solução	x_1	x_2	x_3	x_4	x_5	ger (x) (x)
0,50	-4,5376	2,2026	1,5712	1,2850	1,9233	0,9966	307
0,52	-4,3749	2,2026	1,5712	2,2194	1,9233	0,9966	233
0,54	-4,3749	2,2026	1,5712	2,2194	1,9233	0,9966	233
0,56	-4,5376	2,2026	1,5712	1,2850	1,9233	0,9966	286
0,58	-4,5376	2,2026	1,5712	1,2850	1,9233	0,9966	286
0,60	-4,3749	2,2026	1,5712	2,2194	1,9233	0,9966	345
0,62	-4,3749	2,2026	1,5712	2,2194	1,9233	0,9966	345
0,64	-4,6876	2,2026	1,5712	1,2850	1,9233	1,7208	125
0,66	-4,6876	2,2026	1,5712	1,2850	1,9233	1,7208	125
0,68	-4,6876	2,2026	1,5712	1,2850	1,9233	1,7208	298
0,70	-4,6876	2,2026	1,5712	1,2850	1,9233	1,7208	298
0,72	-4,3749	2,2026	1,5712	2,2194	1,9233	0,9966	247
0,74	-4,3749	2,2026	1,5712	2,2194	1,9233	0,9966	247
0,76	-4,5248	2,2026	1,5712	2,2194	1,9233	1,7208	287
0,78	-4,5248	2,2026	1,5712	2,2194	1,9233	1,7208	287
0,80	-4,5248	2,2026	1,5712	2,2194	1,9233	1,7208	144
0,82	-4,5248	2,2026	1,5712	2,2194	1,9233	1,7208	144
0,84	-4,5248	2,2026	1,5712	2,2194	1,9233	1,7208	154
0,86	-4,5248	2,2026	1,5712	2,2194	1,9233	1,7208	154
0,88	-4,4831	2,2026	1,5712	2,2194	1,1139	1,7208	271
0,90	-4,4831	2,2026	1,5712	2,2194	1,1139	1,7208	271
0,92	-4,3749	2,2026	1,5712	2,2194	1,9233	0,9966	102
0,94	-4,3749	2,2026	1,5712	2,2194	1,9233	0,9966	102
0,96	-4,6876	2,2026	1,5712	1,2850	1,9233	1,7208	306
0,98	-4,6876	2,2026	1,5712	1,2850	1,9233	1,7208	306
1,00	-4,6876	2,2026	1,5712	1,2850	1,9233	1,7208	44

Melhor solução conhecida: $f(x) = -4,6876$,
com $x = (2,2026; 1,5712; 1,2850; 1,9233; 1,7208)$.

Tabela 3.2: Função de *Michalewicz* no cenário A — resultados

3.2 Função penalidade

Para que fosse possível uma comparação entre as soluções obtidas para um problema e entre as soluções dos diversos problemas teste, criou-se uma medida de penalidade (γ). Quatro fatores foram ponderados: o esforço do algoritmo (o número de pontos avaliados ao longo de todas as gerações), a velocidade com que a melhor solução é encontrada, a diferença de valor da função objetivo e a distância entre a melhor solução encontrada e a melhor conhecida.

Assim, seja x^* a melhor solução conhecida de um problema. A penalidade γ será:

$$\gamma = \frac{ger \cdot N}{\alpha_1} + \alpha_2 \cdot ger(x) + \alpha_3 \cdot (f(x) - f(x^*)) + \alpha_4 \cdot \|x - x^*\|$$

onde:

ger é o número total de gerações

N é o tamanho da população

$ger \cdot N$ é o número de indivíduos que são avaliados até a última geração

$ger(x)$ é a geração da melhor solução encontrada

$(f(x) - f(x^*))$ é a diferença entre o melhor valor encontrado e o melhor conhecido

$\|x - x^*\|$ é a distância (norma euclidiana) entre a melhor solução encontrada e a conhecida

$\alpha_1, \alpha_2, \alpha_3$ e α_4 são os coeficientes que penalizam os fatores descritos anteriormente

Com os coeficientes de penalidade, busca-se penalizar os processos que necessitam avaliar muitos pontos para encontrar boas soluções, que necessitam de muitas gerações e que distam muito do melhor resultado conhecido. No entanto, esses fatores possuem importâncias diferentes:

- primeiramente deve-se penalizar as soluções que estejam distantes da melhor solução conhecida, portanto α_3 e α_4 devem possuir pesos maiores;
- depois, os problemas que necessitam avaliar muitos indivíduos: α_1 deve possuir uma dimensão grande, uma vez que divide o total dos indivíduos avaliados;
- em último lugar, o número de gerações necessárias para encontrar a melhor solução. O Algoritmo Genético não deve convergir em um número pequeno de gerações — pois, poderia gerar soluções prematuras e ruins — mas em um número razoável de gerações deve encontrar boas soluções. Portanto, o coeficiente α_2 deve possuir pequena magnitude.

Os valores adotados para esses coeficientes foram determinados experimentalmente: a partir de um estudo em um pequeno conjunto de problemas teste, com resultados de fácil comparação (onde todas as situações atingiram a melhor solução conhecida, por exemplo). Após alguns testes, os valores que claramente privilegiavam as melhores soluções nesse pequeno conjunto de problemas eram: $\alpha_1 = 4000$, $\alpha_2 = 1$, $\alpha_3 = 200$ e $\alpha_4 = 100$.

Note que a medida de penalidade escolhida é uma função que pondera quatro fatores (ou seja, uma função multi-objetivos), que poderiam ser abordados através de outros métodos, apresentados no capítulo 5. Uma possibilidade seria estudá-los procurando a linha frontal de Pareto; outra seria estabelecer uma ordem de prioridade entre os objetivos. No entanto, para facilitar a comparação entre as diversas soluções em todos os problemas estudados, optou-se por utilizar uma única função linear que pondera todos os objetivos.

Outras medidas de desempenho — como a *performance profiling*, que avalia o custo do processamento de diversos modos [18] — também poderiam ser utilizadas.

3.3 Método de estudo dos parâmetros

Assim, calculou-se a penalidade para o primeiro problema no cenário A, que foi transformada em uma grandeza entre 0 e 1 (penalidade normalizada — γ_A).

p_c	γ_A	γ
0,50	0,69	395,69
0,52	0,78	441,55
0,54	0,78	441,55
0,56	0,64	374,69
0,58	0,64	374,69
0,60	1,00	553,55
0,62	1,00	553,55
0,64	0,16	131,25
0,66	0,16	131,25
0,68	0,50	304,25
0,70	0,50	304,25
0,72	0,81	455,55
0,74	0,81	455,55

PR	γ_A	γ
0,76	0,72	413,11
0,78	0,72	413,11
0,80	0,44	270,11
0,82	0,44	270,11
0,84	0,46	280,11
0,86	0,46	280,11
0,88	0,84	470,98
0,90	0,84	470,98
0,92	0,52	310,55
0,94	0,52	310,55
0,96	0,52	312,25
0,98	0,52	312,25
1,00	0,00	50,25

p_c é a probabilidade de recombinação

γ_A é a penalidade normalizada no cenário A

onde:

$$\gamma_A = \frac{\gamma - \gamma_{min}}{\gamma_{max} - \gamma_{min}}$$

γ_{min} é a menor penalidade γ

γ_{max} é a maior penalidade γ

γ é a penalidade

Tabela 3.3: Função de *Michalewicz* no cenário A — penalidades

Podemos observar que, nesse exemplo, a menor penalidade ocorre quando a probabilidade de recombinação é igual a 1, ou seja, quando todos os indivíduos de uma nova geração são formados por reprodução sexuada.

Figura 3.1: Penalidade normalizada da função Michalewicz — no cenário A

A penalidade média dos três cenários para o primeiro problema, encontra-se na tabela a seguir.

p_c	γ_n	γ_A	γ_B	γ_C
0,50	0,45	0,69	0,35	0,32
0,52	0,33	0,78	0,03	0,17
0,54	0,64	0,78	0,98	0,17
0,56	0,65	0,64	0,30	1,00
0,58	0,88	0,64	1,00	1,00
0,60	0,50	1,00	0,23	0,28
0,62	0,49	1,00	0,20	0,28
0,64	0,10	0,16	0,07	0,08
0,66	0,10	0,16	0,05	0,08
0,68	0,24	0,50	0,20	0,01
0,70	0,21	0,50	0,11	0,01
0,72	0,56	0,81	0,72	0,16
0,74	0,42	0,81	0,29	0,16

p_c	γ_n	γ_A	γ_B	γ_C
0,76	0,38	0,72	0,22	0,19
0,78	0,35	0,72	0,12	0,19
0,80	0,21	0,44	0,18	0,00
0,82	0,46	0,44	0,95	0,00
0,84	0,42	0,46	0,46	0,36
0,86	0,30	0,46	0,08	0,36
0,88	0,39	0,84	0,14	0,19
0,90	0,38	0,84	0,12	0,19
0,92	0,23	0,52	0,06	0,13
0,94	0,28	0,52	0,19	0,13
0,96	0,29	0,52	0,12	0,23
0,98	0,25	0,52	0,00	0,23
1,00	0,13	0,00	0,15	0,23

p_c é a probabilidade de recombinação

γ_n é a média das penalidades normalizadas dos cenários A, B e C

γ_A , γ_B e γ_C são as penalidades normalizadas dos cenários A, B e C respectivamente

Tabela 3.4: Função de *Michalewicz* nos três cenários

Graficamente, podemos observar que, para a primeira função, acima de 0,62 a penalidade normalizada mantém-se abaixo de 0,4. Conclui-se que para essa função são recomendadas probabilidades de recombinação maiores que 0,62. A idéia deste capítulo é estender esse conceito a outros parâmetros e a toda classe de problemas de minimização em caixas.

Figura 3.2: Penalidade normalizada média da função Michalewicz

3.4 Probabilidade de recombinação

Os resultados para os problemas de minimização em caixas se encontram no Apêndice A.

Graficamente, observamos que a penalidade normalizada média indica que as altas probabilidades de recombinação (acima de 0,76) tendem a gerar menores penalidades ($\bar{\gamma}_n < 0,4$). Por outro lado, segundo a penalidade total normalizada, os melhores resultados ($\gamma_{tn} < 0,4$) encontram-se entre 0,74 e 0,82.

Figura 3.3: Penalidade normalizada média *versus* probabilidade de recombinação

Figura 3.4: Penalidade total normalizada *versus* probabilidade de recombinação

Essa aparente divergência de resultados provém da diferença conceitual entre essas duas medidas. A penalidade normalizada média é uma média das penalidades normalizadas e, portanto, possui uma curva suave, pois, atribui a mesma importância para cada

problema (uma medida de zero a um). Já a penalidade total normalizada é a conversão da soma total das penalidades; logo, um problema mais difícil, que possui uma alta penalidade, influencia mais a medida do que outros mais simples. Em outras palavras, se para uma probabilidade de recombinação o algoritmo gera um resultado muito distante da melhor solução conhecida, isso influenciará muito a penalidade total normalizada e terá uma influência muito menor na penalidade normalizada média.

Outros índices que podemos medir para determinar a melhor faixa de probabilidade de recombinação são: o número de problemas que possuem baixas penalidades normalizadas ($\gamma_{A, B \text{ ou } C} < 0,1$) em cada um dos cenários e o número médio de gerações em que surgiu a melhor solução encontrada.

Figura 3.5: Número de problemas com baixas penalidades *versus* probabilidade de recombinação

Figura 3.6: Número médio de gerações da melhor solução *versus* probabilidade de recombinação

Graficamente, observamos que o número de problemas com baixa penalidade tende a crescer e o número médio de gerações tende a diminuir timidamente com o aumento da probabilidade de recombinação. Principalmente com relação ao número de problemas com baixa penalidade (acima de 18), podemos indicar que uma faixa de probabilidade de recombinação entre 0,75 e 1,00 gera bons resultados.

Podemos concluir que os melhores resultados encontrados, bem como os intervalos de probabilidades mais vantajosos para problemas de minimização em caixas, de acordo com as medidas anteriores, são:

	p_c	<i>Intervalo</i>
penalidade normalizada média	1,00	0,76 a 1,00
penalidade total normalizada	0,78	0,74 a 0,82
número de problemas com baixa penalidade normalizada	1,00	0,76 a 1,00
número de gerações, dentro as melhores soluções	1,00	0,74 a 1,00
probabilidade média e intervalo principal	0,95	0,76 a 1,00

Tabela 3.5: Melhores resultados para probabilidade de recombinação

3.5 Probabilidade de mutação

O método utilizado para o estudo da melhor probabilidade de mutação é idêntico ao da de recombinação. Os resultados, para todas as funções nos cenários A, B e C, encontram-se no Apêndice A.

Figura 3.7: Penalidade normalizada média *versus* probabilidade de mutação

Figura 3.8: Penalidade total normalizada *versus* probabilidade de mutação

Graficamente, observamos que a penalidade normalizada média sugere uma probabilidade de mutação entre 0,010 e 0,060 ($\bar{\gamma}_n < 0,25$) e, a penalidade total normalizada, entre 0,020 e 0,060 ($\gamma_{tn} < 0,10$). Note que os gráficos possuem tendência semelhantes: probabilidades altas ou muito próximas de zero geram resultados ruins. No entanto, o

primeiro é mais suave por se tratar de uma média. Podemos observar que este operador ajuda a busca pela melhor solução, pois, todos os problemas apresentam resultados ruins com probabilidade igual a zero; porém, com altas probabilidades, a convergência do algoritmo é afetada.

Os resultados dos outros índices encontram-se nos gráficos a seguir.

Figura 3.9: Número de problemas com baixas penalidades *versus* probabilidade de mutação

Figura 3.10: Número médio de gerações da melhor solução *versus* probabilidade de mutação

Observa-se que o número de problemas com baixa penalidade possui bons resultados para probabilidades de mutação entre 0,005 e 0,040 (acima de 60). O número médio de gerações oscila entre 250 e 325 para os melhores resultados do gráfico anterior; como o desvio padrão é alto, o número de gerações deve ser significativamente acima da média para a obtenção de resultado confiáveis. Nota-se também que, quando a probabilidade de mutação é zero, o método rapidamente converge para uma solução ruim, o que mostra a importância deste operador.

Podemos concluir que os melhores resultados encontrados, bem como os intervalos de probabilidades mais vantajosos são:

	p_m	<i>Intervalo</i>
penalidade normalizada média	0,025	0,010 a 0,060
penalidade total normalizada	0,020	0,020 a 0,060
número de problemas com baixa penalidade normalizada	0,020	0,005 a 0,040
número de gerações, dentre as menores soluções	0,015	0,010 a 0,040
probabilidade média e intervalo principal	0,020	0,020 a 0,040

Tabela 3.6: Melhores resultados para probabilidade de mutação

3.6 Tamanho da população

O resultado geral para o tamanho da população encontra-se no Apêndice A.

Figura 3.11: Penalidade normalizada média *versus* tamanho da população

Figura 3.12: Penalidade total normalizada *versus* tamanho da população

Observa-se em ambos os gráficos que a penalidade decresce com o aumento do tamanho da população: populações acima de 75 indivíduos possuem melhores resultados ($\bar{\gamma}_n < 0,32$ e $\gamma_{tn} < 0,20$). Também possuem uma tendência semelhante, o que indica um comportamento homogêneo nos diferentes problemas.

Os resultados dos outros índices encontram-se a seguir.

Figura 3.13: Número de problemas com baixas penalidades *versus* tamanho da população

Figura 3.14: Número médio de gerações da melhor solução *versus* tamanho da população

Os gráficos acima confirmam a conclusão dos outros dois anteriores: populações com mais de 100 indivíduos possuem melhores resultados; o primeiro indica também que a faixa dentre 275 e 475 indivíduos possui um desempenho ainda um pouco superior.

Podemos concluir que os melhores resultados encontrados e os intervalos de tamanho mais vantajosos são:

	<i>N</i>	<i>Intervalo</i>
penalidade normalizada média	200	100-300
penalidade total normalizada	300	125-300
número de problemas com baixa penalidade normalizada	275	100-300
número de gerações, dentre as menores soluções	300	100-300
tamanho médio e intervalo principal	275	125 a 300

Tabela 3.7: Melhores resultados para o tamanho da população

3.7 Conclusão

Portanto, para problemas de minimização em caixas, segundo os resultados obtidos, convém que os parâmetros estudados estejam nos seguintes intervalos:

	<i>Valor médio</i>	<i>Principal intervalo</i>
p_c	0,95	0,80 a 1,00
p_m	0,020	0,020 a 0,040
N	275	125 a 300
n^o de gerações	-	> 390

Tabela 3.8: Principais resultados do capítulo

Nos próximos capítulos serão determinadas as faixas com melhores resultados para problemas de otimização com restrições e multi-objetivos.

Problemas com restrições

Um problema geral de otimização com restrições contém, além dos limites das variáveis, restrições lineares e não lineares, de igualdade ou desigualdade.

Os métodos para solucionarem esse tipo de problema se diferenciam pelo modo de tratar as restrições. Os modelos tradicionais de otimização procuram transformar problemas restritos em problemas com restrições de caixa, através da eliminação das restrições por penalizações (quadráticas, logarítmica, etc.). Então, a partir de uma única solução inicial, esses algoritmos, em princípio, devem caminhar em direção a um ótimo local.

Uma vez que os Algoritmos Genéticos não trabalham com uma única solução a cada iteração, mas com conjuntos de indivíduos, os métodos tradicionais precisam ser modificados. Outros algoritmos específicos, apoiados nas características dos Algoritmos Genéticos, foram propostos nos últimos anos. Michalewicz e Schoenauer [19] e Coello Coello [2] apresentaram artigos com os principais métodos conhecidos e uma análise crítica de cada um. Nesses trabalhos, os métodos foram agrupados em quatro categorias:

- que preservam a factibilidade das soluções;
- baseados em penalizações;
- outras maneiras de abordar as soluções;
- híbridos.

A seguir encontra-se uma breve descrição de cada categoria.

4.1 Métodos que preservam a factibilidade das soluções

São métodos que, a partir da eliminação de restrições ou de operadores específicos, geram apenas soluções factíveis. Ou seja, a partir de uma população inicial, onde todos os indivíduos são factíveis, as futuras gerações também serão factíveis. Geralmente, utilizam-se variáveis reais, que possibilitam o desenvolvimento dos operadores específicos de mutação ou recombinação para cada problema.

4.1.1 Genocop

O *Genocop* (*GE*netical algorithm for *N*umerical *O*ptimization of *C*onstrained Problems), desenvolvido por Michalewicz e Janikow (1991), trabalha com essa estratégia em problemas com restrições lineares. As restrições de igualdade são utilizadas para eliminar algumas variáveis e formar um problema equivalente, apenas com restrições de desigualdade.

O Genocop trabalha com variáveis reais e, como o problema equivalente possui uma região factível convexa, os operadores garantem que pais factíveis gerem apenas filhos factíveis. Geralmente a recombinação dá-se através da combinação convexa.

O método apresenta principalmente duas desvantagens: trabalha apenas com restrições lineares e necessita de uma população inicial factível, o que pode não ser fácil de obter dependendo das restrições de desigualdade.

4.1.2 *Searching the boundary of feasible region* (Busca da fronteira da região factível)

Outro método que preserva a factibilidade das soluções é a Busca da fronteira da região factível, que possui o objetivo de detectar com precisão os limites da região factível. Como alguns problemas possuem soluções na fronteira, esse método pode ajudar a explorar a região factível com maior eficiência.

A partir de uma população inicial factível e com variáveis reais, esse método trabalha com operadores de mutação e recombinação que são determinados a partir das restrições, de modo que garantam que a nova população pertença à região factível. Uma aplicação imediata desse método é para problemas com apenas uma restrição: com algumas modificações, essa restrição pode ser convertida em um operador de mutação ou de recombinação.

Apesar de eficiente, esse método dificilmente pode ser generalizado. Geralmente desenvolvem-se operadores complexos e específicos para um único problema.

4.2 Métodos baseados em penalizações

Os métodos mais comuns para problemas de otimização com restrições são os baseados em penalizações:

- as restrições são transformadas em uma função penalidade que vale zero para uma solução factível e é positiva para uma infactível.
- o valor da penalidade é proporcional à distância da solução à fronteira da região factível ou equivalente a quanto foram violada as restrições.

Assim, um problema com restrições transforma-se em um problema de minimização em caixas:

$$\hat{f}(\vec{x}) = f(\vec{x}) + \phi(\vec{x})$$

sujeito a:

$$l \leq x \leq u$$

onde:

ϕ é a penalidade no ponto x

Geralmente a penalidade é obtida diretamente a partir das restrições, como mostra o exemplo abaixo:

$$\phi(\vec{x}) = \sum_1^m a_i \cdot H_i(\vec{x}) + \sum_1^p b_i \cdot G_i(\vec{x})$$

onde:

$$H_i(\vec{x}) = |h_i(\vec{x})|^\alpha, 1 \leq i \leq m$$

$$G_i(\vec{x}) = \max[0; g_i(\vec{x})]^\beta, 1 \leq i \leq p$$

a_i e b_i são constantes (geralmente positivas)

α e β normalmente valem 1 ou 2

Os métodos propostos na última década resumem-se em algumas variações do exemplo anterior.

4.2.1 Método de penalidades estáticas

Proposto por Homaifar, Lai, e Qi (1994), baseia-se no princípio de que cada restrição possui diversos níveis de violação. Assim, determinam-se quantos (k) e quais são os níveis de cada uma das restrições (i), associando-se um coeficiente de penalidade R_{ki} . O método trabalha somente com restrições de desigualdade; todas as restrições de igualdade são utilizadas para eliminar variáveis. A partir de uma população inicial aleatória, calcula-se o valor de cada indivíduo pela expressão:

$$\hat{f}(\vec{x}) = f(\vec{x}) + \sum_{i=1}^p R_{ki} \cdot \max[0; g_i(\vec{x})]^2$$

Apesar do grande número de parâmetros necessários, parece que o método apresenta bons resultados quando os coeficientes de penalidade são ajustados com precisão; porém, para fixar os níveis de violação, é necessário um conhecimento prévio do problema.

Kuri (1998) sugeriu um modelo de penalidades estáticas mais simples, onde a penalidade de uma solução é proporcional ao número de restrições violadas:

$$\hat{f}(\vec{x}) = \begin{cases} f(\vec{x}) & \text{se } x \text{ é factível} \\ K - s \cdot \frac{K}{m+p} & \text{caso contrário} \end{cases}$$

onde:

s é o número de restrições não violadas

K é uma constante

m é o número de restrições de igualdade

p é o número de restrições de desigualdade

Note que as soluções que possuem o mesmo número de restrições violadas, sejam de igualdade ou desigualdade, possuem a mesma avaliação. Isso acaba dificultando o processo evolutivo do algoritmo, pois, não é avaliado o grau de violação de cada restrição e o método pode, em alguns casos, não distinguir soluções que estão próximas (melhores) das distantes (piores) da região factível.

4.2.2 Método de penalidades dinâmicas

Proposto por Joines e Houck (1994). Os coeficientes de penalidade crescem com o número de gerações:

$$\hat{f}(\vec{x}) = f(\vec{x}) + (A \cdot t)^\alpha \cdot \phi(\vec{x})$$

onde:

$$\phi(\vec{x}) = \sum_1^m |h_i(\vec{x})| + \sum_1^p \max[0; g_i(\vec{x})]^\beta$$

onde:

A, α e β são constantes

t é a geração do indivíduo $\vec{x}$

Os autores sugerem que as constantes α e β sejam 2 e $A, 0, 5$. Esse método possui menos parâmetros que o anterior, porém os coeficientes de penalidade podem crescer muito com o aumento do número de gerações (t), dificultando a busca pelos ótimos locais. Para funções quadráticas, o método possui um bom desempenho.

4.2.3 Genocop II

O *Genetic algorithm for Numerical Optimization of COstrained Problems II* foi proposto por Michalewicz e Attia (1994). Trata-se de uma evolução do Genocop (ou Genocop I) inspirada no *Simulated Annealing*. As penalidades aplicadas crescem com o número de gerações, através da redução da “temperatura” do sistema:

$$\hat{f}(\vec{x}, \tau) = f(\vec{x}) + \frac{1}{2\tau} \phi(\vec{x})$$

onde:

$$\phi(\vec{x}) = \sum_1^m |h_i(\vec{x})|^2 + \sum_1^p \max[0; g_i(\vec{x})]^2$$

τ é a temperatura do sistema

A população inicial consiste em inúmeras cópias de uma única solução que satisfaz as restrições lineares. O decréscimo da temperatura é feito paulatinamente, a cada geração, a partir de um valor inicial τ_0 até um final τ_f , quando a busca termina.

Os autores sugerem o seguinte resfriamento:

$$\begin{aligned}\tau_0 &= 1 \\ \tau_{i+1} &= 0,1\tau_i \\ \tau_f &= 0,000001\end{aligned}$$

A técnica proposta por Michalewicz e Attia é eficiente, no entanto exige uma fase inicial onde geram-se pontos que satisfazem todas as restrições lineares (de igualdade ou desigualdade), como no Genocop I, e também operadores especiais que garantam que as novas gerações satisfaçam essas restrições. No entanto, o método é muito sensível aos valores e à velocidade de resfriamento do sistema.

4.2.4 Método de penalidades adaptativas

Proposto por Bean e Hadja-Alouane (1992), onde a penalidade dos indivíduos de uma geração é influenciada pelos melhores das últimas gerações:

$$\hat{f}(\vec{x}) = f(\vec{x}) + \lambda(t) \left(\sum_1^m |h_i(\vec{x})| + \sum_1^p \max[0; g_i(\vec{x})]^2 \right)$$

onde $\lambda(t)$ é atualizado a cada geração t :

$$\lambda(t+1) = \begin{cases} \frac{1}{\beta_1} \lambda(t) & \text{se } \vec{b}^i \in \Omega \forall (t-k+1) \leq i \leq t \\ \beta_2 \lambda(t) & \text{se } \vec{b}^i \notin \Omega \forall (t-k+1) \leq i \leq t \\ \lambda(t) & \text{caso contrário} \end{cases}$$

onde:

$\vec{b}^i$ é o melhor indivíduo da geração i

β_1 e β_2 são constantes, tais que $\beta_1, \beta_2 > 1$ e $\beta_1 \neq \beta_2$

k é o número de gerações avaliadas

Deste modo, o método reduz o coeficiente de penalidade quando os melhores indivíduos das últimas k gerações são todos factíveis e o aumenta, quando são infactíveis. Se nas últimas k gerações houve melhores indivíduos factíveis e infactíveis, o coeficiente $\lambda(t)$ não é alterado.

Para implementar esse método, as principais dificuldades são estipular o número de gerações avaliadas k e os coeficientes β_1 e β_2 , fatores que claramente influenciam os resultados.

4.2.5 Pena de morte

A maneira mais fácil de lidar com as soluções infactíveis é eliminá-las, ou seja, todas possuem aptidão zero. Os experimentos realizados por Michalewicz (1995) indicam que esse método possui uma eficiência baixa e o desvio padrão das soluções é relativamente alto. Coit e Smith (1996) concluíram que os outros métodos de penalidade possuem um melhor desempenho. De fato, sabe-se que as soluções infactíveis possuem informações que, combinadas com outras soluções, podem levar a boas regiões factíveis (são importantes caminhos para o algoritmo); ou seja, a perda dessas informações empobrece a busca.

4.3 Outras maneiras de abordar as restrições

Os métodos a seguir baseiam-se em diferentes tratamentos das restrições. Alguns consideram cada restrição de uma vez, outros acrescentam distintas penalizações ou distinguem as restrições em lineares e não lineares.

4.3.1 Memória de comportamento

O *behavioral memory method*, proposto por Schoenauer e Xanthakis (1993), considera uma restrição por vez. Quando uma porcentagem da população é factível para essa restrição, passa-se a considerar a seguinte. O pseudocódigo do método encontra-se a seguir.

⇒ Gere uma população aleatória
 ⇒ Seja $j = 1$
 ⇒ Evolua a população, até que uma porcentagem ϕ seja factível para a primeira restrição

$$\hat{f}(\vec{x}) = M - g_1(\vec{x})$$
 onde:
 M é suficientemente grande
 ⇒ $j = j + 1$
 ⇒ Evolua a população, até que haja uma proporção ϕ de indivíduos factíveis à j -ésima restrição. Os pontos que não satisfizerem às restrições anteriores $(g_1(\vec{x}), \dots, g_{j-1}(\vec{x}))$, devem ser eliminados.

$$\hat{f}(\vec{x}) = M - g_j(\vec{x})$$
 onde:
 M deve ser ajustado a cada geração
 ⇒ Se $j < m$, repita os dois passos anteriores.
 Caso contrário, evolua a população atual otimizando diretamente a função $f(\vec{x})$ e rejeitando os indivíduos infactíveis

Alguns fatores influenciam este método, como a ordem escolhida para as restrições e a porcentagem ϕ . Observe que a penalidade de morte é utilizada diversas vezes no processo, pois, o método guarda na memória todas as restrições que já foram analisadas.

4.3.2 Método da superioridade dos pontos factíveis

Proposto por Powel e Skolnick (1993), baseia-se nos métodos tradicionais de penalizações, onde os indivíduos são avaliados por:

$$\hat{f}(\vec{x}) = \begin{cases} f(\vec{x}) & \text{se } \vec{x} \in \Omega \\ f(\vec{x}) + r \cdot (\sum_1^m |h_i(\vec{x})| + \sum_1^p \max[0; g_i(\vec{x})]) & \text{se } \Omega = \emptyset \\ f(\vec{x}) + r \cdot (\sum_1^m |h_i(\vec{x})| + \sum_1^p \max[0; g_i(\vec{x})]) + \Theta(t, \vec{x}) & \text{caso contrário} \end{cases}$$

onde:

$\Theta(t, \vec{x})$ é um termo adicional para os indivíduos inactivos, quando $\Omega \neq \emptyset$:

$$\Theta(t, \vec{x}) = \max\{0; \max_{\vec{x} \in \Omega} f(\vec{x}) - \min_{\vec{x} \in \mathbb{R}^n - \Omega} f(\vec{x}) + r \cdot (\sum_1^m |h_i(\vec{x})| + \sum_1^p \max[0; g_i(\vec{x})])\}$$

r é uma constante

t é a geração do indivíduo $\vec{x}$

Em outras palavras, o termo $\Theta(t, \vec{x})$ garante que todos os indivíduos activos da geração t possuam melhor ou igual avaliação do que os inactivos, ou seja, $\hat{f}(\vec{x}) \leq f(\vec{y}) \forall \vec{x} \in \Omega$ e $\vec{y} \notin \Omega$.

Para problemas em que a região activa representa apenas uma pequena percentagem das restrições de caixa, o método pode apresentar alguma dificuldade para encontrar soluções activas.

4.3.3 Método de penalidade livre de parâmetros

Proposto por Deb K. (2000). Baseado no método anterior, onde a avaliação de um ponto inactivo não depende do valor da função objetivo neste ponto:

$$\hat{f}(\vec{x}) = f(\vec{x}) + \sum_1^m |h_i(\vec{x})| + \sum_1^p \max[0; g_i(\vec{x})] + \Theta(t, \vec{x})$$

onde:

$$\Theta(t, \vec{x}) = \begin{cases} 0 & \text{se } \vec{x} \in \Omega \\ -f(\vec{x}) & \text{se } \nexists \vec{x} \text{ tal que } \vec{x} \in \Omega \\ -f(\vec{x}) + \max_{\vec{x} \in \Omega} \{f(\vec{x})\} & \text{caso contrário} \end{cases}$$

Note que o método é semelhante ao anterior, porém os termos referentes às restrições não possuem o coeficiente de penalidade r . O termo $\Theta(t, \vec{x})$ continua garantindo que, em uma geração t com indivíduos activos e inactivos, todos os activos serão melhores avaliados que os inactivos. A aptidão de um inactivo não dependerá da função objetivo neste ponto, mas será influenciada pelo pior indivíduo activo.

4.3.4 Genocop III

O *Genetic algorithm for Numerical Optimization of COnstrained Problems III* foi proposto por Michalewicz (1996). Trata-se de uma evolução do *Genocop I*. O método trabalha com restrições lineares e não lineares, onde duas populações separadas se desenvolvem e a avaliação dos indivíduos de uma afeta a aptidão dos da outra.

A primeira população P_S consiste em pontos de busca S que satisfazem apenas as restrições lineares: com as restrições de igualdade, eliminam-se algumas variáveis do problema e, com as de desigualdade, são determinados os operadores de mutação e recombinação, de modo que todos os indivíduos satisfaçam as restrições lineares. Observe que essa parte do método é idêntica ao *Genocop I*. Seja F o conjunto de pontos que satisfazem as restrições lineares e Ω o conjunto factível; então $\Omega \subset F$.

A segunda população P_R consiste em pontos de referência R que satisfazem todas as restrições. Os pontos de referência $\vec{r}$, por serem factíveis, são avaliados diretamente: $\hat{f}(\vec{r}) = f(\vec{r})$.

Os pontos de busca $\vec{s}$, se forem factíveis ($\vec{s} \in \Omega$), também serão avaliados diretamente: $\hat{f}(\vec{s}) = f(\vec{s})$. Caso contrário ($\vec{s} \in F - \Omega$), é selecionado, pelo método da roleta, um ponto de referência $\vec{r}$; pontos aleatórios entre $\vec{r}$ e $\vec{s}$ são gerados, a partir de uma sequência aleatória de números reais a entre zero e um: $\vec{z} = a\vec{s} + (1 - a)\vec{r}$, $a \in (0; 1)$. O primeiro $\vec{z}$ factível ($\vec{z} \in \Omega$) determina a avaliação do ponto $\vec{s}$: $\hat{f}(\vec{s}) = \hat{f}(\vec{z}) = f(\vec{z})$. O ponto $\vec{z}$ pode substituir o ponto de referência $\vec{r}$ caso tenha uma melhor avaliação ($\hat{f}(\vec{z}) < \hat{f}(\vec{r})$) ou, caso contrário, com uma probabilidade p_r , pode substituir $\vec{s}$.

O Genocop III apresenta algumas vantagens com relação a outros métodos:

- todos os indivíduos são avaliados diretamente pela função objetivo, evitando possíveis influências de penalizações;
- possui poucos parâmetros;
- possui sempre uma população factível;
- as vizinhanças dos melhores pontos de referência são exploradas com maior frequência.

No entanto, o *Genocop III* inicia o método com uma população factível e requer operadores especiais para evoluir os pontos de busca.

4.4 Métodos híbridos

São métodos que combinam o Algoritmo Genético com outras técnicas de otimização, como, por exemplo, o método de Lagrange, heurísticas ou metaheurísticas (*scatter search*, busca tabu, *ant colony*, etc.). A seguir encontram-se alguns exemplos dessas técnicas.

4.4.1 Multiplicadores de Lagrange

Adeli e Cheng (1994) propuseram um Algoritmo Genético baseado na sequência de minimização de Lagrange:

$$\hat{f}(\vec{x}) = f(\vec{x}) + \frac{1}{2} \sum_1^p \gamma_i \{\max[0; g_i(\vec{x}) + \mu_i]\}^2$$

onde:

γ_i e μ_i são parâmetros associados à i -ésima restrição

O termo μ_i é definido a partir da máxima violação da restrição i nas gerações anteriores.

O coeficiente γ_i deve crescer ao longo da busca. Os autores sugerem um parâmetro $\beta > 1$, que multiplica γ_i a cada geração.

Kim e Myung (1998) propuseram um algoritmo combinando com o método lagrangeano aumentado. Durante a primeira fase do algoritmo, as soluções são avaliadas pela função:

$$\hat{f}(\vec{x}) = f(\vec{x}) + \frac{C}{2} \left(\sum_1^m |h_i(\vec{x})|^2 + \sum_1^p \max[0; g_i(\vec{x})]^2 \right)$$

onde:

C é uma constante

Essa fase tem o objetivo de gerar soluções factíveis: quando há um número suficiente, o procedimento proposto por Adeli e Chang passa a ser usado.

O número de parâmetros e os respectivos ajustes são as principais desvantagens desses métodos.

4.4.2 Otimização aleatória

O *Constrained optimization by random evolution* (CORE) foi proposto por Belur (1997). Trata-se de uma técnica híbrida combinando um algoritmo evolutivo com outro qualquer de otimização sem restrições.

Quando o algoritmo evolutivo possui uma solução infactível, a seguinte função é minimizada:

$$C(\vec{x}) = \sum_{i \in C_1} h_i^2(\vec{x}) + \sum_{j \in C_2} g_j(\vec{x})$$

onde:

$$C_1 = \{i = 1, \dots, m / |h_i(\vec{x})| > \varepsilon_C\}$$

$$C_2 = \{j = 1, \dots, q / g_j(\vec{x}) > 0\}$$

ε_C é a tolerância para as restrições de igualdade $h_i(\vec{x})$

A menor solução encontrada substitui a infactível.

Para alguns problemas esse método pode não ser vantajoso, pois, reparar soluções infactíveis nem sempre é um processo simples e o custo computacional do método pode ser alto.

4.4.3 Sistema imune

Forrest e Perelson (1991) exploraram o modelo de imunidade, onde uma população é evoluída através de um conjunto de antígenos, formando anticorpos (melhores soluções).

Smith et al. (1992) propuseram um modo de manter a diversidade em funções multimodais e Hajela e Lee (1995) estenderam esse modelo para funções com restrições, cujo pseudocódigo encontra-se a seguir.

- ⇒ Gere uma população aleatória; calcule para cada solução o valor da função objetivo e a soma das restrições violadas.
- ⇒ Separe as soluções factíveis das infactíveis. Ordene cada grupo a partir do valor da função objetivo. Calcule o valor médio da avaliação das soluções factíveis.
- ⇒ Escolha um número de soluções factíveis que possuem aptidão próxima ao valor médio calculado no passo anterior e as separe em uma nova população. Elas serão chamadas de antígenos.
- ⇒ As soluções infactíveis serão submetidas à simulação de um sistema de imunidade, gerando anticorpos com relação aos antígenos do passo anterior. Essa simulação produz uma subpopulação com menores violações das restrições.
- ⇒ Forme uma população com os indivíduos factíveis e os melhores da subpopulação do passo anterior. Aplique o Algoritmo Genético com a aptidão igual à função objetivo.

A simulação de um sistema de imunidade tenta transformar as soluções infactíveis através de algumas factíveis (antígenos). Há inúmeras maneiras de realizar esse processo. Pode-se escolher todos os indivíduos infactíveis ou apenas alguns, próximos à aptidão média dos infactíveis, por exemplo. A maneira mais simples de transformá-los em anticorpos é realizar uma troca de genes (recombinação) com os antígenos.

As principais dificuldades do método são gerar indivíduos factíveis na população inicial e o seu desempenho quando a proporção entre factíveis e infactíveis é muito grande.

4.5 Escolha do método e problemas teste

Os métodos anteriores possuem características diferentes e podem ser mais ou menos adaptados para diferentes tipos de problemas. Como será resolvido um conjunto de problemas teste, procuramos selecionar um método que pudesse ser implementado facilmente para diversas funções; ou seja, os métodos em que os operadores (mutação e recombinação) devem ser específicos para cada tipo de problema foram descartados.

Dentre vários métodos possíveis e que parecem eficientes, optou-se pelo *Método de Penalidade Livre de Parâmetros*. Miettinen, Mäkelä e Toivanen (2003) informam que apresenta um bom desempenho; segundo os testes feitos, o método dificilmente não encontra soluções factíveis e geralmente apresenta soluções próximas da melhor conhecida, em um número não muito alto de iterações [21].

Para estudar os problemas restritos, utilizamos 16 funções, que se encontram no Apêndice B. Procuramos selecionar funções com características variadas, como mostra a tabela a seguir:

<i>função</i>	<i>dimensão</i>	<i>função objetivo</i>	ρ	<i>li</i>	<i>ld</i>	<i>ni</i>	<i>nd</i>
1	8	linear	0,000578	-	6	-	-
2	5	quadrática	26,960078	-	-	-	6
3	2	linear	44,200537	-	-	-	2
4	2	quadrática	0,332226	-	-	1	-
5	4	não linear	0,000000	-	2	3	-
6	2	não linear	24,99898	-	2	-	-
7	2	não linear	0,861168	-	-	-	2
8	23	não linear	0,000000	-	-	1	-
9	10	não linear	0,000000	3	-	-	-
10	2	não linear	7,329000	-	-	-	2
11	2	quadrática	96,644521	-	-	-	2
12	7	não linear	0,524944	-	-	-	4
13	2	não linear	0,006711	-	-	-	2
14	2	quadrática	37,492715	-	1	-	1
15	6	quadrática	23,404995	-	2	-	-
16	13	quadrática	0,237391	-	9	-	-

li: restrições lineares de igualdade

ld: restrições lineares de desigualdade

ni: restrições não lineares de igualdade

nd: restrições não lineares de desigualdade

ρ : percentagem factível das restrições de caixa

Tabela 4.1: Conjunto de problemas teste com restrições

O parâmetro ρ é a percentagem estimada do espaço de busca, formado pelas restrições de caixa, que é factível, ou seja, a proporção da região definida pelos limites das variáveis ($l_i \leq x_i \leq u_i, i = 1, \dots, n$) que satisfaz as outras restrições $|h_i(\vec{x})| = 0, i = 1, \dots, m$ e $g_i(\vec{x}) \leq 0, i = 1, \dots, p$. Este número foi calculado por Miettinen, Mäkelä e Toivanen (2003): trata-se da percentagem factível de 100 milhões de pontos gerados aleatoriamente dentro das restrições de caixa [21].

4.6 Probabilidade de recombinação

Todos os problemas testes foram resolvidos nos três cenários (A, B e C — seção 3.1) com a mesma função penalidade apresentada para os problemas de minimização em caixas. Os resultados encontram-se no Apêndice B.

Figura 4.1: Penalidade normalizada média *versus* probabilidade de recombinação

Figura 4.2: Penalidade total normalizada *versus* probabilidade de recombinação

A penalidade total normalizada indica que uma alta probabilidade de recombinação (entre de 0,76 e 1,00) tende a gerar melhores resultados ($\gamma_{tn} < 0,43$). A penalidade normalizada média possui uma “tendência” a diminuir com o aumento da probabilidade de recombinação, o que confirma a conclusão anterior ($\bar{\gamma}_n < 0,15$ para probabilidades de recombinação de 0,8 a 0,98).

Do mesmo modo que os problemas de minimização em caixas, a diferente oscilação dos gráficos provém da diferença conceitual entre as duas medidas. A penalidade normalizada média tende a apresentar uma curva suave (por ser uma média), atribuindo um peso semelhante para cada problema. Já a penalidade total normalizada é influenciada por alguns problemas com alta penalidade bruta (piores soluções), o que intensifica as oscilações do gráfico.

Os resultados dos outros índices encontram-se a seguir.

Figura 4.3: Número de problemas com baixas penalidades *versus* probabilidade de recombinação

Figura 4.4: Número médio de gerações da melhor solução *versus* probabilidade de recombinação

Graficamente, observa-se que o número de problemas com baixa penalidade tende a crescer, possuindo bons resultados para probabilidades acima de 0,84 (acima de 11 problemas). O número médio de gerações, para os melhores resultados, oscila aproximadamente entre 310 e 360 independentemente da probabilidade de recombinação. O desvio padrão encontrado foi alto, indicando a necessidade de um número significativamente acima da média para obter resultados confiáveis.

Podemos concluir que os melhores resultados encontrados, bem como os intervalos de probabilidades mais vantajosos são:

	p_c	<i>Intervalo</i>
penalidade normalizada média	0,92	0,80 a 1,00
penalidade total normalizada	0,82	0,80 a 0,98
número de problemas com baixa penalidade normalizada	0,84	0,84 a 1,00
número de gerações, dentro as menores soluções	0,92	não há
<i>probabilidade média e intervalo principal</i>	<i>0,88</i>	<i>0,84 a 0,98</i>

Tabela 4.2: Melhores resultados para probabilidade de recombinação

4.7 Probabilidade de mutação

O resultado geral para as funções com restrições encontra-se no Apêndice B.

Figura 4.5: Penalidade normalizada média *versus* probabilidade de mutação

Figura 4.6: Penalidade total normalizada *versus* probabilidade de mutação

Graficamente, observa-se que a penalidade normalizada média possui bons resultados com uma probabilidade de mutação entre de 0,005 e 0,050 ($\bar{\gamma}_n < 0,23$) e, a penalidade total normalizada, geralmente entre 0,010 e 0,105 ($\gamma_{tn} < 0,15$). Note que os gráficos possuem tendências semelhantes, porém um é mais suave e o outro, mais acentuado; ambos mostram que uma probabilidade alta ou igual a zero geram resultados ruins. Conclui-se que a mutação ajuda na busca pela melhor solução, porém, com altas probabilidades, a convergência do algoritmo é afetada.

Os gráficos dos outros índices encontram-se a seguir.

Figura 4.7: Número de problemas com baixas penalidades *versus* probabilidade de mutação

Figura 4.8: Número médio de gerações da melhor solução *versus* probabilidade de mutação

Graficamente, observa-se que o número de problemas com baixas penalidades possui bons resultados para probabilidades de mutação entre 0,005 e 0,050 (acima de 25 problemas). O número médio de gerações oscila entre 200 e 250 para os melhores resultados. Novamente o desvio padrão encontrado foi alto, indicando a conveniência de um número significativamente acima da média para obter resultados confiáveis. Quando a probabilidade de mutação é zero, o método rapidamente converge para uma solução ruim, o que mostra a importância deste operador.

Pode-se concluir que os melhores resultados encontrados, bem como os intervalos de probabilidades mais vantajosos são:

	p_m	<i>Intervalo</i>
penalidade normalizada média	0,045	0,005 a 0,075
penalidade total normalizada	0,065	0,010 a 0,105
número de problemas com baixa penalidade normalizada	0,020	0,005 a 0,050
número de gerações, dentro as menores soluções	0,020	> 0,005
<i>probabilidade média e intervalo principal</i>	<i>0,038</i>	<i>0,010 a 0,050</i>

Tabela 4.3: Melhores resultados para probabilidade de mutação

4.8 Tamanho da população

O resultado geral para determinar o melhor tamanho de população encontra-se no Apêndice B.

Figura 4.9: Penalidade normalizada média *versus* tamanho da população

Figura 4.10: Penalidade total normalizada *versus* tamanho da população

A penalidade normalizada média indica, graficamente, que populações de 100 a 525 indivíduos possuem boas soluções ($\bar{\gamma}_n < 0,34$).

Já a penalidade total normalizada possui bom desempenho com populações entre 100 e 225 ($\bar{\gamma}_n < 0,31$).

Os resultados dos outros índices encontram-se a seguir.

Figura 4.11: Número de problemas com baixas penalidades *versus* tamanho da população

Figura 4.12: Número médio de gerações da melhor solução *versus* tamanho da população

O primeiro gráfico indica que populações entre 125 e 275 indivíduos possuem bons resultados (acima de 17 problemas). O número de gerações necessárias ficou entre 240 e 340 para populações acima de 100 indivíduos.

Pode-se concluir que os melhores resultados encontrados, bem como os intervalos de tamanhos mais vantajosos são:

	<i>N</i>	<i>Intervalo</i>
penalidade normalizada média	225	100-300
penalidade total normalizada	275	100-275
número de problemas com baixa penalidade normalizada	150	100-300
número de gerações, dentro as menores soluções	200	100-300
<i>tamanho médio e intervalo principal</i>	<i>210</i>	<i>100 a 275</i>

Tabela 4.4: Melhores resultados para o tamanho da população

4.9 Conclusão

Para problemas de otimização com restrições, podemos concluir que os parâmetros os principais intervalos para os parâmetros estudados são:

	<i>Valor médio</i>	<i>Principal intervalo</i>
p_c	0,88	0,84 a 0,98
p_m	0,038	0,010 a 0,050
N	210	100 a 275
n^o de gerações	-	> 360

Tabela 4.5: Principais resultados do capítulo

Pode-se observar que os resultados são muito semelhantes aos dos problemas de minimização em caixas.

Problemas Multi-objetivos

Todos os problemas estudados nos capítulos anteriores apresentavam apenas um critério a ser otimizado, pelo qual, através da função objetivo e das restrições, era medida a aptidão de cada indivíduo da população. No entanto, no mundo real, muitos problemas são marcados pela presença de vários objetivos conflitantes e seria um empobrecimento reduzi-los a uma única função a ser otimizada.

Normalmente a decisão sobre qual é o melhor resultado dentre vários objetivos é o resultados de uma comparação de várias boas soluções. Os algoritmos evolutivos, por trabalharem com populações de soluções, retratam de um modo apropriado a natureza deste tipo de busca, pois geram um conjunto de boas soluções em um único processo. Já as outras técnicas tradicionais exigem vários processos para encontrar um número suficiente de bons resultados.

Algumas aplicações recentes desse tipo de otimização foram estudos de coeficientes de arrasto em projetos de asas de avião [22] [23] [25], problemas de seqüenciamento de produção [16] [24], estudo de projetos de estruturas [1] e problemas de logística [26].

Para comparar os diferentes objetivos, alguns algoritmos trabalham com a noção de utilidade ou valor de cada solução. Trata-se de uma analogia ao processo intuitivo de avaliação do ser humano, que geralmente atribui graus de importâncias para as diferentes características ou peso para cada um dos objetivos. Os melhores projetos são aqueles que possuem um maior valor ou utilidade.

Zitzler, Deb e Thiele [28], Zitzler, Laumanns e Bleuler [29] e Coello Coello [3] realizaram estudos comparativos e análises críticas dos principais métodos conhecidos. Nesses trabalhos, os métodos foram classificados em três categorias principais:

- métodos que agregam os objetivos em uma única função;
- outros métodos não baseados no conceito de Pareto;

- métodos que utilizam o conjunto ótimo de Pareto.

5.1 Métodos que agregam os objetivos em uma única função

A abordagem mais simples para otimização multi-objetivos tenta unir os diversos objetivos em uma única função. Como o Algoritmo Genético básico trabalha com uma única função objetivo, a abordagem mais imediata sugere uma combinação dos objetivos (através de somas, multiplicações ou outras operações) para calcular diretamente a aptidão de cada uma das soluções.

Em princípio, esses métodos perdem a informação de cada um dos objetivos e, eventualmente, priorizam alguns em detrimento de outros, nem sempre gerando os melhores resultados. Também é importante ressaltar que em diversos casos, a priori, não se pode ponderar os objetivos e deseja-se obter informações de cada objetivo e de cada solução para se tomar decisões.

5.1.1 Agregação de objetivos por pesos variáveis

Proposto por Hajela e Lin (1992), onde a função objetivo é determinada pela soma ponderada entre os k objetivos:

$$\hat{f}(\vec{x}) = \sum_{i=1}^k \alpha_i \cdot f_i(\vec{x})$$

onde:

$$\sum_{i=1}^k \alpha_i = 1$$

$$\alpha_i > 0$$

Como os resultados variam de acordo com os coeficientes α_i , pode-se resolver um problema diversas vezes, utilizando diferentes conjunto de coeficientes, e analisar cada um dos resultados gerados.

Outra possibilidade de evitar que alguns objetivos sejam privilegiados é garantir que todos tenham a mesma ordem de grandeza. Assim, a seguinte função seria minimizada:

$$\hat{f}(\vec{x}) = \sum_1^k \alpha_i f_i(\vec{x}) \cdot c_i$$

onde:

c_i é uma constante proporcional a cada objetivo i

A principal desvantagem do método é determinar coeficientes apropriados para cada problema. Por outro lado, a *Agregação de objetivos por pesos variáveis* é, do ponto de vista computacional, muito eficiente e pode ser utilizada para gerar uma população inicial para outros métodos.

5.1.2 Goal programming

Trata-se de uma técnica antiga aplicada ao Algoritmo Genético, desenvolvida por Charnes e Cooper (1961) e Ijiri (1965), onde se estipula uma meta para cada objetivo. Esses valores são incorporados à função objetivo, que minimiza a diferença absoluta entre o valor de cada objetivo e a sua meta:

$$\hat{f}(\vec{x}) = \sum_1^k |f_i(\vec{x}) - T_i|$$

onde:

T_i é a meta de cada objetivo i

Quando as metas estipuladas são próximas às melhores soluções para um problema, o método é eficiente. No entanto, quando não há um conhecimento prévio do problema, as metas podem influir priorizando alguns objetivos. Dependendo da região factível, o método pode ser ineficiente.

5.1.3 Goal attainment

Nesse método se estipulam dois vetores w_i e b_i , com metas que cada objetivo f_i deve alcançar. Para cada solução x , o método minimiza o seguinte problema:

$$\begin{aligned}
& \min \alpha \\
& \text{sujeito a:} \\
& g_j(\vec{x}) \leq 0, j = 1, \dots, p \\
& b_i + \alpha \cdot w_i \geq f_i(\vec{x}), i = 1, \dots, k \\
& \sum_{i=1}^k |w_i| = 1 \\
& \text{onde:} \\
& \alpha \text{ é uma variável irrestrita}
\end{aligned}$$

Quando o método gera um valor negativo para α , então a meta estipulada foi alcançada e um resultado melhor foi encontrado. Caso contrário, a solução x não atingiu a meta proposta.

Entre as desvantagens do método, pode-se ressaltar o número de parâmetros e a dificuldade, em alguns casos, de distinguir duas soluções diferentes que produzem o mesmo resultado (α). Por outro lado, computacionalmente, o método é simples

5.1.4 The ε -Constraint Method

Prioriza a otimização de um dos objetivos, considerando os demais como restrições do problema:

$$\begin{aligned}
& \hat{f}(\vec{x}) = \min f_r(\vec{x}) \\
& \text{sujeito a:} \\
& f_i(\vec{x}) \leq \varepsilon_i, i = 1, \dots, k \text{ e } i \neq r
\end{aligned}$$

Esse processo pode ser repetido para diversos ε_i até se encontrar um valor satisfatório para cada restrição. Após se calcular a solução para cada objetivo ($r = 1, \dots, k$), pode-se comparar os resultados com os limites ε_i encontrados. Apesar da simplicidade, o ponto fraco do *The ε -Constraint Method* é o tempo de processamento, principalmente quando há muitos objetivos.

5.2 Outros métodos não baseados no conceito de Pareto

Dentre os vários métodos existentes, serão apresentados os dois principais.

5.2.1 Vector Evaluated Genetic Algorithms (*VEGA*)

Método proposto por Schaffer (1984), o *VEGA* baseia-se em uma adaptação do software GÊNESIS (utilizado nos capítulos anteriores e desenvolvido por Grefenstette [14]), onde cada objetivo é utilizado para selecionar uma parte da população que irá reproduzir. Quando todos os objetivos já escolheram os seus reprodutores, estes são sorteados aleatoriamente para os processos de mutação e recombinação.

A principal dificuldade do *VEGA* é efetuar a busca em algumas regiões factíveis: por utilizar cada um dos objetivos diretamente, o método algumas vezes seleciona indivíduos bons para uns objetivos em detrimento de outros, ou seja, nem sempre otimiza o conjunto dos objetivos.

Schaffer efetuou algumas tentativas para melhorar o seu método como, por exemplo, substituir o sorteio aleatório por uma roleta, onde os melhores indivíduos (utilizando os conceitos do conjunto ótimo de Pareto, explicado nas próximas páginas) possuem maior probabilidade de serem escolhidos. Porém, quando o problema possui poucas soluções boas, o método continua apresentando a mesma deficiência.

5.2.2 Ordem Lexicográfica

Nesse método os objetivos são ordenados pelo programador por ordem de importância. A solução ótima é obtida otimizando as funções objetivo, começando pela mais importante ($f_1(\vec{x})$), até a última função ($f_k(\vec{x})$).

Portanto, o primeiro problema formulado será:

$$\begin{aligned} &\min f_1(\vec{x}) \\ &\text{sujeito a:} \\ &g_j(\vec{x}) \leq 0, j = 1, \dots, p \\ &u \leq x \leq l \end{aligned}$$

Seja x_1^* e $f_1^* = f_1(x_1^*)$ a solução obtida. Então o segundo problema será formulado como:

$$\begin{aligned} &\min f_2(\vec{x}) \\ &\text{sujeito a:} \\ &f_1(\vec{x}) = f_1^* \\ &g_j(\vec{x}) \leq 0, j = 1, \dots, p \\ &u \leq x \leq l \end{aligned}$$

Seja x_2^* e $f_2^* = f_2(x_2^*)$ a solução obtida. Esse procedimento será repetido para os k objetivos:

$$\begin{aligned} &\min f_i(\vec{x}) \\ &\text{sujeito a:} \\ &f_l(\vec{x}) = f_l^*, l = 1, \dots, i - 1 \\ &g_j(\vec{x}) \leq 0, j = 1, \dots, p \\ &u \leq x \leq l \end{aligned}$$

A solução obtida no fim, x_k^* será a solução do problema.

Apesar de ter a simplicidade da *agregação de objetivos por pesos variáveis* e o *VEGA*, a *ordem lexicográfica* sempre privilegiará os primeiros objetivos em detrimento dos outros. Escolher aleatoriamente os objetivos equivale a fazer uma soma ponderada por pesos.

5.3 Métodos que utilizam o conjunto ótimo de Pareto

A seguir, apresentaremos três definições necessárias para o conceito do conjunto ótimo de Pareto.

DEFINIÇÃO 1: Considere, sem perder a generalidade, um problema de minimização multi-objetivos com n variáveis de decisão e k objetivos:

$$\begin{aligned}
& \min f(\vec{x}) = (f_1(\vec{x}), f_2(\vec{x}), \dots, f_k(\vec{x})), \vec{x} \in \mathbf{R}^n \\
& \text{sujeito a:} \\
& h_i(\vec{x}) = 0, i = 1, \dots, m \\
& g_j(\vec{x}) \leq 0, j = 1, \dots, p \\
& l \leq x \leq u
\end{aligned}$$

onde x é chamado vetor de decisão. Dizemos que um vetor de decisão a *domina* um vetor de decisão b se, e somente se,

$$\begin{aligned}
& \forall i \in 1, \dots, k : f_i(a) = f_i(b) \\
& \exists j \in 1, \dots, k : f_j(a) < f_j(b)
\end{aligned}$$

O que será representado por:

$$a \prec b$$

DEFINIÇÃO 2: Seja $\vec{a}$ um vetor de decisão qualquer:

1. O vetor de decisão a é dito não dominado em um conjunto X se, e somente se, não há vetor de decisão em X que domina a , ou seja:

$$\nexists a' \in X : a' \prec a$$

2. O vetor de decisão a pertence ao *conjunto ótimo de Pareto* se, e somente se, a é não dominado no conjunto X .

Convém também definir os conceitos de ótimo local e global, analogamente às definições para problemas com um único objetivo.

DEFINIÇÃO 3: Considere um conjunto de vetores de decisão $X' \subseteq X$.

1. O conjunto X' é denominado um *conjunto ótimo local de Pareto* se, e somente se,

$$\forall a \in X', \nexists \text{ um vetor de decisão } b : b \prec a \wedge \|b - a\| < \varepsilon \wedge \|f(b) - f(a)\| < \delta$$

onde:

$\|\cdot\|$ é a norma euclidiana

$\varepsilon, \delta > 0$

2. O conjunto X é chamado *conjunto ótimo global de Pareto* se, e somente se,

$$\forall a \in X', \nexists \text{ um vetor de decisão } b \in X : b \prec a$$

Note que um conjunto ótimo global de Pareto, não contém todas as soluções “ótimas” de um problema de otimização com um único objetivo, mas todas as soluções não dominadas. Esse conjunto será chamado de linha frontal do Pareto.

A principal dificuldade dos algoritmos de otimização multi-objetivos consiste em:

- avaliar e selecionar bons indivíduos, ou seja, soluções que levem a encontrar um conjunto ótimo de Pareto;
- manter a diversidade na população, evitando convergências prematuras e buscando linhas frontais do Pareto onde as soluções estejam bem distribuídas.

Diferentes abordagens foram propostas para harmonizar esses dois critérios.

5.3.1 Multi-objective Optimization Genetic Algorithm (*MOGA*)

Proposto por Fonseca e Fleming (1993), o *MOGA* avalia todos os indivíduos da população e atribui aos não dominados a qualificação um. Os dominados recebem como qualificação o número de soluções que os dominam mais um.

Após determinar os conjuntos de pontos que possuem a mesma aptidão, o algoritmo seleciona, pelo método da roleta, quais desses conjuntos participarão da reprodução. Para

evitar a convergência prematura, o algoritmo também utiliza os métodos de formação de nicho.

Por ser simples e eficiente, muitas pesquisas foram realizadas com o *MOGA*; no entanto, o algoritmo geralmente possui um desempenho inferior aos outros métodos que utilizam o conceito de Pareto.

5.3.2 Niche Pareto Genetic Algorithm (*NPGA*)

O *NPGA* foi proposto por Horn e Nafpliotis (1993). Um conjunto de t soluções da população é selecionado para a escolha de um reprodutor. Dois são sorteados e comparados com as t soluções: se um deles é não dominado e o outro, dominado, o primeiro é escolhido; caso contrário (ambos são dominados ou não dominados), o que tiver menos indivíduos da população inteira (não somente dentre os t selecionados) dentro de uma vizinhança σ é escolhida.

A principal desvantagem do *NPGA* é a introdução de dois parâmetros, dos quais dependem a eficácia do método; por outro lado, geralmente é um método rápido.

5.3.3 Nondominated Sorting Genetic Algorithm (*NSGA*)

Srinivas e Deb (1994) propuseram o *NSGA*. As soluções não dominadas recebem uma alta qualificação. Para manter a diversidade da população, essa qualificação é dividida por um número proporcional à quantidade de soluções numa vizinhança σ de cada uma. Assim, os pontos não dominados que possuem uma vizinhança menos populosa, possuem maior potencial reprodutivo.

Os indivíduos não dominados são separados da população. Os outros indivíduos são classificados em não dominados ou dominados e o mesmo procedimento é repetido, sendo que a qualificação inicial do segundo nível dos não dominados será menor que a pior qualificação dos não dominados do primeiro nível.

Quando a aptidão de todos os indivíduos foi determinada, os reprodutores são selecionados.

O método é sensível ao parâmetro para calcular a vizinhança σ e, computacionalmente, é menos eficiente que o *MOGA*; no entanto, facilmente o número de objetivos pode ser aumentado.

5.3.4 Strength Pareto Evolutionary Algorithm (*SPEA*)

O *SPEA* foi proposto por Zitzler e Theile (1999). O algoritmo trabalha com a população que será evoluída e com uma outra externa, formada por parte das soluções não dominadas de cada geração.

Para a reprodução, o método forma um conjunto com as soluções da população corrente e as da externa. Cada indivíduo é avaliado pelo número de soluções que ele domina, garantindo que a pior solução não dominada é melhor que qualquer outra dominada.

Uma vantagem do *SPEA* é não acrescentar nenhum parâmetro, como aconteceu nos casos anteriores.

5.4 Nondominated Sorting Genetic Algorithm II

Para o estudo a seguir, foi escolhida uma nova versão do algoritmo desenvolvido por Srinivas e Deb [7], o *Nondominated Sorting Genetic Algorithm II* (*NSGA II*).

O *NSGA* possui um bom desempenho [28], porém várias críticas foram feitas à primeira versão:

- *alto custo computacional*: o *NSGA* possui complexidade $O(kN^3)$, onde k é o número de objetivos e N o tamanho da população, o que eleva o seu custo em problemas com população grandes;
- *ausência de elitismo*: alguns resultados [28] demonstraram que o elitismo melhora o desempenho dos algoritmos genéticos multi-objetivos, ajudando-os a encontrar boas soluções;
- *aumento do número de parâmetros do algoritmo*: para aumentar a diversidade, o *NSGA* calcula a quantidade de indivíduos que estão em uma vizinhança σ de cada solução. Em certo sentido, isto acaba dificultando o desempenho do algoritmo, uma vez que os resultados dependem do ajuste deste parâmetro.

Para sanar essas deficiências, foi publicada uma nova versão do método, computacionalmente mais simples, com estratégia elitista e sem o acréscimo de parâmetros. Esse algoritmo é descrito a seguir.

1. **Determinação das linhas frontais**: ao invés de simplesmente comparar os indivíduos de uma população para encontrar os não dominados, o *NSGA II* calcula, para cada solução i , o número de pontos n_i que a dominam e o conjunto S_i de soluções que i domina. Esses cálculos exigem $O(kN^2)$ comparações.

Se $n_i = 0$, então i é uma solução não dominada. O algoritmo retira da população todos os pontos não dominados, que formam o conjunto F_1 dos pontos da *linha frontal do Pareto*. Uma vez que estes pontos não pertencem mais a população, eles não dominam mais os pontos dos conjuntos S_i (com $n_i = 0$), portanto para todos os pontos j , onde $j \in S_i : n_i = 0$, é reduzida uma unidade de n_j . Após isso, os pontos com $n_j = 0$, serão separados em um conjunto H . Quando todos os pontos j foram verificados, H conterá todos os pontos que formam a *segunda linha frontal*; este processo será repetido até determinar todas as linhas frontais.

Abaixo se encontra o código do para determinar as linhas frontais de uma geração.

```

⇒ seja  $P$  um conjunto de pontos
⇒ para cada  $p \in P$ 
    para cada  $q \in P$ 
        se  $(p \prec q)$  então
             $S_p = S_p \cup \{q\}$ 
        senão se  $(q \prec p)$  então
             $n_p = n_p + 1$ 
    se  $n_p = 0$  então
         $F_1 = F_1 \cup p$ 
⇒  $i = 1$ 
⇒ enquanto  $F_i \neq \emptyset$ 
     $H = \emptyset$ 
    para cada  $p \in F_i$ 
        para cada  $q \in S_p$ 
             $n_q = n_q - 1$ 
            se  $n_q = 0$  então  $H = H \cup q$ 
     $i = i + 1$ 
     $F_i = H$ 
    A  $i$ -ésima linha frontal é formada
    por todos os elementos do conjunto  $H$ 

```

2. **Estimativa da distância dos vizinhos:** para estimar a densidade da vizinhança de uma solução, o *NSGA* define uma vizinhança σ e calcula o número de pontos que pertencem a esta região. O *NSGA II* utiliza outra medida: estima esta densidade a partir da distância dos vizinhos, ou seja, a distância entre os pontos da mesma linha frontal mais próximos à direita e à esquerda de uma solução.

O algoritmo usado para calcular a estimativa de densidade dos pontos de uma linha frontal encontra-se a seguir.

Seja F_j a j -ésima linha frontal

$\Rightarrow l = |F_j|$

l é o n° de pontos da j -ésima linha frontal

$\Rightarrow i = 2$

$\Rightarrow$ enquanto $i < l$

$d_i = 0$

distância inicial

$i = i + 1$

$\Rightarrow d_l = d_l = \infty$

pontos extremos são sempre selecionados

$\Rightarrow$ para cada objetivo k

para $i = 2, \dots, (l - 1)$

$d_i = d_i + |f_k(i + 1) - f_k(i - 1)|$ calcula-se a estimativa da distância

3. **Avaliação das soluções:** o processo de seleção baseia-se em uma avaliação de cada ponto. Para comparar as diversas soluções, o *NSGA II* utiliza os dois conceitos apresentados anteriormente: as linhas de não dominância e a distância dos vizinhos.

Se um ponto x pertence a uma linha frontal de menor índice que outro y , então x será melhor que y e, se as duas soluções pertencem à mesma linha frontal, então a solução que possui maior distância será melhor.

Em outras palavras, seja o operador de comparação ($\geq_n$) para classificar os indivíduos de uma geração. Sejam p e q duas soluções, tal que $p \in F_i$ e $q \in F_j$. Então:

$$p \geq_n q \text{ se } (i < j) \text{ ou } (i = j \text{ e } d_p > d_q)$$

4. **Laço principal:** inicialmente o *NSGA II* cria uma população aleatória P_0 de tamanho N , que é classificada de acordo com o nível de não dominância (as soluções da primeira linha frontal possuem avaliação 1, as da segunda, 2 e assim por diante). O algoritmo minimizará, ao longo das gerações, a avaliação das soluções.

Através dos operadores (seleção, recombinação e mutação) cria-se uma nova população de filhos Q_0 também de tamanho N . Uma população R_0 , de tamanho $2N$, é formada pela união do conjunto dos pais e dos filhos. Todos os indivíduos são avaliados e classificados, de acordo com o operador $\geq_n$, e os N melhores formarão a nova população P_1 . Os novos pais gerarão os próximos filhos Q_1 e assim por diante.

A seleção dos pais que irão gerar a população de filhos Q é feita por sorteio: com igual probabilidade para todos, sorteiam-se dois pais e o mais apto é selecionado.

A seguir se encontra o pseudocódigo com o laço principal do *NSGA II*.

$\Rightarrow R_t = P_t \cup Q_t$	pais e filhos formam uma única população
$\Rightarrow F = \text{linhas não dominantes } (R_t)$	todas as linhas não dominantes são calculadas $F = (F_1, F_2, \dots)$
$\Rightarrow i = 1$	
$\Rightarrow \text{enquanto } P_{t+1} < N$	enquanto que o tamanho da população dos novos pais é menor que N
cálculo da distância dos vizinhos (F_i)	calcula a distância dos vizinhos na i -ésima linha não dominante
$P_{t+1} = P_{t+1} \cup F_i$	inclui a i -ésima linha no conjunto dos candidatos a futuros pais
$i = i + 1$	
$\Rightarrow \text{ordena } (P_{t+1}, \geq_n)$	ordena a nova população
$\Rightarrow P_{t+1} = P_{t+1}[0 : n]$	os N melhores indivíduos serão os novos pais
$\Rightarrow Q_{t+1} = \text{nova população } (P_{t+1})$	através dos operadores de seleção geram-se os novos filhos
$\Rightarrow t = t + 1$	

Finalmente, a partir do conhecimento detalhado do funcionamento do *NSGA II*, podemos avaliar se as deficiências da primeira versão do algoritmo foram sanadas.

As complexidades computacionais das principais rotinas do algoritmo são:

- classificação das linhas não dominantes: $O(kN^2)$
- cálculo da distância dos vizinhos: $O(kN \log N)$
- ordenamento ($\geq_n$): $O(2N \log(2N))$

Portanto a maior complexidade é dada pela classificação das linhas não dominantes, $O(kN^2)$, e é menor que a primeira versão do algoritmo: $O(kN^3)$.

O elitismo também está presente no *NSGAII*, pois, as novas gerações são escolhidas a partir dos filhos e também dos antigos pais, ou seja, sempre as melhores soluções permanecem na população.

Já o número de parâmetros do problema não foi aumentado, uma vez que todas as rotinas, até a estimativa da densidade (cálculo da distância dos vizinhos), não exigem a participação de nenhum tipo de parâmetro.

5.5 Aplicação do NSGA II

Para estudar os problemas multi-objetivos, utilizamos 9 problemas que possuem duas funções objetivo e que se encontram no Apêndice C [8] [28]. Procuramos selecionar funções com características variadas, como mostra a tabela abaixo:

Tabela 5.1: Conjunto de problemas teste Multi-objetivos

<i>função</i>	<i>dimensão</i>	<i>características do Pareto-ótimo</i>
1	30	convexo
2	30	não convexo
3	30	várias partes convexas não contínuas
4	10	possui 21^9 ótimos locais
5	10	espaço de busca não uniforme
6	2	multimodal
7	2	várias partes convexas não contínuas
8	2	não convexo
9	2	multimodal; distribuição não uniforme no Pareto

A seguir encontra-se a descrição da primeira função e a população do algoritmo em algumas gerações para o cenário A. Repare que em 200 gerações o algoritmo está próximo de convergir e que a população possui uma distribuição uniforme ao longo da curva ótima de Pareto (representada pela linha contínua dos gráficos).

$$\min(f_1(x), f_2(x))$$

onde:

$$f_1(x) = x_1$$

$$g(x) = 1 + 9 \sum_{i=2}^{30} \frac{x_i}{29}$$

$$h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}}$$

$$f_2(x) = g(x) \cdot h(f_1, g)$$

$$0 \leq x \leq 1$$

O pareto ótimo é formado com $g(x) = 1$

Figura 5.1: Exemplo de um problema multi-objetivo

Todos os 9 problemas testes foram resolvidos nos três cenários (A, B e C) com uma função penalidade adaptada. Nos capítulos anteriores, foram calculadas as distâncias entre uma solução e o ótimo e também a diferença de valores da função objetivo nesses pontos (a descrição da penalidade encontra-se na seção 3.2). Uma vez que a otimização multi-objetivos procura o conjunto ótimo de Pareto, neste capítulo será utilizada uma medida de distância entre uma solução encontrada e a linha ótima de Pareto.

$$\gamma = \frac{ger \cdot N}{\alpha_1} + \alpha_2 \cdot ger(x) + (\alpha_3 + \alpha_4) \cdot (f_1(x) - f_1(y))$$

onde:

y é o ponto da linha ótima de Pareto onde $f_2(y) = f_2(x)$

$f_1(x)$ e $f_1(y)$ são os valores da primeira função objetivo para os pontos x e y

5.6 Probabilidade de recombinação

O resultado geral encontra-se no Apêndice C.

Figura 5.2: Penalidade normalizada média *versus* probabilidade de recombinação

Figura 5.3: Penalidade total normalizada *versus* probabilidade de recombinação

Graficamente, observamos que a penalidade normalizada média e a penalidade total normalizada diminuem com o aumento da probabilidade de recombinação. A primeira medida apresenta bons resultados para probabilidades a partir de 0.70 ($\bar{\gamma}_n < 0.36$) e a segunda, a partir de 0.88 ($\gamma_{tn} < 0.30$).

Ambos os gráficos possuem a mesma tendência, sendo a penalidade normalizada média mais suave.

Os resultados dos outros índices encontram-se a seguir.

n° de problemas

Figura 5.4: Número de problemas com baixas penalidades *versus* probabilidade de recombinação

n° de gerações

Figura 5.5: Número médio de gerações da melhor solução *versus* probabilidade de recombinação

Observa-se que o número de problemas com baixa penalidade tem um crescimento significativo com a probabilidade acima de 0,88. O número médio de gerações oscila entre 80 e 115 gerações e possui uma leve tendência de redução com o aumento da probabilidade de recombinação.

Podemos concluir que os melhores resultados encontrados, bem como os intervalos de probabilidades mais vantajosos são:

	p_c	<i>Intervalo</i>
penalidade normalizada média	0,98	0,70 a 1,00
penalidade total normalizada	0,94	0,88 a 1,00
número de problemas com baixa penalidade normalizada	0,98	0,88 a 1,00
número de gerações, dentro as menores soluções	0,98	0,88 a 1,00
<i>probabilidade média e intervalo principal</i>	<i>0,97</i>	<i>0,88 a 1,00</i>

Tabela 5.2: Melhores resultados para probabilidade de recombinação

5.7 Probabilidade de mutação

O resultado geral encontra-se no Apêndice C.

Figura 5.6: Penalidade normalizada média *versus* probabilidade de mutação

Figura 5.7: Penalidade total normalizada *versus* probabilidade de mutação

Observa-se que a penalidade normalizada média e a penalidade total normalizada possuem a mesma tendência, com bons resultados entre 0,005 e 0,055 ($\bar{\gamma}_n < 0,19$ e $\gamma_{tn} < 0,06$). Ao contrário dos casos anteriores, ambas as curvas são suaves, o que indica um comportamento semelhante em todos os problemas.

Os resultados dos outros índices encontram-se a seguir.

Figura 5.8: Número de problemas com baixas penalidades *versus* probabilidade de mutação

Figura 5.9: Número médio de gerações da melhor solução *versus* probabilidade de mutação

Graficamente, observamos que, para probabilidades de mutação entre 0,005 e 0,030 obtêm-se muitos problemas com baixa penalidade (acima de 20). O número médio de gerações oscila entre 100 e 160 para os melhores resultados e, novamente, o desvio padrão obtido foi alto.

Podemos concluir que os melhores resultados encontrados, bem como os intervalos de probabilidades mais vantajosos são:

	p_m	<i>Intervalo</i>
penalidade normalizada média	0,020	0,005 a 0,055
penalidade total normalizada	0,020	0,005 a 0,055
número de problemas com baixa penalidade normalizada	0,020	0,005 a 0,030
número de gerações, dentro as menores soluções	0,020	-
<i>probabilidade média e intervalo principal</i>	<i>0,020</i>	<i>0,005 a 0,030</i>

Tabela 5.3: Melhores resultados para probabilidade de mutação

5.8 Tamanho da população

O resultado geral para determinar o melhor tamanho de população encontra-se no Apêndice C.

Figura 5.10: Penalidade normalizada média *versus* tamanho da população

Figura 5.11: Penalidade total normalizada *versus* tamanho da população

Pela primeira vez penalidades normalizadas apresentam tendências distintas: a penalidade normalizada média indica que populações menores possuem melhores resultados e a penalidade total normalizada, que populações grandes são mais vantajosas. Outra curiosidade: a penalidade normalizada média não possui uma curva suave, como era de se esperar.

No entanto, tal discrepância deve-se ao resultado do problema 4: por ter 21⁹ ótimos locais, trata-se do problema mais difícil, possuindo uma penalidade muito maior que os outros e influenciando muito a penalidade total normalizada. Os gráficos abaixo apresentam os resultados dos outros 8 problemas. Observe que a tendência das duas medidas são idênticas.

Figura 5.12: Penalidade normalizada média (exceto o problema 4) *versus* tamanho da população

Figura 5.13: Penalidade total normalizada (exceto o problema 4) *versus* tamanho da população

Os resultados dos outros índices encontram-se a seguir.

Figura 5.14: Número de problemas com baixas penalidades *versus* tamanho da população

Figura 5.15: Número médio de gerações da melhor solução *versus* tamanho da população

Note que os gráficos também possuem resultados distintos: para populações pequenas, há mais problemas com baixa penalidade e para populações entre 200 e 300 indivíduos são necessárias menos gerações para atingir boas soluções.

Podemos concluir que os melhores resultados encontrados, bem como os intervalos de tamanhos mais vantajosos são:

	<i>N</i>	<i>Intervalo</i>
menor penalidade normalizada média	75	25-175
menor penalidade total normalizada	75	25-200
número de problemas com baixa penalidade normalizada	25	25-150
menor número de gerações, dentro as menores soluções	275	225-300
<i>tamanho médio e intervalo principal</i>	<i>100</i>	<i>25 a 150</i>

Tabela 5.4: Melhores resultados para o tamanho da população

5.9 Conclusão

Para problemas multi-objetivos, podemos concluir que os melhores resultados e os principais intervalos para os parâmetros estudados são:

	<i>Valor médio</i>	<i>Principal intervalo</i>
p_c	0,97	0,88 a 1,00
p_m	0,020	0,005 a 0,030
N	100	25 a 150
n^o de gerações	-	> 160

Tabela 5.5: Principais resultados do capítulo

Conclusões

6.1 Resultados globais

Após considerarmos cada tipo de problema individualmente, conviria estudar os parâmetros de modo global, confrontando os resultados de cada capítulo com a média dos resultados.

6.1.1 Probabilidade de recombinação

O resultado geral encontra-se no Apêndice D.

Figura 6.1: Penalidade normalizada média *versus* probabilidade de recombinação

Figura 6.2: Penalidade total normalizada *versus* probabilidade de recombinação

Graficamente, observamos que a penalidade normalizada média possui a mesma tendência em todos os tipos de problemas: maiores probabilidades de recombinação (entre de 0,80 e 1,00) tendem a gerar melhores resultados.

A penalidade total normalizada apresenta a mesma tendência (bons resultados entre 0,80 e 1,00), exceto para os problemas de minimização em caixas, que possui o melhor intervalo entre 0,74 e 0,82.

Conforme já foi ressaltado nos capítulos anteriores, a divergência dos gráficos provém da diferença conceitual entre as duas medidas. A penalidade normalizada média, por tratar-se de uma média, possui uma curva suave e atribui um peso semelhante para cada tipo de problema. Parece ser uma medida mais adequada do que a penalidade total normalizada.

Para comparar a quantidade de problemas com baixas penalidades, os resultados serão apresentados em porcentagens, uma vez que nos capítulos anteriores foram resolvidas quantidades diferentes de problemas.

% de problemas

Figura 6.3: Porcentagem de problemas com baixas penalidades *versus* probabilidade de recombinação

n° de gerações

Figura 6.4: Número médio de gerações da melhor solução *versus* probabilidade de recombinação

O número de problemas com baixa penalidade possui um nítido crescimento de acordo com o aumento da probabilidade de recombinação, apresentando bons resultados a partir de 0,80. Já para a geração média da melhor solução, há uma tendência de redução do número de gerações com o aumento da probabilidade: quanto maior a probabilidade de recombinação, maior será a chance de as melhores soluções transmitirem o seu material

genético para outros indivíduos, atraindo-os para si e ajudando o ótimo a surgir em um número menor de gerações.

Podemos concluir que os melhores resultados encontrados, bem como os intervalos de probabilidades mais vantajosos são:

	p_c	<i>Intervalo</i>
penalidade normalizada média	0,94	0,76 a 1,00
penalidade total normalizada	0,80	0,80 a 1,00
número de problemas com baixa penalidade normalizada	0,98	0,78 a 1,00
número de gerações, dentro as menores soluções	0,94	não há
<i>probabilidade média e intervalo principal</i>	<i>0,92</i>	<i>0,80 a 1,00</i>

Tabela 6.1: Melhores resultados para probabilidade de recombinação

6.1.2 Probabilidade de mutação

O resultado global e os gráficos comparativos encontram-se abaixo.

Figura 6.5: Penalidade normalizada média *versus* probabilidade de mutação

Figura 6.6: Penalidade total normalizada *versus* probabilidade de mutação

Nitidamente pode-se observar que tanto a penalidade normalizada média quanto a penalidade total normalizada possuem a mesma tendência para todos os tipos de problemas, com bons resultados entre 0,005 e 0,050.

Os resultados dos outros índices encontram-se a seguir.

% de problemas

Figura 6.7: Porcentagem de problemas com baixas penalidades *versus* probabilidade de mutação

n° de gerações

Figura 6.8: Número médio de gerações da melhor solução *versus* probabilidade de mutação

Graficamente, observa-se que todos os tipos de problemas possuem a mesma tendência no número de problemas com baixa penalidade: bons resultados também com probabilidade de mutação entre 0,005 e 0,050. O número médio de gerações oscila entre 200 e 250 para a média dos melhores resultados; no entanto, entre os tipos de problemas, esse número varia bastante. Convém lembrar que os desvios padrões obtidos nos capítulos anteriores foram altos e, portanto, torna-se necessário utilizar um número significativamente maior de gerações do que a média para obter resultados confiáveis.

Em todos os tipos de problemas, quando a probabilidade de mutação é zero, o método rapidamente converge para soluções ruins, mostrando a importância deste operador para o bom desempenho do algoritmo. Repare também que para probabilidades maiores que 0,050 o algoritmo necessita de um número um pouco maior de gerações para encontrar as melhores soluções do que no intervalo com os melhores resultados (entre 0,005 e 0,050). Pode-se concluir que um número grande de mutações retarda o surgimento da melhor solução e dificulta o desempenho do algoritmo: as mutações em grande quantidade acabam mais afastando as boas soluções do ponto ótimo do que aproximando.

Podemos concluir que os melhores resultados encontrados, bem como os intervalos de probabilidades mais vantajosos são:

	p_c	<i>Intervalo</i>
penalidade normalizada média	0,020	0,005 a 0,050
penalidade total normalizada	0,020	0,005 a 0,050
número de problemas com baixa penalidade normalizada	0,020	0,005 a 0,050
número de gerações, dentro as menores soluções	0,020	0,005 a 0,050
<i>probabilidade média e intervalo principal</i>	<i>0,020</i>	<i>0,005 a 0,050</i>

Tabela 6.2: Melhores resultados para probabilidade de mutação

6.1.3 Tamanho da população

O resultado global e os gráficos comparativos encontram-se abaixo.

Figura 6.9: Penalidade normalizada média *versus* tamanho da população

Figura 6.10: Penalidade total normalizada *versus* tamanho da população

A penalidade normalizada média possui uma tendência semelhante entre os tipos de problemas, exceto para os multi-objetivos, e os melhores resultados encontram-se entre 75 e 225. A penalidade total normalizada possui bons resultados entre 100 a 250.

Os resultados dos outros índices encontram-se a seguir.

% de problemas

Figura 6.11: Porcentagem de problemas com baixas penalidades *versus* tamanho da população

n° de gerações

Figura 6.12: Número médio de gerações da melhor solução *versus* tamanho da população

Graficamente, observa-se que a porcentagem média de problemas com baixas penalidades acompanha a mesma tendência dos resultados dos problemas de minimização em caixas e com restrições: populações com mais de 100 indivíduos possuem melhores resultados. Apenas os problemas multi-objetivos possuem uma tendência oposta: populações com menos de 150 pontos apresentam melhores soluções.

A geração média da melhor solução, mostra que os problemas multi-objetivos necessitam geralmente de 50 a 100 gerações para encontrar boas soluções e os outros tipos

de problemas, acima de 300 gerações. Provavelmente, isto se deve ao fato de que enquanto os problemas multi-objetivos encontram um conjunto de pontos (entre inúmeros possíveis da linha ótima de Pareto), os outros buscam uma única solução e, portanto, necessitam de um esforço maior. Outro fator, que pode ter influenciado os resultados, é que o Algoritmo Genético utilizado nos problemas multi-objetivos é mais sofisticado (e provavelmente mais eficiente) do que o das outras classes de problemas. Assim, o número de gerações necessárias pode estar influenciado mais pela eficiência do algoritmo utilizado do que propriamente pelas classes de problemas.

Note também que o número de gerações necessárias diminui no intervalo de 25 a 200 e permanece constante nas populações com mais de 200 pontos, ou seja, quanto maior o número de pontos, mais cedo surge o ótimo; porém, acima de 200, o aumento da população não contribui mais para essa tendência. Pode-se concluir que 200 pontos é o número suficiente de informações que o algoritmo necessita para encontrar boas soluções em um número razoável de gerações.

Podemos concluir que os melhores resultados encontrados, bem como os intervalos de probabilidades mais vantajosos são:

	<i>N</i>	<i>Intervalo</i>
penalidade normalizada média	125	75 a 225
penalidade total normalizada	275	100 a 275
número de problemas com baixa penalidade normalizada	125	> 100
número de gerações, dentre as menores soluções	-	100 a 200
<i>tamanho médio e intervalo principal</i>	175	100 a 200

Tabela 6.3: Melhores resultados para o tamanho da população

6.1.4 Resultados globais

A tabela a seguir apresenta a média dos resultados dos capítulos anteriores.

	<i>Valor médio</i>	<i>Principal intervalo</i>
p_c	0,92	0,80 a 1,00
p_m	0,020	0,005 a 0,050
N	175	100 a 200
n° de gerações	-	> 400

Tabela 6.4: Resultados globais

6.2 Análise de sensibilidade

Até o momento, pode-se concluir que, de um modo geral, os resultados globais divergem pouco dos de cada tipo de problema. No entanto, ainda falta uma análise da influência dos coeficientes de penalidades nos resultados; em outras palavras, será que os resultados não são conseqüências dos valores adotados para os coeficientes e representam apenas um caso particular?

Para estudar esta questão, foi feita uma análise de sensibilidade em cada um dos parâmetros α_i (descritos na seção 3.2) em função dos melhores resultados para as penalidades normalizadas médias:

1. α_1 : somente para a medida do tamanho da população há influência do coeficiente α_1 , o que já se era de esperar. O gráfico a seguir apresenta a variação do melhor tamanho da população pela variação deste coeficiente.

Figura 6.13: Coeficiente α_1 versus tamanho da população

Como é natural, o tamanho da população aumenta com o crescimento deste coeficiente, pois, deste modo, os problemas com um maior tamanho de população são privilegiados. Para os problemas multi-objetivos, populações entre 50 e 150 indivíduos apresentam bons resultados; para os outros problemas, o gráfico indica uma população entre 150 e 400, confirmando os resultados obtidos anteriormente.

2. α_2 : penaliza o número de gerações da melhor solução encontrada e é, portanto, o que mais influencia os resultados.

A probabilidade de recombinação global é pouco sensível ao α_2 , possuindo os melhores valores entre 0,88 e 0,92. Já para os outros problemas, o melhor valor oscila

Figura 6.14: Coeficiente α_2 versus probabilidade de recombinação

dentro de um intervalo previsto nos capítulos anteriores: para os problemas de minimização em caixas, entre 0,90 e 0,95; para os problemas com restrições, entre 0,80 e 0,84, e para os multi-objetivos, entre 0,94 e 0,98.

Figura 6.15: Coeficiente α_2 versus probabilidade de mutação

Para a probabilidade de mutação, o melhor valor para o resultado global oscila entre 0,010 e 0,020. Nos outros problemas, há uma variação pequena, novamente dentro de um intervalo previsto.

Figura 6.16: Coeficiente α_2 versus tamanho da população

Para o tamanho da população, o coeficiente α_2 confirma os resultados anteriores: os problemas multi-objetivos possuem boas soluções para populações de até 150 indivíduos e os outros, para populações de aproximadamente 200 pontos (ou até 300 para os problemas de minimização em caixas).

Convém lembrar que o coeficiente α_2 influencia bastante na qualidade dos problemas através da penalização, ou seja, um valor alto de α_2 seleciona problemas que rapidamente encontraram boas soluções e não necessariamente aqueles que, de fato, localizaram os melhores pontos. Em outras palavras, o coeficiente α_2 pode privilegiar uma convergência prematura e não é interessante que possua um valor alto. Um ponto importante desse coeficiente é servir de desempate quando diferentes parâmetros geram o mesmo resultado final.

3. α_3 e α_4 : os resultados globais, para todas as medidas, não possuem sensibilidade com relação a esses coeficientes. Os valores oscilam raras vezes e de maneira pouco significativa.

Portanto, a análise de sensibilidade apenas confirma os melhores resultados:

	p_c		p_m		N		$Gerações$
	resultado	intervalo	resultado	intervalo	resultado	intervalo	resultado
minimização em caixas	0,95	0,80 a 1,00	0,020	0,020 a 0,040	275	125 a 300	> 390
com restrições	0,88	0,84 a 0,98	0,038	0,010 a 0,050	210	100 a 275	> 360
multi-objetivos	0,97	0,88 a 1,00	0,020	0,005 a 0,030	100	25 a 150	> 160
global	0,92	0,80 a 1,00	0,020	0,005 a 0,050	175	100 a 200	> 400

Tabela 6.5: Principais resultados do trabalho

6.3 Novas medidas

Finalmente, para confirmar esses resultados, resolveram-se três outros problemas (um de minimização em caixas, outro com restrições e o terceiro multi-objetivos) com os parâmetros específicos para cada tipo de problema, apresentados na tabela anterior. No Apêndice E encontram-se as descrições dos problemas e as tabelas com os resultados.

Figura 6.17: Novas medidas para a probabilidade de recombinação

Graficamente, podemos observar que a probabilidade de recombinação não possui claramente uma faixa com bons resultados. Para o problema de minimização em caixas, a menor penalidade acontece com probabilidade de recombinação 0,70; porém há várias

probabilidades com bons resultados, incluindo o intervalo entre 0,84 a 1,00. Para o problema com restrições, os melhores resultados acontecem para probabilidades de 0,52 e 0,74; porém não há uma faixa definida para bons resultados. Já para a função multi-objetivos, a menor penalidade acontece para probabilidade de 0,94.

Esses resultados indicam que a probabilidade de recombinação é um parâmetro com desempenho pouco previsível para um problema específico. Geralmente, as probabilidades maiores (entre 0,80 e 1,00) tendem a possuir um melhor desempenho; porém, observando os resultados dos testes dos capítulos anteriores, geralmente é possível encontrar boas soluções com probabilidades menores que 0,80.

Figura 6.18: Novas medidas para a probabilidade de mutação

A probabilidade de mutação possui um comportamento mais previsível que a probabilidade de recombinação. Para o problema de minimização em caixas, a menor penalidade acontece com probabilidade de recombinação 0,020 e há bons resultados na faixa de zero a 0,060. Para o problema com restrições, o intervalo entre 0,015 e 0,030 possui bons resultados e, curiosamente, há também outro entre 0,15 e 0,16 com desempenho semelhante. Já para a função multi-objetivos, a menor penalidade acontece para probabilidade de 0,040 e os bons resultados estão entre 0,005 e 0,050.

Conclui-se que probabilidade de mutação é, de fato, um parâmetro com desempenho mais previsível. Geralmente, as probabilidade menores (entre 0,005 e 0,050) tendem a possuir um melhor desempenho; nos resultados desse último teste, apenas a função com restrições teve um comportamento não esperado, ainda que não sejam contraditórias com as conclusões finais desse parâmetro.

Figura 6.19: Novas medidas para o tamanho da população

Com relação ao tamanho da população, os problemas de minimização em caixas e multi-objetivos apresentaram um comportamento dentro do esperado, com bons resultados para populações acima de 150 e entre 50 e 100 indivíduos, respectivamente. Já o problema com restrições, possui alguns tamanhos com bons resultados, porém não há um intervalo definido; o melhor resultado (275) encontra-se dentro da melhor faixa estimada no item anterior. Os resultados confirmam as conclusões da seção anterior.

6.4 Conclusão final

Com exceção da probabilidade de mutação, pode-se observar que há um grande intervalo em que os parâmetros estudados possuem bons resultados; ou seja, o Algoritmo Genético é pouco sensível à probabilidade de recombinação e ao tamanho da população.

Já a probabilidade de mutação, tem a função de dar variabilidade à população e levar o algoritmo a regiões não exploradas. Dentro de uma faixa determinada, esse operador enriquece a busca, como foi possível observar em todos os capítulos anteriores. Segundo os biólogos, no processo evolutivo, a mutação é considerada a matéria-prima da evolução e a seleção natural, o mecanismo de escolha das características mais adaptadas ou de redução das menos aptas; de fato, observa-se que esses conceitos estão enraizados na essência e no desempenho do algoritmo.

Por outro lado, de um modo geral, os parâmetros possuem um comportamento semelhante nos diversos tipos de problema. Parece que, apesar da existência de diversos métodos ou adaptações do Algoritmo Genético para funções restritas e problemas multi-

objetivos, as conclusões apresentadas, eventualmente, poderiam ser aplicadas a outras classes de problemas.

Um fator que não foi abordado, e que poderia ser o foco de um futuro trabalho, é o estudo da relação entre a dimensão dos problemas e os parâmetros do algoritmo. A partir de uma análise superficial dos resultados obtidos, pôde-se observar que problemas de maiores dimensões necessitam de mais gerações para encontrar boas soluções; ou seja, o custo desses problemas pode ser consideravelmente maior para problemas com muitas variáveis. No entanto, com o avanço tecnológico e uma disponibilidade cada vez maior de capacidade de processamento, o custo tende a ter um peso menor e o Algoritmo Genético, a ser ainda mais vantajoso para esse tipo de problema.

Outros trabalhos futuros que poderiam dar continuidade ou aprofundar esse estudo:

- estudo de outras medidas de comparação (penalidade) dos resultados: funções multi-objetivos, *performance profiling*, etc;
- análise estatística dos resultados;
- aplicação dos métodos e estudo dos parâmetros para problemas reais (científicos, de engenharia ou de finanças);
- estudo dos parâmetros em outros métodos do Algoritmos Genético;
- análise de problemas multi-objetivos com restrições;
- comparação dos métodos para problemas com restrições ou multi-objetivos;
- estudo de outros tipos de problemas (binários, combinatórios, etc.);
- comparação do Genético com outros métodos de otimização;
- estudo dos métodos híbridos, principalmente com outros algoritmos com princípios de otimização;
- aplicação e análise das sugestões de Goldberg (medidas de similaridade, controle adaptativo de parâmetros, número de populações, tamanho da população, genes dominantes e recessivos).

Problemas de minimização em caixas

A.1 Conjunto de problemas teste

1. Função Michalewicz:

$$f(x) = - \sum_{i=1}^n \left(\sin(x_i) \cdot \sin\left(\frac{i \cdot x_i^2}{\pi}\right)^{2m} \right)$$

onde:

$$m = 10, n = 5$$

$$0 \leq x_i \leq \pi$$

$$\text{mínimo global: } f(x^*) = -4,687$$

2. Função Michalewicz:

$$\text{com } m = 10, n = 10$$

$$\text{mínimo global: } f(x^*) = -9,66$$

3. Função Rosenbrock's valley (ou De Jong's function 2):

$$f(x) = \sum_{i=1}^{n-1} (100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2)$$

onde:

$$-2,048 \leq x_i \leq 2,048$$

$$\text{com } n = 5$$

$$\text{mínimo global: } f(x^*) = 0 \text{ e } x_i^* = 1, i = 1, \dots, n$$

4. Função Rosenbrock's valley (ou De Jong's function 2):

com $n = 10$ mínimo global: $f(x^*) = 0$ e $x_i^* = 1, i = 1, \dots, n$

5. Função Schwefel

$$f(x) = \sum_{i=1}^n (-x_i \cdot \text{sen}(\sqrt{|x_i|}))$$

onde:

$$-500 \leq x_i \leq 500$$

com $n = 5$ mínimo global: $f(x^*) = -2094,914436$ e $x_i^* = 420,9687, i = 1, \dots, n$

6. Função Schwefel

com $n = 10$ mínimo global: $f(x^*) = -4189,8289$ e $x_i^* = 420,9687, i = 1, \dots, n$

7. Função Ackley's Path

$$f(x) = -a \cdot e^{-b \cdot \sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2}} - e^{\frac{1}{n} \sum_{i=1}^n \cos(c \cdot x_i)} + a + e$$

onde:

$$a = 20 \quad b = 0,2 \quad c = 2\pi$$

$$-32,768 \leq x_i \leq 32,768$$

com $n = 5$ mínimo global: $f(x^*) = 0$ e $x_i^* = 0, i = 1, \dots, n$

8. Função Ackley's Path

com $n = 10$ mínimo global: $f(x^*) = 0$ e $x_i^* = 0, i = 1, \dots, n$

9. Função Griewangk

$$f(x) = \frac{\sum_{i=1}^n x_i^2}{4000} - \prod_{i=1}^n \frac{\cos(x_i)}{\sqrt{i}} + 1$$

onde:

$$-600 \leq x_i \leq 600$$

com $n = 5$

mínimo global: $f(x^*) = 0$ e $x_i^* = 0, i = 1, \dots, n$

10. Função Griewangk

com $n = 10$

mínimo global: $f(x^*) = 0$ e $x_i^* = 0, i = 1, \dots, n$

11. Função Rastrigin

$$f(x) = 10 \cdot n + \sum_{i=1}^n (x_i^2 - 10 \cdot \cos(2\pi x_i))$$

onde:

$$-5,12 \leq x_i \leq 5,12$$

com $n = 5$

mínimo global: $f(x^*) = 0$ e $x_i^* = 0, i = 1, \dots, n$

12. Função Rastrigin

com $n = 10$

mínimo global: $f(x^*) = 0$ e $x_i^* = 0, i = 1, \dots, n$

13. Função Easom

$$f(x) = -\cos x_1 \cdot \cos x_2 \cdot e^{-((x_1-\pi)^2 + (x_2-\pi)^2)}$$

onde:

$$-100 \leq x_i \leq 100, i = 1, 2$$

mínimo global: $f(x^*) = -1$ e $x_i^* = \pi, i = 1, 2$

14. Modificação da função Easom para $n=4$

$$f(x) = -\cos x_1 \cdot \cos x_2 \cdot \cos x_3 \cdot \cos x_4 \cdot e^{-((x_1-\pi)^2+(x_2-\pi)^2+(x_3-\pi)^2+(x_4-\pi)^2)}$$

onde:

$$-100 \leq x_i \leq 100, i = 1, \dots, 4$$

$$\text{mínimo global: } f(x^*) = -1 \text{ e } x_i^* = \pi, i = 1, \dots, 4$$

15. Função Goldstein-Price

$$f(x) = [1 + (x_1 + x_2 + 1)^2 \cdot (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \\ \cdot [30 + (2x_1 - 3x_2)^2 \cdot (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$$

onde:

$$-2 \leq x_i \leq 2, i = 1, 2$$

$$\text{mínimo global: } f(x^*) = 3 \text{ e } x^* = (0, -1)$$

16. Modificação da função Goldstein-Price para $n=4$

$$f(x) = [1 + (x_1 + x_2 + 1)^2 \cdot (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \\ \cdot [30 + (2x_1 - 3x_2)^2 \cdot (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)] \\ \cdot [1 + (x_3 + x_4 + 1)^2 \cdot (19 - 14x_3 + 3x_3^2 - 14x_4 + 6x_3x_4 + 3x_4^2)] \\ \cdot [30 + (2x_3 - 3x_4)^2 \cdot (18 - 32x_3 + 12x_3^2 + 48x_4 - 36x_3x_4 + 27x_4^2)]$$

onde:

$$-2 \leq x_i \leq 2, i = 1, \dots, 4$$

$$\text{mínimo global: } f(x^*) = 9 \text{ e } x^* = (0, -1, 0, -1)$$

17. Função Six-hump camel back

$$f(x) = (4 - 2.1 \cdot x_1^2 + x_1^{4/3}) \cdot x_1^2 + x_1x_2 + (-4 + 4x_2^2) \cdot x_2^2$$

onde:

$$-3 \leq x_1 \leq 3, -2 \leq x_2 \leq 2$$

$$\text{mínimo global: } f(x^*) = -1.0316 \text{ e } x^* = (-0, 0898; 0, 7126), (0, 0898; -0, 7126)$$

18. Modificação da Função Six-hump camel back para $n=4$

$$f(x) = (4 - 2, 1 \cdot x_1^2 + x_1^{4/3})x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2 + (4 - 2, 1 \cdot x_3^2 + x_3^{4/3})x_3^2 + x_3x_4 + (-4 + 4x_4^2)x_4^2$$

onde:

$$-3 \leq x_1 \leq 3, -2 \leq x_2 \leq 2, -3 \leq x_3 \leq 3, -2 \leq x_4 \leq 2$$

mínimo global: $f(x^*) = -2,063256$ e $x^* = (-0,0898; 0,7126; -0,0898; 0,7126), (0,0898; -0,7126; -0,0898; 0,7126), (-0,0898; 0,7126; 0,0898; -0,7126), (0,0898; -0,7126; 0,0898; -0,7126)$

19. Função Moved axis parallel hyper-ellipsoid

$$f(x) = \sum_{i=1}^n (i \cdot x_i^2)$$

onde:

$$-5,12 \leq x_i \leq 5,12$$

com $n = 5$

mínimo global: $f(x^*) = 0$ e $x_i^* = 5 \cdot i, i = 1, \dots, n$

20. Função Moved axis parallel hyper-ellipsoid

com $n = 10$

mínimo global: $f(x^*) = 0$ e $x_i^* = 5 \cdot i, i = 1, \dots, n$

21. Função de Branin

$$f(x) = a(x_2 - bx_1^2 + cx_1 - d)^2 + e(1 - f) \cos(x_1) + e$$

onde:

$$a = 1, b = \frac{5,1}{4\pi^2}, c = \frac{5}{\pi}, d = 6, e = 10, f = \frac{1}{8\pi}$$

$$-5 \leq x_1 \leq 10, 0 \leq x_2 \leq 15$$

mínimo global: $f(x^*) = 0,397887$ e $x_i^* = (-\pi; 12,275), (\pi; 2,275)$ e $(9,42478; 2,475)$

22. Modificação da função de Branin para $n=6$

$$f(x) = a(x_2 - bx_1^2 + cx_1 - d)^2 + e(1 - f) \cos(x_1) + e +$$

$$a(x_4 - bx_3^2 + cx_3 - d)^2 + e(1 - f) \cos(x_3) + e +$$

$$a(x_6 - bx_5^2 + cx_5 - d)^2 + e(1 - f) \cos(x_5) + e$$

onde:

$$a = 1, b = \frac{5,1}{4\pi^2}, c = \frac{5}{\pi}, d = 6, e = 10, f = \frac{1}{8\pi}$$

$$-5 \leq x_i \leq 10, i = 1, 3, 5; 0 \leq x_j \leq 15, j = 2, 4, 6$$

mínimo global: $f(x^*) = 1,193661$ e

x_i^* é formado pela combinação dos três pontos ótimos da função

23. Função rotated hyper-ellipsoid

$$f(x) = \sum_{i=1}^n \sum_{j=1}^i x_j^2$$

onde:

$$-65,536 \leq x_i \leq 65,536$$

com $n = 5$

mínimo global: $f(x^*) = 0$ e $x_i^* = 0, i = 1, \dots, n$

24. Função rotated hyper-ellipsoid

com $n = 10$

mínimo global: $f(x^*) = 0$ e $x_i^* = 0, i = 1, \dots, n$

25. Função soma de potências

$$f(x) = \sum_{i=1}^n |x_i|^{i+1}$$

onde:

$$-1 \leq x_i \leq 1$$

com $n = 5$

mínimo global: $f(x^*) = 0$ e $x_i^* = 0, i = 1, \dots, n$

26. Função soma de potências

com $n = 10$ mínimo global: $f(x^*) = 0$ e $x_i^* = 0, i = 1, \dots, n$

A.2 Resultados

Tabela A.1: Penalidades *versus* probabilidade de recombinação

p_c	$\bar{\gamma}_n$	γ_{tn}
0,50	0,55	1,00
0,52	0,50	0,50
0,54	0,49	0,33
0,56	0,51	0,93
0,58	0,51	0,60
0,60	0,49	0,67
0,62	0,48	0,50
0,64	0,44	0,60
0,66	0,43	0,78
0,68	0,42	0,50
0,70	0,40	0,50
0,72	0,44	0,52
0,74	0,43	0,35

p_c	$\bar{\gamma}_n$	γ_{tn}
0,76	0,36	0,00
0,78	0,34	0,00
0,80	0,37	0,26
0,82	0,38	0,26
0,84	0,35	0,50
0,86	0,36	0,50
0,88	0,38	0,67
0,90	0,34	0,67
0,92	0,35	0,67
0,94	0,34	0,67
0,96	0,35	0,67
0,98	0,39	0,67
1,00	0,29	0,50

 p_c é a probabilidade de recombinação $\bar{\gamma}_n$ é a penalidade normalizada média: média das penalidades normalizadas dos 26 problemas γ_{tn} é a penalidade total normalizada: conversão da penalidade total em penalidade normalizada

Tabela A.2: Número de problemas e de gerações *versus* probabilidade de recombinação

p_c	np	Geração	σ
0,50	7	431	277
0,52	16	417	291
0,54	18	421	290
0,56	6	424	301
0,58	6	418	303
0,60	11	398	288
0,62	11	401	290
0,64	16	400	284
0,66	18	402	283
0,68	17	393	279
0,70	17	399	277
0,72	12	390	263
0,74	13	387	263

p_c	np	Geração	σ
0,76	22	378	275
0,78	22	381	274
0,80	23	384	269
0,82	21	380	270
0,84	21	366	270
0,86	22	374	269
0,88	22	380	296
0,90	25	387	295
0,92	20	380	270
0,94	22	372	274
0,96	18	377	271
0,98	19	380	272
1,00	28	373	280

p_c é a probabilidade de recombinação

np é o número de problemas com baixa penalidade ($< 0,1$ em cada um dos cenários)

Geração é o número médio de gerações em que nasceu a melhor solução encontrada e o seu respectivo desvio padrão σ

Tabela A.3: Penalidades *versus* probabilidade de mutação

p_m	$\tilde{\gamma}_n$	γ_{tn}
0,000	0,43	1,00
0,005	0,14	0,22
0,010	0,13	0,22
0,015	0,13	0,15
0,020	0,13	0,00
0,025	0,15	0,00
0,030	0,19	0,00
0,035	0,18	0,00
0,040	0,18	0,07
0,045	0,21	0,08
0,050	0,24	0,07
0,055	0,24	0,08
0,060	0,26	0,07
0,065	0,28	0,44
0,070	0,28	0,20
0,075	0,28	0,23
0,080	0,29	0,17
0,085	0,30	0,23
0,090	0,31	0,16
0,095	0,30	0,25
0,100	0,34	0,18

p_m	$\tilde{\gamma}_n$	γ_{tn}
0,105	0,36	0,40
0,110	0,35	0,60
0,115	0,36	0,47
0,120	0,37	0,48
0,125	0,38	0,33
0,130	0,38	0,42
0,135	0,39	0,19
0,140	0,39	0,46
0,145	0,44	0,36
0,150	0,45	0,45
0,155	0,41	0,32
0,160	0,43	0,50
0,165	0,45	0,55
0,170	0,48	0,58
0,175	0,51	0,76
0,180	0,49	0,35
0,185	0,51	0,78
0,190	0,54	0,73
0,195	0,53	0,93
0,200	0,56	0,78

p_m é a probabilidade de mutação

$\tilde{\gamma}_n$ é a penalidade normalizada média: média das penalidades normalizadas dos 26 problemas

γ_{tn} é a penalidade total normalizada: conversão da penalidade total em penalidade normalizada

Tabela A.4: Número de problemas e de gerações *versus* probabilidade de mutação

p_m	np	Geração	σ
0,000	29	40	35
0,005	56	206	155
0,010	51	237	166
0,015	55	263	172
0,020	56	279	173
0,025	50	295	171
0,030	48	306	168
0,035	48	317	168
0,040	46	322	168
0,045	45	326	163
0,050	35	335	165
0,055	40	336	160
0,060	37	344	165
0,065	36	344	158
0,070	31	350	156
0,075	32	349	153
0,080	27	342	151
0,085	26	348	145
0,090	30	349	148
0,095	31	340	151
0,100	26	349	145

p_m	np	Geração	σ
0,105	24	345	140
0,110	22	347	144
0,115	22	348	143
0,120	20	344	138
0,125	24	334	146
0,130	19	343	137
0,135	21	348	133
0,140	18	334	136
0,145	15	347	122
0,150	15	338	126
0,155	13	338	134
0,160	11	334	128
0,165	12	338	140
0,170	14	341	129
0,175	9	336	127
0,180	11	329	125
0,185	10	329	127
0,190	11	336	132
0,195	9	328	133
0,200	7	333	131

p_m é a probabilidade de mutação

np é o número de problemas com baixa penalidade ($< 0,1$ em cada um dos cenários)

Geração é o número médio de gerações em que nasceu a melhor solução encontrada e o seu respectivo desvio padrão σ

Tabela A.5: Penalidades *versus* tamanho da população

N	$\bar{\gamma}_n$	γ_{tn}
25	0,76	1,00
50	0,41	0,48
75	0,28	0,13
100	0,22	0,20
125	0,20	0,00
150	0,19	0,00
175	0,20	0,07
200	0,19	0,00
225	0,19	0,00
250	0,21	0,00
275	0,19	0,00
300	0,18	0,00

N	$\bar{\gamma}_n$	γ_{tn}
325	0,22	0,00
350	0,21	0,00
375	0,23	0,00
400	0,25	0,00
425	0,24	0,00
450	0,24	0,00
475	0,26	0,00
500	0,27	0,00
525	0,28	0,00
550	0,29	0,00
575	0,32	0,00
600	0,29	0,00

N é tamanho da população

$\bar{\gamma}_n$ é a penalidade normalizada média: média das penalidades normalizadas dos 26 problemas

γ_{tn} é a penalidade total normalizada: conversão da penalidade total em penalidade normalizada

Tabela A.6: Número de problemas e de gerações *versus* tamanho da população

N	np	Geração	σ
25	11	468	256
50	20	447	302
75	28	372	289
100	38	355	296
125	36	342	297
150	39	332	305
175	36	334	310
200	38	328	307
225	38	318	309
250	38	319	326
275	41	334	335
300	43	312	311

N	np	Geração	σ
325	45	320	330
350	39	306	326
375	43	312	329
400	37	319	332
425	42	326	341
450	37	300	335
475	39	316	339
500	35	306	341
525	37	302	324
550	32	297	327
575	29	319	346
600	30	288	318

N é tamanho da população

np é o número de problemas com baixa penalidade ($< 0,1$ em cada um dos cenários)

Geração é o número médio de gerações em que nasceu a melhor solução encontrada e o seu respectivo desvio padrão σ

Problemas com restrições

B.1 Conjunto de problemas teste

1. Primeiro problema

$$\min f(x) = x_1 + x_2 + x_3$$

sujeito a:

$$-1 + 0,0025(x_4 + x_6) \leq 0$$

$$-1 + 0,0025(-x_4 + x_5 + x_7) \leq 0$$

$$-1 + 0,01(-x_5 + x_8) \leq 0$$

$$100x_1 - x_1x_6 + 8333,33252x_4 - 83333,333 \leq 0$$

$$x_2x_4 - x_2x_7 - 1250x_4 + 1250x_5 \leq 0$$

$$x_3x_5 - x_3x_8 - 2500x_5 + 1250000 \leq 0$$

$$100 \leq x_1 \leq 10000$$

$$1000 \leq x_2 \leq 10000$$

$$1000 \leq x_3 \leq 10000$$

$$10 \leq x_4 \leq 1000$$

$$10 \leq x_5 \leq 1000$$

$$10 \leq x_6 \leq 1000$$

$$10 \leq x_7 \leq 1000$$

$$10 \leq x_8 \leq 1000$$

mínimo global: $f(x^*) = 7049,25$ e

$$x^* = (579,19; 1360,13; 5109,92; 182,01; 295,60; 217,99; 286,40; 395,60)$$

2. Segundo problema

$$\min f(x) = 37,293239x_1 + 0,8356891x_1x_5 + 5,3578547x_3^2 - 40792,141$$

sujeito a:

$$-0,0022053x_3x_5 + 0,0056858x_2x_5 + 0,0006262x_1x_4 - 6,665593 \leq 0$$

$$0,0022053x_3x_5 - 0,0056858x_2x_5 - 0,0006262x_1x_4 - 85,334407 \leq 0$$

$$0,0071317x_2x_5 + 0,0021813x_3^2 + 0,0029955x_1x_2 - 29,48751 \leq 0$$

$$-0,0071317x_2x_5 - 0,0021813x_3^2 - 0,0029955x_1x_2 + 9,48751 \leq 0$$

$$0,0047026x_3x_5 + 0,0019085x_3x_4 + 0,0012547x_1x_3 - 15,699039 \leq 0$$

$$-0,0047026x_3x_5 - 0,0019085x_3x_4 - 0,0012547x_1x_3 + 10,699039 \leq 0$$

$$78 \leq x_1 \leq 102$$

$$33 \leq x_2 \leq 45$$

$$27 \leq x_3 \leq 45$$

$$27 \leq x_4 \leq 45$$

$$27 \leq x_5 \leq 45$$

$$\text{mínimo global: } f(x^*) = -30665,5387 \text{ e } x^* = (78; 33; 29,9953; 45; 36,7758)$$

3. Terceiro problema

$$\min f(x) = -x_1 - x_2$$

sujeito a:

$$x_2 \leq 2 + 2x_1^4 - 8x_1^3 + 8x_1^2$$

$$x_2 \leq 4x_1^4 - 32x_1^3 + 88x_1^2 - 96x_1 + 36$$

$$0 \leq x_1 \leq 3$$

$$0 \leq x_2 \leq 4$$

$$\text{mínimo global: } f(x^*) = -5,50796 \text{ e } x^* = (2,3295; 3,1785)$$

4. Quarto problema

$$\min f(x) = -12x_1 - 7x_2 + x_2^2$$

sujeito a:

$$-2x_1^4 + 2 - x_2 = 0$$

$$0 \leq x_1 \leq 2$$

$$0 \leq x_2 \leq 3$$

$$\text{mínimo global: } f(x^*) = -16,73889 \text{ e } x^* = (0,7175; 1,4700)$$

5. Quinto problema

$$\min f(x) = 3x_1 + 0,000001x_1^3 + 2x_2 + \frac{0,000002}{3x_2^3}$$

sujeito a:

$$x_4 - x_3 + 0,55 \geq 0$$

$$x_3 - x_4 + 0,55 \geq 0$$

$$1000 \cdot \text{sen}(-x_3 - 0,25) + 1000 \cdot \text{sen}(-x_4 - 0,25) + 894,8 - x_1 = 0$$

$$1000 \cdot \text{sen}(x_3 - 0,25) + 1000 \cdot \text{sen}(x_3 - x_4 - 0,25) + 894,8 - x_2 = 0$$

$$1000 \cdot \text{sen}(x_4 - 0,25) + 1000 \cdot \text{sen}(x_4 - x_3 - 0,25) + 1294,8 = 0$$

$$0 \leq x_i \leq 1200, i = 1, 2$$

$$-0,55 \leq x_i \leq 0,55, i = 3, 4$$

$$\text{mínimo global: } f(x^*) = 5126,4981 \text{ e } x^* = (679,9453; 1026,0670; 0,1188764; -0,3962336)$$

6. Sexto problema

$$\min f(x) = x_2 + 10^{-5}(x_2 - x_1)^2 - 1$$

sujeito a:

$$\frac{x_1}{\sqrt{3}} - x_2 \geq 0$$

$$x_1 - \sqrt{3} \cdot x_2 + 6 \geq 0$$

$$0 \leq x_i \leq 2, i = 1, 2$$

$$\text{mínimo global: } f(x^*) = -1 \text{ e } x^* = (0; 0)$$

7. Sétimo problema

$$\min f(x) = \frac{\text{sen}^3(2\pi x_1) \cdot \text{sen}(2\pi x_2)}{x_1^3(x_1 + x_2)}$$

sujeito a:

$$x_1^2 - x_2 + 1 \leq 0$$

$$1 - x_1 + (x_2 - 4)^2 \leq 0$$

$$0 \leq x_i \leq 10, i = 1, 2$$

$$\text{mínimo global: } f(x^*) = -0,095825 \text{ e } x^* = (1,2281; 4,2453)$$

8. Oitavo problema

$$\min f(x) = (\sqrt{n})^n \cdot \prod_{i=1}^n x_i$$

sujeito a:

$$\sum_{i=1}^n x_i^2 = 1$$

$$0 \leq x_i \leq 1, i = 1, \dots, n$$

com $n = 23$

$$\text{mínimo global: } f(x^*) = 1 \text{ e } x_i^* = \frac{1}{\sqrt{23}}, i = 1, \dots, n$$

9. Nono problema

$$\min f(x) = \sum_{i=1}^{10} x_i \left(c_i + \ln \left(\frac{x_i}{x_1 + \dots + x_{10}} \right) \right)$$

sujeito a:

$$x_1 + 2x_2 + 2x_3 + x_6 + x_{10} = 2$$

$$x_3 + x_7 + x_8 + 2x_9 + x_{10} = 1$$

$$x_4 + 2x_5 + x_6 + x_7 = 1$$

$$0,000001 \leq x_i \leq 1, i = 1, \dots, 10$$

onde:

$$c_1 = -6.089; c_2 = -17.164; c_3 = -34.054; c_4 = -5.914; c_5 = -24.721;$$

$$c_6 = -14.986; c_7 = -24.100; c_8 = -10.708; c_9 = -26.662; c_{10} = -22.179$$

$$\text{mínimo global: } f(x^*) = -47,76065 \text{ e } x^* = (0,04034785; 0,15386976; 0,77497089;$$

$$0,00167479; 0,48468539; 0,00068965; 0,02826479; 0,01849179; 0,03849563; 0,10128126)$$

10. Décimo problema

$$\min f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$$

sujeito a:

$$-x_1 - x_2^2 \leq 0$$

$$-x_1^2 - x_2 \leq 0$$

onde:

$$0 \leq x_i \leq 1, i = 1, 2$$

$$\text{mínimo global: } f(x^*) = 0,25 \text{ e } x^* = (0,50; 0,25)$$

11. Décimo primeiro problema

$$\min f(x) = 0,01x_1^2 + x_2^2$$

sujeito a:

$$-x_1x_2 + 25 \leq 0$$

$$-x_1^2 - x_2^2 + 25 \leq 0$$

onde:

$$2 \leq x_1 \leq 50, 0 \leq x_2 \leq 50$$

$$\text{mínimo global: } f(x^*) = 5 \text{ e } x^* = (15,811388; 1,581139)$$

12. Décimo segundo problema

$$\min f(x) = (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2 + 10x_5^6 + 7x_6^2 + x_7^4 + \\ -4x_6x_7 - 10x_6 - 8x_7$$

sujeito a:

$$127 - 2x_1^2 - 3x_2^4 - x_3 - 4x_4^2 - 5x_5 \geq 0$$

$$196 - 23x_1 - x_2^2 - 6x_6^2 + 8x_7 \geq 0$$

$$282 - 7x_1 - 3x_2 - 10x_3^2 - x_4 + x_5 \geq 0$$

$$-4x_1^2 - x_2^2 + 3x_1x_2 - 2x_3^2 - 5x_6 + 11x_7 \geq 0$$

onde:

$$-10 \leq x_i \leq 10, i = 1, \dots, 7$$

$$\text{mínimo global: } f(x^*) = 680,6300573 \text{ e}$$

$$x^* = (2,3304990; 1,9513720; -0,4775414; 4,3657260; -0,6244870; 1,0381310; 1,5942270)$$

13. Décimo terceiro problema

$$\min f(x) = (x_1 - 10)^3 + (x_2 - 20)^3$$

sujeito a:

$$(x_1 - 5)^2 + (x_2 - 5)^2 - 100 \geq 0$$

$$-(x_1 - 6)^2 - (x_2 - 5)^2 - 82,81 \geq 0$$

onde:

$$13 \leq x_1 \leq 100, 0 \leq x_2 \leq 100$$

$$\text{mínimo global: } f(x^*) = -6961,81381 \text{ e } x^* = (14,095; 0,84296)$$

14. Décimo quarto problema

$$\min f(x) = (x_1 - 2)^2 + (x_2 - 1)^2$$

sujeito a:

$$-x_1^2 + x_2 \geq 0$$

$$x_1 + x_2 \leq 0$$

onde:

$$0 \leq x_i \leq 10, i = 1, 2$$

mínimo global: $f(x^*) = 1$ e $x^* = (1; 1)$

15. Décimo quinto problema

$$\min_{x,y} f(x, y) = c^T x - 0,5x^T Qx + dy$$

sujeito a:

$$6x_1 + 3x_2 + 3x_3 + 2x_4 + x_5 \leq 6,5$$

$$10x_1 + 10x_3 + y \leq 20$$

onde:

$$0 \leq x \leq 1, 0 \leq y \leq 100$$

$$c = (-10, 5; -7, 5; -3, 5; -2, 5; -1, 5)$$

$$d = -10$$

$$Q = 100I$$

mínimo global: $f(x^*) = -361,5$ e $x^* = (0; 1; 0; 1; 1), y^* = 20$

16. Décimo sexto problema

$$\min_{x,y} f(x, y) = c^T x - 0,5x^T Qx + dy$$

sujeito a:

$$2x_1 + 2x_2 + y_6 + y_7 \leq 10$$

$$2x_1 + 2x_3 + y_6 + y_8 \leq 10$$

$$2x_2 + 2x_3 + y_7 + y_8 \leq 10$$

$$-8x_1 + y_6 \leq 0$$

$$-8x_2 + y_7 \leq 0$$

$$-8x_3 + y_8 \leq 0$$

$$-2x_4 - y_1 + y_6 \leq 0$$

$$-2y_2 - y_3 + y_7 \leq 0$$

$$-2y_4 - y_5 + y_8 \leq 0$$

onde:

$$0 \leq x \leq 1$$

$$0 \leq y_i \leq 1, i = 1, 2, 3, 4, 5, 9$$

$$0 \leq y_i \leq 3, i = 6, 7, 8$$

$$c = (5; 5; 5; 5)$$

$$d = (-1; -1; -1; -1; -1; -1; -1; -1; -1)$$

$$Q = 100I$$

mínimo global: $f(x^*) = -195$ e $(x^*) = (1; 1; 1; 1), y^* = (1; 1; 1; 1; 1; 3; 3; 3; 1)$

B.2 Resultados

Tabela B.1: Penalidades *versus* probabilidade de recombinação

p_c	$\bar{\gamma}_n$	γ_{tn}	p_c	$\bar{\gamma}_n$	γ_{tn}
0,50	0,56	0,40	0,76	0,40	1,00
0,52	0,54	0,53	0,78	0,40	1,00
0,54	0,50	0,53	0,80	0,43	0,01
0,56	0,49	0,08	0,82	0,37	0,00
0,58	0,50	0,09	0,84	0,36	0,02
0,60	0,48	0,62	0,86	0,37	0,03
0,62	0,52	0,62	0,88	0,40	0,11
0,64	0,44	0,17	0,90	0,43	0,11
0,66	0,43	0,15	0,92	0,38	0,07
0,68	0,40	0,50	0,94	0,36	0,01
0,70	0,38	0,51	0,96	0,38	0,07
0,72	0,45	0,07	0,98	0,38	0,08
0,74	0,49	0,07	1,00	0,39	0,53

p_c é a probabilidade de recombinação

$\bar{\gamma}_n$ é a penalidade normalizada média: média das penalidades normalizadas dos 16 problemas

γ_{tn} é a penalidade total normalizada: conversão da penalidade total em penalidade normalizada

$\sum \gamma$ é a penalidade total: soma das penalidades das 26 funções irrestritas nos 3 cenários

Tabela B.2: Número de problemas e de gerações *versus* probabilidade de recombinação

p_c	np	Geração	σ
0,50	4	400	291
0,52	8	340	274
0,54	9	332	281
0,56	9	402	281
0,58	7	397	287
0,60	10	372	322
0,62	10	390	315
0,64	11	373	295
0,66	9	355	301
0,68	12	318	276
0,70	13	328	276
0,72	11	401	304
0,74	9	395	304

p_c	np	Geração	σ
0,76	9	343	306
0,78	10	343	308
0,80	10	358	310
0,82	10	343	313
0,84	15	310	297
0,86	12	320	300
0,88	12	354	297
0,90	13	358	297
0,92	11	326	288
0,94	12	327	290
0,96	12	351	307
0,98	12	345	303
1,00	13	327	280

p_c é a probabilidade de recombinação

np é o número de problemas com baixa penalidade ($< 0,1$ em cada um dos cenários)

Geração é o número médio de gerações em que nasceu a melhor solução encontrada e o seu respectivo desvio padrão σ

Tabela B.3: Penalidades *versus* probabilidade de mutação

p_m	$\bar{\gamma}_n$	γ_{tn}
0,000	0,44	0,86
0,005	0,19	0,16
0,010	0,17	0,07
0,015	0,23	0,08
0,020	0,19	0,00
0,025	0,21	0,15
0,030	0,22	0,06
0,035	0,20	0,34
0,040	0,20	0,00
0,045	0,22	0,09
0,050	0,23	0,04
0,055	0,31	0,13
0,060	0,29	0,03
0,065	0,29	0,00
0,070	0,31	0,07
0,075	0,32	0,06
0,080	0,42	0,08
0,085	0,35	0,08
0,090	0,37	0,22
0,095	0,37	0,02
0,100	0,40	0,13

p_m	$\bar{\gamma}_n$	γ_{tn}
0,105	0,37	0,10
0,110	0,38	0,23
0,115	0,44	0,22
0,120	0,46	0,51
0,125	0,40	0,47
0,130	0,39	0,18
0,135	0,43	0,25
0,140	0,43	0,34
0,145	0,46	0,56
0,150	0,44	0,28
0,155	0,41	0,10
0,160	0,51	0,50
0,165	0,49	1,00
0,170	0,46	0,46
0,175	0,48	0,53
0,180	0,53	0,56
0,185	0,46	0,23
0,190	0,49	0,39
0,195	0,47	0,53
0,200	0,46	0,18

p_m é a probabilidade de mutação

$\bar{\gamma}_n$ é a penalidade normalizada média: média das penalidades normalizadas dos 16 problemas

γ_{tn} é a penalidade total normalizada: conversão da penalidade total em penalidade normalizada

Tabela B.4: Número de problemas e de gerações *versus* probabilidade de mutação

p_m	np	Geração	σ
0,000	19	42	77
0,005	26	215	185
0,010	27	251	187
0,015	26	218	189
0,020	31	218	169
0,025	25	229	167
0,030	24	228	160
0,035	26	195	160
0,040	25	232	169
0,045	22	227	172
0,050	23	214	161
0,055	20	223	161
0,060	17	219	156
0,065	21	228	161
0,070	19	227	143
0,075	16	226	157
0,080	14	269	149
0,085	17	225	152
0,090	16	233	162
0,095	17	224	158
0,100	14	247	165

p_m	np	Geração	σ
0,105	15	218	144
0,110	13	218	142
0,115	9	221	142
0,120	12	213	146
0,125	12	222	148
0,130	15	231	165
0,135	10	219	142
0,140	9	266	151
0,145	11	239	159
0,150	9	248	137
0,155	12	212	138
0,160	9	269	153
0,165	10	222	135
0,170	12	240	148
0,175	10	229	147
0,180	6	238	139
0,185	12	241	145
0,190	8	251	140
0,195	5	206	137
0,200	10	223	150

p_m é a probabilidade de mutação

np é o número de problemas com baixa penalidade ($< 0,1$ em cada um dos cenários)

Geração é o número médio de gerações em que nasceu a melhor solução encontrada e o seu respectivo desvio padrão σ

Tabela B.5: Penalidades *versus* tamanho da população

N	$\bar{\gamma}_n$	γ_{tn}
25	0,72	0,22
50	0,44	0,53
75	0,39	0,71
100	0,32	0,09
125	0,32	0,13
150	0,26	0,31
175	0,29	0,02
200	0,25	0,24
225	0,27	0,13
250	0,29	0,54
275	0,25	0,00
300	0,31	0,91

N	$\bar{\gamma}_n$	γ_{tn}
325	0,32	0,37
350	0,26	0,88
375	0,27	0,38
400	0,30	0,37
425	0,34	1,00
450	0,32	0,32
475	0,31	0,63
500	0,30	0,54
525	0,34	0,16
550	0,36	0,47
575	0,39	0,68
600	0,35	0,13

N é tamanho da população

$\bar{\gamma}_n$ é a penalidade normalizada média: média das penalidades normalizadas dos 16 problemas

γ_{tn} é a penalidade total normalizada: conversão da penalidade total em penalidade normalizada

Tabela B.6: Número de problemas e de gerações *versus* tamanho da população

N	np	Geração	σ
25	6	438	282
50	13	331	302
75	7	354	272
100	11	290	287
125	18	320	298
150	21	293	308
175	18	314	323
200	17	290	275
225	17	315	308
250	19	300	313
275	18	324	316
300	15	290	292

N	np	Geração	σ
325	15	311	327
350	20	260	306
375	17	253	259
400	14	276	287
425	14	300	289
450	17	277	307
475	19	292	296
500	15	284	306
525	15	262	271
550	11	292	277
575	14	298	314
600	14	282	290

N é tamanho da população

np é o número de problemas com baixa penalidade ($< 0,1$ em cada um dos cenários)

Geração é o número médio de gerações em que nasceu a melhor solução encontrada e o seu respectivo desvio padrão σ

Problemas Multi-objetivos

C.1 Conjunto de problemas teste

1. Primeiro problema

$$\min(f_1(x), f_2(x))$$

onde:

$$f_1(x) = x_1$$

$$g(x) = 1 + 9 \sum_{i=2}^{30} \frac{x_i}{29}$$

$$h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}}$$

$$f_2(x) = g(x) \cdot h(f_1, g)$$

$$0 \leq x \leq 1$$

O pareto ótimo é formado com $g(x) = 1$

2. Segundo problema

$$\min(f_1(x), f_2(x))$$

onde:

$$f_1(x) = x_1$$

$$g(x) = 1 + 9 \sum_{i=2}^{30} \frac{x_i}{29}$$

$$h(f_1, g) = 1 - \left(\frac{f_1}{g} \right)^2$$

$$f_2(x) = g(x) \cdot h(f_1, g)$$

$$0 \leq x \leq 1$$

O pareto ótimo é formado com $g(x) = 1$

3. Terceiro problema

$$\min(f_1(x), f_2(x))$$

onde:

$$f_1(x) = x_1$$

$$g(x) = 1 + 9 \sum_{i=2}^{30} \frac{x_i}{29}$$

$$h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}} - \frac{f_1}{g} \cdot \text{sen}(10\pi f_1)$$

$$f_2(x) = g(x) \cdot h(f_1, g)$$

$$0 \leq x \leq 1$$

O pareto ótimo é formado com $g(x) = 1$

4. Quarto problema

$$\min(f_1(x), f_2(x))$$

onde:

$$f_1(x) = x_1$$

$$g(x) = 91 + \sum_{i=2}^{10} (x_i^2 - 10 \cdot \cos(4\pi x_i))$$

$$h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}}$$

$$f_2(x) = g(x) \cdot h(f_1, g)$$

$$0 \leq x_1 \leq 1, -5 \leq x_i \leq 5, i = 2, \dots, 10$$

O pareto ótimo é formado com $g(x) = 1, 25$

5. Quinto problema

$$\min(f_1(x), f_2(x))$$

onde:

$$f_1(x) = 1 - e^{-4x_1} \cdot \text{sen}^6(6\pi x_1)$$

$$g(x) = 1 + \left(\sum_{i=2}^{10} x_i \right)^{0,25}$$

$$h(f_1, g) = 1 - \left(\frac{f_1}{g} \right)^2$$

$$f_2(x) = g(x) \cdot h(f_1, g)$$

$$0 \leq x \leq 1$$

O pareto ótimo é formado com $g(x) = 1$

6. Sexto problema

$$\min(f_1(x), f_2(x))$$

onde:

$$f_1(x) = x_1$$

$$f_2(x) = 2 - e^{-\left(\frac{x_2-0.2}{0.004}\right)^2} - 0,8e^{-\left(\frac{x_2-0.6}{0.4}\right)^2}$$

$$0 \leq x \leq 1$$

O pareto ótimo é formado com $f_2(x) = \frac{0.7057}{f_1(x)}$

7. Sétimo problema

$$\min(f_1(x), f_2(x))$$

onde:

$$f_1(x) = x_1$$

$$g(x) = 1 + 10x_2$$

$$h(f_1, g) = 1 - \left(\frac{f_1}{g}\right)^2 - \frac{f_1}{g} \cdot \text{sen}(8\pi f_1)$$

$$f_2(x) = g(x) \cdot h(f_1, g)$$

$$0 \leq x \leq 1$$

O pareto ótimo é formado com $f_2(x) = 1 - f_1(x)^2 - f_1(x) \cdot \text{sen}(8\pi f_1(x))$

8. Oitavo problema

$$\min(f_1(x), f_2(x))$$

onde:

$$f_1(x) = 4x_1$$

$$g(x) = \begin{cases} 4 - 3e^{\left(-\frac{x_2-0.2}{0.02}\right)^2} & \text{se } x_2 \leq 0,4 \\ 4 - 2e^{\left(-\frac{x_2-0.7}{0.2}\right)^2} & \text{se } 0,4 \leq x_2 \leq 1 \end{cases}$$

$$h(x) = \begin{cases} 1 - \left(\frac{f_1}{g}\right)^{0.25} & \text{se } f_1 \leq g \\ 0 & \text{caso contrário} \end{cases}$$

$$f_2(x) = g(x) \cdot h(f_1, g)$$

$$0 \leq x \leq 1$$

O pareto ótimo é formado com $f_2(x) = 1 - f_1(x)^{0.25}$

9. Nono problema

$$\min(f_1(x), f_2(x))$$

onde:

$$f_1(x) = 1 - (e^{-4x_1} \cdot (\text{sen}^4(5\pi x_1)))$$

$$g(x) = \begin{cases} 4 - 3e^{(-\frac{x_2-0,2}{0,02})^2} & \text{se } x_2 \leq 0,4 \\ 4 - 2e^{(-\frac{x_2-0,7}{0,2})^2} & \text{se } 0,4 \leq x_2 \leq 1 \end{cases}$$

$$h(x) = \begin{cases} 1 - \left(\frac{f_1}{g}\right)^{0,25} & \text{se } f_1 \leq g \\ 0 & \text{caso contrário} \end{cases}$$

$$f_2(x) = g(x) \cdot h(f_1, g)$$

$$0 \leq x \leq 1$$

O pareto ótimo é formado com $f_2(x) = 1 - f_1(x)^4$

C.2 Resultados

Tabela C.1: Penalidades *versus* probabilidade de recombinação

p_c	$\bar{\gamma}_n$	γ_{tn}	p_c	$\bar{\gamma}_n$	γ_{tn}
0,50	0,51	0,59	0,76	0,31	0,34
0,52	0,37	0,19	0,78	0,23	0,32
0,54	0,40	0,31	0,80	0,29	0,31
0,56	0,39	0,28	0,82	0,29	0,37
0,58	0,43	0,24	0,84	0,36	0,34
0,60	0,44	0,49	0,86	0,30	0,49
0,62	0,50	1,00	0,88	0,24	0,27
0,64	0,39	0,36	0,90	0,26	0,12
0,66	0,42	0,53	0,92	0,24	0,29
0,68	0,46	0,48	0,94	0,16	0,00
0,70	0,35	0,30	0,96	0,17	0,30
0,72	0,30	0,14	0,98	0,13	0,01
0,74	0,27	0,34	1,00	0,24	0,23

p_c é a probabilidade de recombinação

$\bar{\gamma}_n$ é a penalidade normalizada média: média das penalidades normalizadas dos 9 problemas

γ_{tn} é a penalidade total normalizada: conversão da penalidade total em penalidade normalizada

Tabela C.2: Número de problemas e de gerações *versus* probabilidade de recombinação

p_c	np	Geração	σ
0,50	9	109	95
0,52	10	102	83
0,54	10	102	84
0,56	10	100	82
0,58	9	111	101
0,60	8	96	63
0,62	8	94	66
0,64	8	100	82
0,66	10	115	107
0,68	8	107	96
0,70	10	91	64
0,72	13	94	88
0,74	12	98	81

p_c	np	Geração	σ
0,76	9	83	46
0,78	13	93	87
0,80	13	94	68
0,82	12	106	95
0,84	10	96	82
0,86	11	106	98
0,88	12	96	90
0,90	12	94	63
0,92	13	93	74
0,94	15	91	88
0,96	16	91	54
0,98	19	80	49
1,00	12	96	84

p_c é a probabilidade de recombinação

np é o número de problemas com baixa penalidade ($< 0,1$ em cada um dos cenários)

Geração é o número médio de gerações em que nasceu a melhor solução encontrada e o seu respectivo desvio padrão σ

Tabela C.3: Penalidades *versus* probabilidade de mutação

p_m	$\bar{\gamma}_n$	γ_{tn}
0,000	0,44	0,27
0,005	0,13	0,03
0,010	0,09	0,01
0,015	0,06	0,02
0,020	0,05	0,00
0,025	0,09	0,03
0,030	0,07	0,01
0,035	0,09	0,02
0,040	0,10	0,04
0,045	0,13	0,05
0,050	0,16	0,08
0,055	0,19	0,09
0,060	0,23	0,11
0,065	0,24	0,16
0,070	0,23	0,17
0,075	0,29	0,23
0,080	0,27	0,24
0,085	0,32	0,31
0,090	0,35	0,34
0,095	0,32	0,37
0,100	0,35	0,41

p_m	$\bar{\gamma}_n$	γ_{tn}
0,105	0,38	0,46
0,110	0,42	0,51
0,115	0,42	0,55
0,120	0,43	0,58
0,125	0,44	0,59
0,130	0,45	0,60
0,135	0,47	0,65
0,140	0,49	0,70
0,145	0,52	0,73
0,150	0,56	0,75
0,155	0,54	0,78
0,160	0,51	0,80
0,165	0,56	0,83
0,170	0,60	0,86
0,175	0,63	0,89
0,180	0,65	0,93
0,185	0,65	0,95
0,190	0,64	0,96
0,195	0,65	0,98
0,200	0,66	1,00

p_m é a probabilidade de mutação

$\bar{\gamma}_n$ é a penalidade normalizada média: média das penalidades normalizadas dos 9 problemas

γ_{tn} é a penalidade total normalizada: conversão da penalidade total em penalidade normalizada

Tabela C.4: Número de problemas e de gerações *versus* probabilidade de mutação

p_m	np	<i>Geração</i>	σ
0,000	6	50	32
0,005	23	87	47
0,010	25	96	68
0,015	25	94	64
0,020	24	102	70
0,025	19	122	114
0,030	18	115	83
0,035	16	126	113
0,040	14	135	118
0,045	14	159	162
0,050	14	154	160
0,055	14	139	134
0,060	12	146	145
0,065	12	150	154
0,070	13	144	141
0,075	8	141	141
0,080	11	143	141
0,085	10	143	142
0,090	10	143	139
0,095	11	135	136
0,100	9	131	139

p_m	np	<i>Geração</i>	σ
0,105	9	132	139
0,110	9	133	139
0,115	10	136	139
0,120	11	139	140
0,125	10	137	140
0,130	9	135	140
0,135	9	135	140
0,140	9	135	139
0,145	9	132	139
0,150	8	130	138
0,155	9	129	136
0,160	9	128	134
0,165	8	127	136
0,170	7	126	139
0,175	6	130	138
0,180	5	133	138
0,185	6	135	139
0,190	6	137	139
0,195	6	132	140
0,200	5	128	140

p_m é a probabilidade de mutação

np é o número de problemas com baixa penalidade ($< 0,1$ em cada um dos cenários)

Geração é o número médio de gerações em que nasceu a melhor solução encontrada e o seu respectivo desvio padrão σ

Tabela C.5: Penalidades *versus* tamanho da população

N	$\bar{\gamma}_n$	γ_{tn}
25	0,24	1,00
50	0,55	0,89
75	0,16	0,59
100	0,52	0,49
125	0,20	0,18
150	0,50	0,26
175	0,28	0,08
200	0,58	0,16
225	0,37	0,09
250	0,66	0,24
275	0,47	0,00
300	0,74	0,12

N	$\bar{\gamma}_n$	γ_{tn}
325	0,54	0,14
350	0,64	0,23
375	0,44	0,33
400	0,59	0,30
425	0,49	0,08
450	0,74	0,09
475	0,59	0,36
500	0,84	0,37
525	0,64	0,25
550	0,79	0,37
575	0,49	0,22
600	0,69	0,23

N é tamanho da população

$\bar{\gamma}_n$ é a penalidade normalizada média: média das penalidades normalizadas dos 9 problemas

γ_{tn} é a penalidade total normalizada: conversão da penalidade total em penalidade normalizada

Tabela C.6: Penalidades (todos os problemas, exceto o 4) *versus* tamanho da população

N	$\bar{\gamma}_n$	γ_{tn}
25	0,15	0,49
50	0,52	0,77
75	0,09	0,00
100	0,53	0,70
125	0,18	0,12
150	0,53	0,71
175	0,29	0,23
200	0,62	0,77
225	0,40	0,35
250	0,72	0,87
275	0,52	0,48
300	0,83	0,97

N	$\bar{\gamma}_n$	γ_{tn}
325	0,54	0,73
350	0,64	0,48
375	0,44	0,66
400	0,59	0,81
425	0,49	0,60
450	0,74	0,96
475	0,59	0,65
500	0,84	0,69
525	0,64	0,44
550	0,79	1,00
575	0,49	0,33
600	0,69	0,75

N é tamanho da população

$\bar{\gamma}_n$ é a penalidade normalizada média: média das penalidades normalizadas dos 8 problemas

γ_{tn} é a penalidade total normalizada: conversão da penalidade total em penalidade normalizada

Tabela C.7: Número de problemas e de gerações *versus* tamanho da população

N	np	Geração	σ
25	19	72	66
50	8	102	77
75	14	63	49
100	3	87	45
125	10	59	31
150	3	83	44
175	1	65	48
200	2	81	50
225	3	59	31
250	2	70	29
275	3	54	13
300	3	69	34

N	np	Geração	σ
325	2	84	26
350	1	73	80
375	3	72	62
400	4	87	70
425	2	81	40
450	1	71	76
475	2	71	31
500	3	73	18
525	3	83	19
550	1	74	63
575	2	86	62
600	2	82	14

N é tamanho da população

np é o número de problemas com baixa penalidade ($< 0,1$ em cada um dos cenários)

Geração é o número médio de gerações em que nasceu a melhor solução encontrada e o seu respectivo desvio padrão σ

Resultados Globais

Tabela D.1: Penalidades *versus* probabilidade de recombinação

p_c	$\bar{\gamma}_n$	γ_{tn}
0,50	0,54	0,66
0,52	0,47	0,40
0,54	0,46	0,39
0,56	0,46	0,43
0,58	0,48	0,31
0,60	0,47	0,59
0,62	0,50	0,71
0,64	0,42	0,38
0,66	0,43	0,49
0,68	0,43	0,49
0,70	0,38	0,44
0,72	0,40	0,24
0,74	0,40	0,25

p_c	$\bar{\gamma}_n$	γ_{tn}
0,76	0,36	0,44
0,78	0,32	0,44
0,80	0,36	0,19
0,82	0,35	0,21
0,84	0,36	0,29
0,86	0,34	0,34
0,88	0,34	0,35
0,90	0,34	0,30
0,92	0,32	0,34
0,94	0,29	0,23
0,96	0,30	0,35
0,98	0,30	0,25
1,00	0,31	0,42

p_c é a probabilidade de recombinação

$\bar{\gamma}_n$ é a penalidade normalizada média

γ_{tn} é a penalidade total normalizada

Tabela D.2: Número de problemas e de gerações *versus* probabilidade de recombinação

p_c	np	Geração
0,50	17	314
0,52	25	286
0,54	26	285
0,56	21	309
0,58	19	309
0,60	22	289
0,62	22	295
0,64	24	291
0,66	26	291
0,68	25	273
0,70	29	272
0,72	29	295
0,74	27	293

p_c	np	Geração
0,76	27	268
0,78	32	272
0,80	33	279
0,82	31	276
0,84	32	258
0,86	31	266
0,88	33	277
0,90	35	280
0,92	32	266
0,94	36	263
0,96	36	273
0,98	40	268
1,00	36	265

p_c é a probabilidade de recombinação

np é o número de problemas com baixa penalidade ($< 0,1$ em cada um dos cenários)

Geração é o número médio de gerações em que nasceu a melhor solução encontrada

Tabela D.3: Penalidades *versus* probabilidade de mutação

p_m	$\bar{\gamma}_n$	γ_{tn}
0,000	0,44	0,71
0,005	0,15	0,13
0,010	0,13	0,10
0,015	0,14	0,08
0,020	0,13	0,00
0,025	0,15	0,06
0,030	0,16	0,02
0,035	0,16	0,12
0,040	0,16	0,04
0,045	0,19	0,07
0,050	0,21	0,06
0,055	0,25	0,10
0,060	0,26	0,07
0,065	0,27	0,20
0,070	0,27	0,15
0,075	0,30	0,17
0,080	0,33	0,16
0,085	0,32	0,21
0,090	0,34	0,24
0,095	0,33	0,22
0,100	0,36	0,24

p_m	$\bar{\gamma}_n$	γ_{tn}
0,105	0,37	0,32
0,110	0,38	0,45
0,115	0,40	0,41
0,120	0,42	0,53
0,125	0,41	0,46
0,130	0,41	0,40
0,135	0,43	0,36
0,140	0,44	0,50
0,145	0,47	0,55
0,150	0,48	0,50
0,155	0,45	0,40
0,160	0,49	0,60
0,165	0,50	0,79
0,170	0,51	0,63
0,175	0,54	0,73
0,180	0,56	0,61
0,185	0,54	0,65
0,190	0,56	0,69
0,195	0,55	0,81
0,200	0,56	0,66

p_m é a probabilidade de mutação

$\bar{\gamma}_n$ é a penalidade normalizada média: média das penalidades normalizadas de todos os problemas

γ_{tn} é a penalidade total normalizada: conversão da penalidade total em penalidade normalizada

Tabela D.4: Número de problemas e de gerações *versus* probabilidade de mutação

p_m	np	Geração
0,000	33	44
0,005	70	169
0,010	71	195
0,015	72	192
0,020	75	199
0,025	62	215
0,030	59	217
0,035	58	213
0,040	54	230
0,045	52	237
0,050	48	234
0,055	48	233
0,060	42	236
0,065	45	241
0,070	42	241
0,075	35	239
0,080	35	251
0,085	35	238
0,090	36	242
0,095	39	233
0,100	32	242

p_m	np	Geração
0,105	32	232
0,110	30	233
0,115	28	235
0,120	30	232
0,125	31	231
0,130	30	236
0,135	27	234
0,140	25	245
0,145	25	239
0,150	23	238
0,155	24	226
0,160	22	244
0,165	22	229
0,170	23	236
0,175	18	231
0,180	15	234
0,185	19	235
0,190	18	241
0,195	14	222
0,200	16	228

p_m é a probabilidade de mutação

np é o número de problemas com baixa penalidade ($< 0,1$ em cada um dos cenários)

Geração é o número médio de gerações em que nasceu a melhor solução encontrada

Tabela D.5: Penalidades *versus* tamanho da população

N	$\bar{\gamma}_n$	γ_{tn}
25	0,58	0,74
50	0,47	0,63
75	0,28	0,48
100	0,35	0,26
125	0,24	0,10
150	0,32	0,19
175	0,26	0,06
200	0,34	0,13
225	0,28	0,07
250	0,39	0,26
275	0,30	0,00
300	0,41	0,34

N	$\bar{\gamma}_n$	γ_{tn}
325	0,36	0,16
350	0,37	0,41
375	0,31	0,22
400	0,38	0,20
425	0,36	0,46
450	0,43	0,16
475	0,39	0,24
500	0,47	0,24
525	0,42	0,14
550	0,48	0,20
575	0,40	0,29
600	0,44	0,10

N é tamanho da população

$\bar{\gamma}_n$ é a penalidade normalizada média: média das penalidades normalizadas de todos os problemas

γ_{tn} é a penalidade total normalizada: conversão da penalidade total em penalidade normalizada

Tabela D.6: Número de problemas e de gerações *versus* tamanho da população

<i>N</i>	<i>np</i>	<i>Geração</i>
25	32	326
50	27	293
75	34	263
100	28	244
125	40	241
150	35	236
175	29	237
200	31	233
225	32	231
250	32	230
275	34	237
300	32	223

<i>N</i>	<i>np</i>	<i>Geração</i>
325	32	235
350	32	214
375	34	212
400	30	228
425	30	235
450	29	220
475	32	229
500	29	224
525	30	212
550	23	223
575	25	234
600	25	215

N é tamanho da população

np é o número de problemas com baixa penalidade ($< 0,1$ em cada um dos cenários)

Geração é o número médio de gerações em que nasceu a melhor solução encontrada

Novos resultados

E.1 Nova função de minimização em caixas

$$f(x) = 21,5 + x_1 \cdot \text{sen}(4\pi x_1) + x_2 \cdot \text{sen}(20\pi x_2)$$

onde:

$$-3,0 \leq x_1 \leq 12,1$$

$$4,1 \leq x_2 \leq 5,8$$

mínimo global: $f(x^*) = -38,850292$ e $x^* = (11,6255; 5,7250)$

p_c	γ_n	γ	<i>Melhor solução</i>	x_1	x_2	<i>Geração(x)</i>
0,50	0,20	131	-38,8503	11,6255	5,7250	56
0,52	0,18	129	-38,8503	11,6255	5,7250	54
0,54	0,18	129	-38,8503	11,6255	5,7250	54
0,56	0,11	123	-38,8503	11,6255	5,7250	48
0,58	0,29	140	-38,8503	11,6255	5,7250	65
0,60	0,15	127	-38,8503	11,6255	5,7250	52
0,62	0,14	126	-38,8503	11,6255	5,7250	51
0,64	0,51	161	-38,7503	11,6255	5,6251	65
0,66	0,18	129	-38,8503	11,6255	5,7250	54
0,68	0,16	128	-38,8503	11,6255	5,7250	53
0,70	0,00	112	-38,8503	11,6255	5,7250	37
0,72	0,21	132	-38,8503	11,6255	5,7250	57
0,74	0,23	134	-38,8503	11,6255	5,7250	59
0,76	0,09	121	-38,8503	11,6255	5,7250	46
0,78	0,24	135	-38,8503	11,6255	5,7250	60
0,80	1,00	209	-38,8503	11,6255	5,7250	134
0,82	0,18	129	-38,8503	11,6255	5,7250	54
0,84	0,21	132	-38,8503	11,6255	5,7250	57
0,86	0,19	130	-38,8503	11,6255	5,7250	55
0,88	0,11	123	-38,8503	11,6255	5,7250	48
0,90	0,38	149	-38,8503	11,6255	5,7250	74
0,92	0,18	129	-38,8503	11,6255	5,7250	54
0,94	0,19	130	-38,8503	11,6255	5,7250	55
0,96	0,25	136	-38,8503	11,6255	5,7250	61
0,98	0,41	152	-38,8503	11,6255	5,7250	77
1,00	0,19	130	-38,8503	11,6255	5,7250	55

Tabela E.1: Penalidade *versus* probabilidade de recombinação - novos resultados em um problema de minimização em caixas

p_m	γ_n	γ	<i>Melhor solução</i>	x_1	x_2	<i>Geração(x)</i>
0,000	0,01	119	-38,8503	11,6255	5,7250	44
0,005	0,02	124	-38,8503	11,6255	5,7250	49
0,010	0,01	112	-38,8503	11,6255	5,7250	37
0,015	0,02	128	-38,8503	11,6255	5,7250	53
0,020	0,00	106	-38,8503	11,6255	5,7250	31
0,025	0,04	141	-38,8503	11,6255	5,7250	66
0,030	0,04	148	-38,8503	11,6255	5,7250	73
0,035	0,08	178	-38,8503	11,6255	5,7250	103
0,040	0,14	239	-38,8503	11,6255	5,7250	164
0,045	0,08	183	-38,8503	11,6255	5,7250	108
0,050	0,07	176	-38,8503	11,6255	5,7250	101
0,055	0,13	233	-38,8503	11,6255	5,7250	158
0,060	0,07	170	-38,8503	11,6255	5,7250	95
0,065	0,26	350	-38,8503	11,6255	5,7250	275
0,070	0,08	184	-38,8503	11,6255	5,7250	109
0,075	0,87	930	-38,8503	11,6255	5,7250	855
0,080	0,16	253	-38,8503	11,6255	5,7250	178
0,085	0,85	909	-38,8503	11,6255	5,7250	834
0,090	0,23	326	-38,8503	11,6255	5,7250	251
0,095	1,00	1054	-38,8503	11,6255	5,7250	979
0,100	0,51	594	-38,8503	11,6255	5,7250	519
0,105	0,81	874	-38,8503	11,6255	5,7250	799
0,110	0,45	532	-38,8503	11,6255	5,7250	457
0,115	0,93	989	-38,8503	11,6256	5,7250	914
0,120	0,61	685	-38,8503	11,6255	5,7250	610
0,125	0,49	573	-38,8503	11,6255	5,7250	498
0,130	0,67	738	-38,8503	11,6255	5,7251	663
0,135	0,95	1008	-38,8503	11,6255	5,7250	933
0,140	0,54	619	-38,8503	11,6255	5,7250	544
0,145	0,59	662	-38,8503	11,6255	5,7250	587

p_m	γ_n	γ	<i>Melhor solução</i>	x_1	x_2	<i>Geração(x)</i>
0,150	0,81	871	-38,8503	11,6256	5,7250	796
0,155	0,88	942	-38,8503	11,6254	5,7250	867
0,160	0,98	1035	-38,8501	11,6251	5,7250	960
0,165	0,83	889	-38,8503	11,6254	5,7250	814
0,170	0,97	1029	-38,8503	11,6256	5,7251	954
0,175	0,95	1009	-38,8503	11,6254	5,7251	934
0,180	0,55	628	-38,8502	11,6255	5,7251	553
0,185	0,41	497	-38,8503	11,6256	5,7250	422
0,190	0,63	701	-38,8502	11,6255	5,7251	626
0,195	0,90	961	-38,8503	11,6256	5,7251	886
0,200	0,48	559	-38,8503	11,6257	5,7250	484

Tabela E.2: Penalidade *versus* probabilidade de mutação - novos resultados em um problema de minimização em caixas

N	γ_n	γ	<i>Melhor solução</i>	x_1	x_2	<i>Geração(x)</i>
25	0,42	420,12	-38,3503	11,1255	5,7250	292
50	0,00	69,25	-38,8503	11,6255	5,7250	63
75	0,15	189,38	-38,8503	11,6255	5,7250	180
100	0,02	82,50	-38,8503	11,6255	5,7250	70
125	1,00	909,63	-38,8503	11,6255	5,7250	894
150	0,00	67,75	-38,8503	11,6255	5,7250	49
175	0,00	66,88	-38,8503	11,6255	5,7250	45
200	0,02	81,00	-38,8503	11,6255	5,7250	56
225	0,04	98,13	-38,8503	11,6255	5,7250	70
250	0,02	82,25	-38,8503	11,6255	5,7250	51
275	0,03	94,38	-38,8503	11,6255	5,7250	60
300	0,00	68,50	-38,8503	11,6255	5,7250	31
325	0,03	88,63	-38,8503	11,6255	5,7250	48
350	0,04	102,75	-38,8503	11,6255	5,7250	59
375	0,04	99,88	-38,8503	11,6255	5,7250	53
400	0,05	112,00	-38,8503	11,6255	5,7250	62
425	0,05	109,13	-38,8503	11,6255	5,7250	56
450	0,05	110,25	-38,8503	11,6255	5,7250	54
475	0,05	108,38	-38,8503	11,6255	5,7250	49
500	0,06	119,50	-38,8503	11,6255	5,7250	57
525	0,07	122,63	-38,8503	11,6255	5,7250	57
550	0,08	131,75	-38,8503	11,6255	5,7250	63
575	0,10	148,88	-38,8503	11,6255	5,7250	77
600	0,09	139,00	-38,8503	11,6255	5,7250	64

Tabela E.3: Penalidade *versus* tamanho da população - novos resultados em um problema de minimização em caixas

E.2 Nova função com restrições

$$\min f(x) = 6,5x_1 - 0,5x_1^2 - x_2 - 2x_3 - 3x_4 - 2x_5 - x_6$$

sujeito a:

$$x_1 + 2x_2 + 8x_3 + x_4 + 3x_5 + 5x_6 \leq 16$$

$$-8x_1 - 4x_2 - 2x_3 + 2x_4 + 4x_5 - 1x_6 \leq -1$$

$$2x_1 + 0,5x_2 + 0,2x_3 - 3x_4 - 1x_5 - 4x_6 \leq 24$$

$$0,2x_1 + 2x_2 + 0,1x_3 - 4x_4 + 2x_5 + 2x_6 \leq 12$$

$$-0,1x_1 - 0,5x_2 + 2x_3 + 5x_4 - 5x_5 + 3x_6 \leq 3$$

onde:

$$0 \leq x_i \leq 10, i = 1, \dots, 5$$

$$0 \leq x_6 \leq 2$$

mínimo global: $f(x^*) = -11$ e $x^* = (0; 6; 0; 1; 1; 0)$

p_c	γ_n	γ	<i>Melhor solução</i>	<i>Geração(x)</i>
0,50	0,73	931,47	-10,9055	855
0,52	0,00	556,15	-10,9291	491
0,54	0,95	1041,58	-10,9922	990
0,56	0,64	882,19	-10,9507	822
0,58	0,93	1030,84	-10,9659	974
0,60	0,43	778,11	-10,9039	708
0,62	0,82	976,07	-10,9748	921
0,64	0,90	1016,26	-10,9297	950
0,66	0,90	1015,26	-10,9146	947
0,68	0,97	1056,14	-10,9459	995
0,70	0,80	968,49	-10,9089	899
0,72	0,51	817,65	-10,9238	751
0,74	0,02	564,69	-10,8945	491
0,76	0,29	707,01	-10,9818	653
0,78	0,95	1044,20	-10,9232	977
0,80	0,85	994,15	-10,9568	935
0,82	0,84	984,56	-10,9312	919
0,84	0,97	1054,43	-10,8603	970
0,86	0,77	951,20	-10,9396	887
0,88	0,84	984,60	-10,8502	898
0,90	0,88	1008,12	-10,9262	943
0,92	0,89	1012,27	-10,9510	952
0,94	0,26	687,39	-10,9448	626
0,96	1,00	1069,00	-10,8810	994
0,98	0,95	1041,40	-10,8705	959
1,00	0,35	736,45	-10,8569	652

Tabela E.4: Penalidade *versus* probabilidade de recombinação - novos resultados em uma função com restrições

p_m	γ_n	γ	<i>Melhor solução</i>	<i>Geração(x)</i>
0,000	0,10	577,62	-10,1860	84
0,005	0,93	1184,38	-10,5264	963
0,010	0,71	1022,24	-10,9534	962
0,015	0,30	726,00	-11,0000	676
0,020	0,08	565,25	-10,9890	513
0,025	0,55	907,58	-10,9640	850
0,030	0,49	861,42	-10,9784	807
0,035	0,75	1055,30	-10,9595	997
0,040	0,66	984,60	-10,8502	898
0,045	0,64	974,40	-10,8618	895
0,050	0,46	840,53	-10,8725	760
0,055	0,78	1072,96	-10,8240	986
0,060	0,80	1086,07	-10,7765	976
0,065	0,70	1013,92	-10,8428	925
0,070	0,69	1007,55	-10,8466	922
0,075	0,36	770,94	-10,8427	684
0,080	0,68	999,08	-10,7978	907
0,085	0,76	1060,38	-10,7574	947
0,090	0,51	879,18	-10,6036	731
0,095	0,86	1129,03	-10,5253	958
0,100	0,81	1092,78	-10,5045	866
0,105	0,78	1070,81	-10,8118	979
0,110	0,64	970,32	-10,7170	863
0,115	0,80	1085,65	-10,7968	989
0,120	0,92	1175,90	-10,4567	959
0,125	0,91	1165,78	-10,5127	997
0,130	0,83	1113,06	-10,6715	994
0,135	0,99	1225,96	-10,3931	946
0,140	0,64	969,87	-10,5014	777
0,145	0,93	1184,63	-10,4411	952

p_m	γ_n	γ	<i>Melhor solução</i>	<i>Geração(x)</i>
0,150	0,11	586,15	-10,3714	396
0,155	0,28	711,18	-10,6347	573
0,160	0,00	505,60	-10,5729	352
0,165	0,81	1093,91	-10,7142	980
0,170	0,78	1076,36	-10,6198	942
0,175	1,00	1233,89	-10,1761	838
0,180	0,53	888,11	-10,7472	784
0,185	0,76	1062,53	-10,4257	808
0,190	0,60	944,26	-10,4608	769
0,195	0,92	1172,58	-10,5405	999
0,200	0,69	1005,17	-10,2313	768

Tabela E.5: Penalidade *versus* probabilidade de mutação - novos resultados em uma função com restrições

N	γ_n	γ	<i>Melhor solução</i>	<i>Geração(x)</i>
25	0,24	768,74	-10,9109	742
50	0,11	708,83	-10,9563	687
75	0,72	1001,95	-10,9402	970
100	0,74	1010,42	-10,9359	972
125	0,45	870,43	-10,9132	820
150	0,81	1045,69	-10,9201	990
175	0,77	1023,44	-10,9239	963
200	0,68	984,60	-10,8502	898
225	0,53	911,84	-10,8889	831
250	0,83	1056,46	-10,9604	986
275	0,00	654,96	-10,9176	569
300	0,86	1070,57	-10,8952	972
325	0,70	991,76	-10,9507	900
350	0,79	1034,47	-10,9272	931
375	0,79	1037,37	-10,9023	921
400	0,25	772,94	-10,9424	661
425	0,81	1042,85	-10,8956	913
450	0,51	899,94	-10,9303	772
475	0,18	740,47	-10,9320	607
500	0,99	1131,04	-10,9523	996
525	0,82	1051,57	-10,8793	891
550	1,00	1136,37	-10,9566	990
575	0,81	1043,87	-10,9356	887
600	0,88	1078,29	-10,8848	901

Tabela E.6: Penalidade *versus* tamanho da população - novos resultados em uma função com restrições

E.3 Nova função multi-objetivos

$$\min(f_1(x), f_2(x))$$

onde:

$$f_1(x) = 1 - e^{-\left(\sum_{i=1}^3 \left(x_i - \frac{1}{\sqrt{3}}\right)\right)^2}$$

$$f_2(x) = 1 - e^{-\left(\sum_{i=1}^3 \left(x_i + \frac{1}{\sqrt{3}}\right)\right)^2}$$

$$-4 \leq x \leq 4$$

p_c	γ_n	γ
0,50	0,60	57,546
0,52	0,52	57,220
0,54	0,23	56,063
0,56	0,85	58,514
0,58	0,75	58,101
0,60	1,00	59,101
0,62	0,63	57,650
0,64	0,91	58,740
0,66	0,40	56,736
0,68	0,18	55,865
0,70	0,11	55,595
0,72	0,70	57,915
0,74	0,60	57,535
0,76	0,80	58,297
0,78	0,11	55,601
0,80	0,39	56,716
0,82	0,40	56,735
0,84	0,89	58,683
0,86	0,38	56,676
0,88	0,23	56,073
0,90	0,97	58,978
0,92	0,40	56,743
0,94	0,00	55,166
0,96	0,11	55,602
0,98	0,16	55,790
1,00	0,17	55,853

Tabela E.7: Penalidade *versus* probabilidade de recombinação - novos resultados em uma função multi-objetivos

p_m	γ_n	γ
0,000	0,10	56,55
0,005	0,08	56,28
0,010	0,08	56,30
0,015	0,00	55,29
0,020	0,08	56,30
0,025	0,08	56,25
0,030	0,04	55,77
0,035	0,03	55,64
0,040	0,00	55,22
0,045	0,06	56,03
0,050	0,05	55,89
0,055	0,07	56,12
0,060	0,10	56,52
0,065	0,09	56,36
0,070	0,24	58,34
0,075	0,08	56,24
0,080	0,18	57,57
0,085	0,11	56,74
0,090	0,01	55,31
0,095	0,07	56,18
0,100	0,03	55,59

p_m	γ_n	γ
0,105	0,05	55,70
0,110	0,06	56,03
0,115	0,10	56,51
0,120	0,13	56,99
0,125	0,17	57,62
0,130	0,24	58,42
0,135	0,23	58,15
0,140	0,22	58,10
0,145	0,23	58,23
0,150	0,24	58,37
0,155	0,29	59,13
0,160	0,34	59,78
0,165	0,32	59,48
0,170	0,30	59,13
0,175	0,72	64,03
0,180	1,00	68,46
0,185	0,79	65,28
0,190	0,65	63,88
0,195	0,79	65,02
0,200	0,97	68,00

Tabela E.8: Penalidade *versus* probabilidade de mutação - novos resultados em uma função multi-objetivos

N	γ_n	γ
25	0,59	73,37
50	0,00	56,32
75	0,06	58,18
100	0,16	60,82
125	0,27	64,05
150	0,36	66,67
175	0,47	69,71
200	0,57	72,76
225	0,68	75,87
250	0,78	78,77
275	0,89	81,85
600	1,00	85,05

Tabela E.9: Penalidade *versus* tamanho da população - novos resultados em uma função multi-objetivos

Referências Bibliográficas

- [1] CASTRO, Rodrigo Evangelista de. Otimização de estruturas com multi-objetivos via algoritmos genéticos. Tese – Universidade Federal do Rio de Janeiro, COPPE, 2001.
- [2] COELLO, Carlos A. Coello. Theoretical and Numerical Constraint-Handling Techniques used with Evolutionary Algorithms: A Survey of State of the Art. *Computer Methods in Applied Mechanics and Engineering*. Vol. 191, n.11-12, p.1245-1287, 2002.
- [3] COELLO, Carlos A. Coello. A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques. *Knowledge and Information System*. Vol. 1, n.3, p. 129-156, 1999.
- [4] COELLO, Carlos A. Coello; CORTÉS, Nareli Cruz. Solving Multiobjective Optimization Problems Using an Artificial Immune System. *Genetic Programming and Evolvable Machines*. Vol. 6, n.2, p. 163-190, 2005.
- [5] DARWIN, Charles R. *A Origem das Espécies*. Esboço de 1842. Clássicos Econômicos Newton, Rio de Janeiro, 1996.
- [6] DAY, Richard O.; KLEEMAN, Mark P.; LAMONT, Gary B. Multi-objective fast messy genetic algorithm solving deception problems. *Evolutionary Computation*, 2004.. Vol. 2, p. 1502-1509, 2004.
- [7] DEB, Kalyanmoy; AGRAWAL, Samir; AMRIT, Pratap; MEYARIVAN, T. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. *IEEE Transaction on evolutionary Computation*, v.6, n.2, p.182-197, 2002.

- [8] DEB, Kalyanmoy. Multi-Objective genetic Algorithms: Problem Difficulties and Construction of Test Problems. *Evolutionary Computation*, v.7(3), p.205-239, 1999.
- [9] DEB, Kalyanmoy. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. Código fonte. <http://www.mlahanas.de/Opt/Optimization.htm>, 2002.
- [10] DELBEM, Alexandre C. B.; CARVALHO, Andre Carlos P.; BRETAS, Newton G. Main Chain Representation for Evolutionary Algorithm Applied to Distribution System Reconfiguration. *IEEE Transactions on Power Systems*, v.20, n.1, 2005.
- [11] ESPINOZA, Felipe P.; MINSKER; Barbara S.; GOLDBERG, David E. Performance Evaluation and Population Reduction for a Self Adaptive Hybrid Genetic Algorithm (SAHGA). *Lecture Notes in Computer Science*. Vol. 2723, p. 922-933, 2003.
- [12] FLOUDA, C.A.; PARDALOS; P.M. *Handbook of test problems in local and global optimization* Imprenta Dordrecht, Boston, Kluwer Academic Publishers, 1999.
- [13] GOLDBERG, David E. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading, Massachusetts, 1989.
- [14] GREFENSTETTE, John J. GENETic Search Implementation System — GENESIS, Código fonte da versão 5.0. www.aic.nrl.navy.mil/galist/src/, 1990.
- [15] KIM, Jong-Hwan; MYUNG, Huyn. Evolutionary Programming Techniques for Constrained Optimization Problems. *IEEE Transactions on Evolutionary Computation*, v.1(2), 129-133, 1997.
- [16] LIU, Tung-Kuan; TSAI, Jinn-Tsong; CHOU, Hyh-Horng. Improved genetic algorithm for the job-shop scheduling problem. *The International Journal of Advanced Manufacturing Technology*, 2005.
- [17] MARTÍNEZ, José Mário; SANTOS, Sandra Augusta. Métodos Computacionais de Otimização. *Departamento de Matemática Aplicada IMECC-UNICAMP*, 1998.
- [18] METZ, E.; LENCEVICIUS, R. Efficient Instrumentation for Performance Profiling. *Proceedings of the 1st Workshop on Dynamic Analysis*, 2003, pp. 143–148.
- [19] MICHALEWICZ, Zbigniew; SCHOENAUER, Marc. Evolutionary Algorithms for Constrained Parameter Optimization Problems. *Evolutionary Computation*, 4(1), p.1-32, 1996.
- [20] MICHALEWICZ, Zbigniew. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, second editon, 1992.

- [21] MIETTINEN, Kaisa; MÄKELÄ, Marko M.; TOIVANEN, Jari. Numerical Comparison of Some Penalty Based Constraint Handling Techniques in Genetic Algorithms. *Journal of Global Optimization*, v.27, p.427-446, 2003.
- [22] OYAMA, Akira. Multidisciplinary optimization of transonic wing design based on evolutionary algorithms coupled with CFD solvers. *Proceedings of the European Congress on Computational Methods in Applied Sciences and Engineering*, ECCOMAS, CIMNE, Barcelona, 2000.
- [23] RAY, T.; TSAI, H. M. Swarm Algorithm for Single - and Multiobjective Airfoil Design Optimization. *AIAA Journal* 2004, v.42, p.366-373, 2004.
- [24] SANKAR, S. Saravana; PONNANBALAM, S. G.; RAJENDRAN, C. A multiobjective genetic algorithm for scheduling a flexible manufacturing system . *The International Journal of Advanced Manufacturing Technology*, v.22, p.229-236, 2003.
- [25] SASAKI, Daisuke; MORIKAWA, Masashi; OBAYASHI, Shigeru, NAKAHASHI, Kazuhiro. Aerodynamic Shape Optimization of Supersonic Wings by Adaptive Range Multiobjective Genetic Algorithms. *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization*, p.639-652, 2001.
- [26] WHITAKER, Roger M.; RAISANEN, Larry; HURLEY, Steve. The Marginal cost of Service Provision in Cellular Networks. *Proceedings of the 38th Hawaii International Conference on System Sciences* , 2005.
- [27] WLOCH, Krzysztof; BENTLEY, Peter J. Optimising the Performance of a Formula One Car Using a Genetic Algorithm. *Lecture Notes in Computer Science*, v. 3242, p. 702, 2004.
- [28] ZITZLER, Eckart; DEB, Kalyanmoy; THIELE, Lothar. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, v.8, n.2, p.173-195, 2000.
- [29] ZITZLER, Eckart; LAUMANN, Marco; BLEULER, Stefan. A Tutorial on Evolutionary Multiobjective Optimization. *Metaheuristics for Multiobjective Optimization*, pp. 3-37, Springer. Lecture Notes in Economics and Mathematical Systems Vol. 535, Berlin, 2004.
- [30] Genetic and Evolutionary Algorithm Toolbox for use with Matlab. <http://www.geatbx.com/docu/fcnindex.html>, 2004.