

DISSERTAÇÃO DE MESTRADO Nº 174

**ALGORITMO GENÉTICO: ESTUDO, NOVAS
TÉCNICAS E APLICAÇÕES**

Gustavo Luis Soares

DATA DA DEFESA: 11.07.97

Programa de Pós-Graduação em Engenharia Elétrica

Centro de Pesquisa e Desenvolvimento em Engenharia Elétrica

Escola de Engenharia da Universidade Federal de Minas Gerais

“Algoritmo Genético: Estudo, Novas Técnicas e Aplicações”

Autor: Gustavo Luís Soares

Orientador: Professor João Antônio de Vasconcelos

Belo Horizonte, 11 de junho de 1997

“Algoritmo Genético: Estudo, Novas Técnicas e Aplicações”

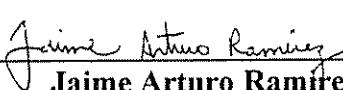
Gustavo Luís Soares

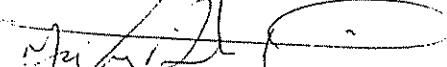
Dissertação de Mestrado submetida à banca examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Minas Gerais, como parte dos requisitos necessários à obtenção do grau de Mestre em Engenharia Elétrica.

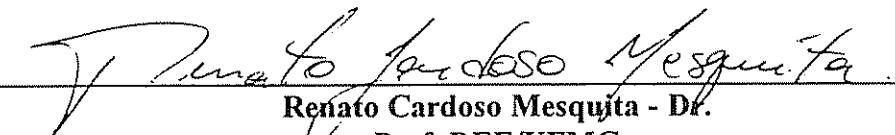
Aprovada em 11 de julho de 1997.

Por:


João Antônio de Vasconcelos - Dr.
Prof. DEE/UFMG - Orientador


Jaime Arturo Ramirez - Ph.D.
Prof. DEE/PUC-MG -


José Luiz Ribeiro Filho - Ph.D.
Prof. NC-RJ/Rede Nacional de Pesquisa - RNP


Renato Cardoso Mesquita - Dr.
Prof. DEE/UFMG


Rodney Rézende Saldanha - Dr.
Prof. DEE/UFMG

A meus pais (Geraldo e Elani) e
à minha noiva (Cristiana),
pelo carinho, incentivo e compreensão
nos momentos mais difíceis.

Agradecimentos

Ao longo deste trabalho, várias pessoas cruzaram o meu caminho e, muitas delas, de formas diferentes, contribuíram para a realização de minha dissertação. Infelizmente, nenhum tipo de agradecimento é suficientemente expressivo para demonstrar meu real reconhecimento. À todas as pessoas que me ajudaram, fica aqui o meu muito obrigado. No entanto, algumas merecem um pouco mais de destaque.

Aos familiares que sempre me apoiaram. De modo especial: ao meu irmão Gilberto Soares, pelas muitas vezes priorizou meu trabalho com relação às suas atividades; à prima Juliana Vieira, pelas tarefas em casa e; à prima Rosália Campos pela correção do texto da dissertação (versão um pouco anterior a que foi entregue à banca examinadora).

Aos amigos do CPDEE, pelo convívio e por compartilharem de meus problemas. De modo especial: aos colegas Carsten Ritter e Marcelo Barros, por correções e sugestões na dissertação; ao aluno Marcus Miranda, pelo auxílio na confecção de textos em inglês; ao aluno de iniciação Adriano Vilela e à aluna de mestrado Simone Viana, pela ajuda na verificação das aplicações; ao grande amigo André Lomônaco que, nos mais diversos momentos, contribuiu significativamente.

Ao professor Renato Mesquita, por sua prontidão no esclarecimento das mais distintas dúvidas.

Ao meu orientador professor João Antônio de Vasconcelos, pela ajuda na escolha do tema de dissertação e também pelo auxílio técnico oferecido durante a execução deste trabalho.

Ao CNPq, pelo financiamento.

Resumo

Esta dissertação tem, como tema principal, estudar os Algoritmos Genéticos (GAs), analisando numericamente seu desempenho perante algumas técnicas já existentes e outras novas. Como parte secundária, os GAs foram utilizados na resolução de problemas reais. A dissertação foi dividida em seis capítulos e três apêndices, os quais terão, a seguir, uma breve descrição.

O Capítulo 1 descreve os algoritmos de otimização em geral, a estrutura básica de um GA e como está dividida a pesquisa em relação aos GAs. Também, neste capítulo, são introduzidas algumas definições de termos técnicos pertinentes ao algoritmo. O Capítulo 2 tem como objetivo demonstrar como o GA trabalha, ou seja, como ele é guiado por regras de probabilidade para a região onde se encontra a solução. O Capítulo 3 trata teoricamente de técnicas que influem na melhoria da performance dos Algoritmos Genéticos. Vários métodos são descritos e outros novos são introduzidos. Em relação aos métodos introduzidos, propõe-se uma nova forma de executar o cruzamento para o código binário (seção 3.7), um novo critério de adaptação das probabilidades de operadores genéticos (seção 3.11) e o mecanismo de redução de intervalo (seção 3.17). Já no Capítulo 4, é feita a análise numérica de algumas técnicas e procedimentos descritos no Capítulo 3. No Capítulo 5, ilustra-se a ação dos GAs através da aplicação a problemas de otimização de corrente induzidas. Finalmente, o Capítulo 6 traz as conclusões.

Os apêndices vieram para complementar o texto. O Apêndice A mostra exemplos de três áreas de aplicação dos GAs. O Apêndice B descreve o tempo de processamento durante as fases experimentais (Capítulos 4 e 5). O Apêndice C lista softwares de GAs que são distribuídos gratuitamente na Internet.

Abstract

This master thesis has, as its main topic, the study of Genetic Algorithm (GA), verifying its performance in comparison to other established and new techniques. As a secondary theme, the optimization algorithm was used for solving real problems. The work was divided into six chapters and three appendixes, which are briefly described below.

Chapter One describes generally the existent optimization algorithms, the basic structure of the GA and the state of art on this method. In this chapter are also introduced some technical definitions related to the algorithm. Chapter Two aims at demonstrating how the GA works, i.e. how it is probabilistically guided into the region where the solution is. Chapter Three approaches theoreticaly some techniques that act on the improvement of Genetic Algorithm performance. Many methods are described and some new others are introduced. In relation of introduced methods, it is showed a new procedure to carry out the crossover to binary code (section 3.7), a new type of the adaptation of genetic operators probabilities (section 3.11) and the mecanism of reduction of domain (section 3.17). In Chapter Four, an experimental analysis of some procedures and techniques described in Chapter Three is carried out. In Chapter Five, the GA is effectivelly employed in eddy currents problem optimization. Finally, conclusions are made in Chapter Six.

The appendixes complement the text. The Appendix A shows examples of three areas of application of GAs. The Appendix B describes the processing time during the experimental stages (Chapters Four and Five). The Appendix C lists the GA freeware softwares available in the Internet.

Simbologia

a	Coeficiente angular da reta (utilizado no escalonamento Linear), parâmetro geométrico (utilizado nas aplicações 1, 2 e 3)
b	Coeficiente linear da reta (utilizado no escalonamento Linear), parâmetro geométrico (utilizado nas aplicações 1, 2 e 3)
b[i]	Vetor que contém o indivíduo binariamente codificado
ES	Execuções com sucesso
CEVI	Cruzamento Entre Vários Indivíduos
C, c	Indica valor constante
C _{max}	Constante utilizada na transformação de problemas de minimização em maximização
C _{min}	Constante utilizada para evitar a negatividade da função objetivo
C _n	Expressão matemática (utilizada nas aplicações 2 e 3)
CPV	Cruzamento Por Variável
CX	Cycle Crossover
d	Espessura da placa condutora (utilizada na aplicação 1)
d _{ij}	Função que mede a distância entre dois individuos (utilizada na função de partilha)
DF	Adaptação dinâmica Dentro da Faixa
DS	Deterministic Sampling
ERO	Edge Recombination Operator
f	Desempenho ou aptidão, freqüência
FF	Adaptação dinâmica Fora da Faixa
ff(X)	Função objetivo
f(H)	Desempenho médio das instâncias de um esquema H
f _i	Desempenho ou aptidão do indivíduo i
f _{max}	Melhor valor de desempenho numa dada geração
f _{med}	Média dos valores de desempenho numa dada geração
F _n	Expressão matemática (utilizada nas aplicações 2 e 3)
f _{s,i}	Desempenho aparente medido a partir da função de partilha (sharing function) em relação ao indivíduo i
f'	Maior valor de desempenho entre os dois indivíduos participantes de um cruzamento (utilizado na formulação da adaptação PI), valor do desempenho f depois do escalonamento
f _{max}	Valor máximo dos desempenhos após o escalonamento Linear

f_{med}	Valor médio dos desempenhos após o escalonamento Linear
$f(x)$	Função desempenho
G	Valor do intervalo do SSGA
GA	Genetic Algorithm
$G[i]$	Vetor que contém indivíduo codificado no código Gray
$g_j(x)$	Denota o conjunto de funções de restrição de desigualdade
H	Um dado esquema
$h_k(x)$	Denota o conjunto de funções de restrição de igualdade
I	Corrente, valor rms da corrente
i	Indicador de indivíduo, variável, outros
I_0	Intervalo inicial
I_n	Intervalo na n-ésima redução, uma das funções de Bessel Modificada
J	Densidade de corrente
j	Taxa de redução de intervalo, notação complexa, indicador de indivíduo
J_1	Função de Bessel do tipo 1
k	Número de alelos de um código, número de indivíduos, variável de integração
k_c	Parâmetro usado na adaptação FF relativo a p_c quando $m_{dg} \leq V_{\min}$ ou $m_{dg} \geq V_{\max}$
k_m	Parâmetro usado na adaptação FF relativo a p_m quando $m_{dg} \leq V_{\min}$ ou $m_{dg} \geq V_{\max}$
K_n	Uma das funções de Bessel Modificada
k_1	Parâmetro usado na adaptação PI acoplada à formulação de p_c quando $f \geq f_{\text{med}}$
k_2	Parâmetro usado na adaptação PI acoplada à formulação de p_m quando $f \geq f_{\text{med}}$
k_3	Parâmetro usado na adaptação PI acoplada à formulação de p_c quando $f < f_{\text{med}}$
k_4	Parâmetro usado na adaptação PI acoplada à formulação de p_m quando $f < f_{\text{med}}$
L	Comprimento total da cadeia de caracteres de um indivíduo
ℓ	Comprimento do cromossomo
ℓ_s	Comprimento de definição de dado esquema
m	Número de representantes um esquema
MES	Média de Execuções com Sucesso
MCF	Média de Cálculos de Função
$m(H,t)$	Número de representantes de um esquema H na t-ésima geração
m_{dg}	Medidor de diversidade genética, corresponde a f_{med}/f_{\max}
n	Indica quantidade ou iteração como: número de certa geração, número de variáveis, número da redução de intervalo, número de pontos de corte num cruzamento
NCF	Número de Cálculos de Função
$n_{\text{pop}}, n_{\text{pop}}$	Número de indivíduos da população
n_s	Número estimado de esquemas processados durante a geração inicial
n-pontos	Cruzamento com n pontos de corte
n_calcs	Número de cálculos de função
$o(H)$	Ordem de um esquema H

OX	Order Crossover
p_c	Probabilidade de cruzamento
$p_{c\max}$	Probabilidade de cruzamento máxima utilizada no critério de adaptação DF
$p_{c\min}$	Probabilidade de cruzamento mínima utilizada no critério de adaptação DF
p_d	Probabilidade de destruição
PI	Adaptação dinâmica Por Indivíduo
p_{inv}	Probabilidade de inversão
p_m	Probabilidade de mutação
$p_{m\max}$	Probabilidade de mutação máxima utilizada no critério de adaptação DF
$p_{m\min}$	Probabilidade de mutação mínima utilizada no critério de adaptação DF
PMX	Partially Matched Crossover
p_r	Precisão ou resolução do espaço de procura (por variável)
p_s	Probabilidade de sobrevivência com valor constante
p_s	Probabilidade de sobrevivência
p_{sel}	Probabilidade de seleção
p_{swap}	Probabilidade de mutação (utilizada em problemas de arranjos e permutação)
$P(X)$	Função de penalidade
q_n	Constante utilizada nas aplicações 2 e 3
r	Parâmetro usado nos métodos de penalidade
RGA	Replacement Genetic Algorithm
$s(d)$	Função de partilha (sharing function) na formação de nichos calculada a partir da função de distância $d_{i,j}$
SGA	Simple Genetic Algorithm
SRS	Stochastic Remainder Sampling
$SSGA$	Steady State Genetic Algorithm
$SSS1$	Simple Subpopulation Schemes - tipo 1
$SSS2$	Simple Subpopulation Schemes - tipo 2
SUS	Stochastic Universal Sampling
T	Indica o número da geração corrente (utilizada para calcular as on-line e off-line performances)
t	Número de certa geração, tempo
TSP	Travelling Salesman Problem
u_i	Corresponde ao valor médio da função de partilha (sharing function) relativa ao indivíduo i
V_{ideal}	Valor ideal de diversidade (utilizado na adaptação DF)
V_{\max}	Valor máximo de diversidade (utilizado nas adaptações DF e FF)
V_{\min}	Valor mínimo de diversidade (utilizado nas adaptações DF e FF)
X	Expressa o vetor indivíduo
x	Valor da on-line performance numa dada geração
\dot{x}	Valor da off-line performance numa dada geração

x_i	Variável i de um dado individuo
x_{\min}	Limite inferior do intervalo de procura de uma variável x_i
x_{\max}	Limite superior do intervalo de procura de uma variável x_i
z	Parâmetro geométrico utilizado na aplicação 1
1ponto	Cruzamento com 1 ponto de corte
2Pontos	Cruzamento com dois pontos de corte
α	Constante de exponenciação das power law functions
δ	Largura de faixa (utilizada para encontrar a melhor faixa de escalonamentos Sigma e Linear)
$\delta(H)$	Comprimento de definição de um esquema H
ε	Erro da norma euclidiana (utilizado para verificar se a solução está na região do ponto global), constante positiva (utilizada para transformação de um problema de minimização em outro de maximização pelo método da inversão da função objetivo)
ϕ	Ângulo (utilizado nas aplicações)
$\phi(X, r)$	Função pseudo-objetivo (avaliada para o vetor solução X e o parâmetro de penalidade r)
μ	Permeabilidade de um dado material
μ_0	Permeabilidade do vácuo
ω	Freqüência angular
σ	Condutividade, desvio padrão
ρ	Resistividade de um dado material
*	Representa os valores 0 ou 1 num esquema
%	Representa os valores 0, 1 ou * num esquema
Σf	Soma de todos os desempenhos de população para uma dada geração

Índice

1. Introdução aos Algoritmos Genéticos.....	3
1.1 Introdução	3
1.2 Técnicas de Procura	4
1.3 Analogia de Mecanismos de Seleção Natural com Sistemas Artificiais	7
1.4 Um Algoritmo Genético Simples	9
1.5 Avaliando o SGA.....	15
1.6 Pesquisa sobre GAs	16
1.7 Conclusão	17
2. Teoria do Processo Evolutivo num GA.....	19
2.1 Introdução	19
2.2 Hipótese dos Blocos de Construção	20
2.3 Teorema Fundamental dos Algoritmos Genéticos.....	21
2.4 Validação do Teorema Fundamental dos GAs.....	24
2.5 Paralelismo Implícito	26
2.6 Esquemas - Visão Geométrica	28
2.7 Conclusão	29
3. Técnicas, Mecanismos e Parâmetros Utilizados pelos GAs.....	31
3.1 Introdução	31
3.2 Transformação do Problema de Otimização na Forma Adequada a Ação dos GAs	31
3.2.1 Introduzindo Restrições.....	32
3.2.2 Evitando a Negatividade da Função Desempenho	33
3.2.3 Transformando um Problema de Minimização em Maximização	35
3.3 Sistemas de Representação	37
3.4 Mapeando Variáveis	40
3.5 Parâmetros dos GAs.....	41
3.6 A Mutação	43
3.7 Métodos de Cruzamento.....	45
3.8 A Inversão	50
3.9 Métodos de Seleção	51
3.10 Escalonando a População	52
3.11 Variação Dinâmica das Probabilidades dos Operadores Cruzamento e Mutação.....	55
3.12 Critérios de Convergência.....	59
3.13 Medindo o Desempenho dos GAs	61
3.14 Tipos Diferentes de GAs	62
3.15 Técnicas Híbridas	64
3.16 Formação de Nichos e de Subpopulações	66
3.17 Redução do Espaço de Procura	71
3.18 Conclusão	74
4. Análise Numérica de Algumas Técnicas Genéticas.....	75
4.1 Introdução	75
4.2 Melhorando o Desempenho dos GAs	76
4.2.1 Funções Teste	76
4.2.2 Testando Métodos Básicos	80
4.2.3 Outras Técnicas	93
4.3 Resultados de Alguns Pesquisadores	100
4.3.1 Resultados de Spears quanto à Formação de Nicho	101
4.3.2 Resultados de Oliver & Outros quanto aos Cruzamentos de Permutação	104

4.3.3 Análise de De Jong & Spears sobre a Interação entre Tipo de Cruzamento e Tamanho da População	105
4.4 Conclusão	108
5. Aplicações.....	110
5.1 Introdução	110
5.2 Aplicações a Problemas de Correntes Induzidas	110
5.2.1 Correntes Induzidas numa Superfície Plana Devido a uma Corrente numa Espira Paralela à Superfície	111
5.2.2 Correntes Induzidas numa Casca Cilíndrica Devido a uma Corrente num Filamento Paralelo Interno ao Cilindro	113
5.2.3 Correntes Induzidas numa Casca Cilíndrica Devido a uma Corrente num Filamento Paralelo Externo ao Cilindro.....	116
5.3 Validação dos Resultados Obtidos	119
5.4 Conclusão	122
6. Conclusão	124
7. Apêndice A: Algumas Aplicações que Utilizam GAs	129
8. Apêndice B: Tempo Computacional	131
9. Apêndice C: Softwares de GAs	133
10. Bibliografia	135

1. Introdução aos Algoritmos Genéticos

1.1 *Introdução*

Este século foi marcado por muitas transformações, sejam elas culturais, políticas, econômicas ou tecnológicas. A economia é dominada atualmente pelas multinacionais. Esse domínio se deve ao fato de elas conseguirem que seus produtos sejam competitivos em qualquer parte do planeta. A concorrência empresarial trouxe consigo a preocupação em encontrar soluções que conduzam a um melhor aproveitamento dos recursos, menores custos, e alto desempenho, entre outros. Muitas vezes, o fator experiência é suficiente na resolução destes problemas. No entanto, solucionar problemas mais complicados pode não ser tarefa fácil. Neste contexto, surge a otimização.

O conceito de otimização está bem identificado como um mecanismo de análise de decisões complexas, envolvendo seleção de valores para variáveis, com o simples objetivo de quantificar performance e medir a qualidade das decisões. A intenção é encontrar a melhor solução, respeitando, se necessário, restrições de viabilidade imposta aos parâmetros do problema.

Devido à dificuldade em se descobrir todas as interações entre variáveis e entre variáveis e restrições, somente em poucas situações se consegue representar completamente um problema real. Desta forma, a formulação para um problema real de otimização geralmente não passa de uma boa aproximação. Para que se possa construir uma aproximação razoável, itens como conhecimento teórico e experiência em modelamento são requeridos para capturar os elementos essenciais do problema. Outro item importante é o bom julgamento na interpretação dos resultados, o qual é indispensável para obter conclusões significativas. Satisfeitos esses quesitos, a otimização é considerada como ferramenta fundamental para análise de problemas reais.

Em se tratando de problemas complexos, nem sempre os métodos de procura encontram realmente a melhor solução. Existem métodos variados, e cada um deles é mais adequado a uma determinada classe de problemas. Um procedimento para escolha do método consiste em realizar um amplo estudo sobre algoritmos de otimização, verificando-se, principalmente, a característica de atingir mais vezes a solução global por número de execuções. Esse é um fator de medida da potencialidade dos algoritmos e, entre os métodos mais eficazes, encontram-se os Algoritmos Genéticos.

Os Algoritmos Genéticos são aplicados como uma técnica de procura e vem ganhando popularidade com inúmeros trabalhos de pesquisadores em aplicações diversas. Em [25] são mostradas vários trabalhos, de áreas diferentes, em que os GAs têm sido empregados com eficácia. O Apêndice A mostra algumas aplicações recentes.

1.2 Técnicas de Procura

Uma das maiores preocupações num projeto de um algoritmo de otimização é a robustez, que é o balanço entre eficiência (rapidez), eficácia (convergência para solução global) e a fácil adaptação a problemas em geral. Se um método é considerado robusto, sua solução é mais confiável e, provavelmente, o custo de reprojeto para adaptar o método a novas situações é reduzido ou até mesmo nulo.

Apesar da individualidade de cada algoritmo, existem algumas semelhanças que motivam a formação de grupos. Na literatura, os três conjuntos principais de métodos de procura são os Determinísticos, Enumerativos e Estocásticos. A Figura 1-1 mostra como se subdividem os algoritmos.

Técnicas de Procura	Métodos Determinísticos	Sem Cálculo de Derivadas	<i>Coordenadas Ciclicas</i> [1], <i>Rosenbrock</i> [1] e [41], outros.
		Com Cálculo de Derivadas	<i>Newton</i> , <i>Steepest Descent</i> [1], [9], [24] e [41]
		Direções Conjugadas	<i>BFGS</i> [1] e [41], <i>DFP</i> , <i>Fletcher & Reeves</i> [1] [9], [24], [41]
		Métodos de Penalidade	<i>Exterior</i> , <i>Interior</i> , <i>Interior Extendida</i> [1], [24] e [41]
		Outros	<i>Elipsóide</i> [3]
	Métodos Enumerativos	<i>Programação Dinâmica</i> [2]	
	Métodos Estocásticos	Algoritmos Evolucionários	<i>Estratégias Evolucionárias</i> [23]
		Outros	<i>Algoritmos Genéticos</i> [7], [11], [18] e [38] <i>Tabu</i> [19] <i>Simulated Annealing</i> [21]

Figura 1-1: Algoritmos de Otimização.

O primeiro grupo, também chamado de Métodos Baseados em Cálculo, é constituído por algoritmos que geralmente fazem uso do cálculo de derivadas e necessitam de algum tipo de informação do gradiente, seja procurando o ponto em que ele se anula, ou usando a direção para qual aponta, ou ainda fazendo aproximação de derivadas. Enfim, há diversos tipos de métodos determinísticos. Para ilustrar as iterações de algoritmos determinísticos, observe o exemplo de minimização de uma função $ff(x)$ mostrado na Figura 1-2. Nesta figura, cada marca "*" ou "o" representa o ponto ótimo daquela iteração para os métodos de *Newton* e *Steepest Descent*, respectivamente.

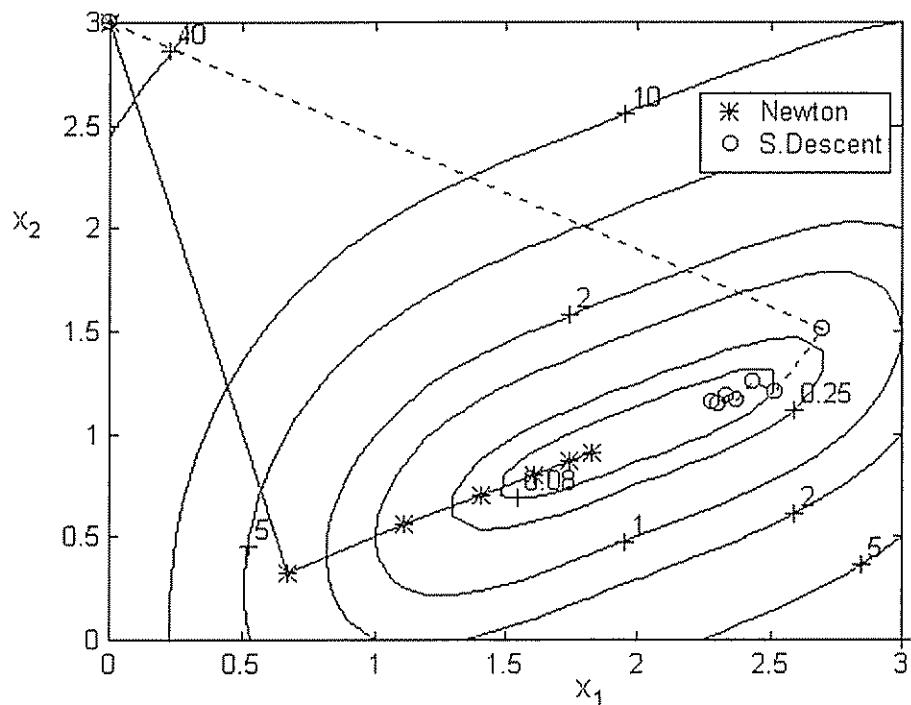


Figura 1-2: Métodos *Steepest Descent* e *Newton* aplicados à função $ff(x)=(x_1 - 2)^4 + (x_1 - 2x_2)^2$.

A procura pelo ponto ótimo, através de derivadas, usa o ponto corrente como ponto de partida para a próxima iteração. Logo, a procura é local, porque ocorre na vizinhança do ponto corrente. Assim, quando esses algoritmos encontram soluções, há grande chance de ser um ótimo local, se a função objetivo for multimodal. Além desse problema, um outro aparece quando a função a ser tratada não é contínua ou de derivação complicada. Para contornar esta última situação, obtém-se os valores das derivadas por aproximações numéricas. Verifica-se, portanto, possíveis fontes de erros no item eficácia. Por outro lado, estes métodos possuem grande rapidez e funcionam excepcionalmente bem para problemas unimodais contínuos, como o da Figura 1-3.

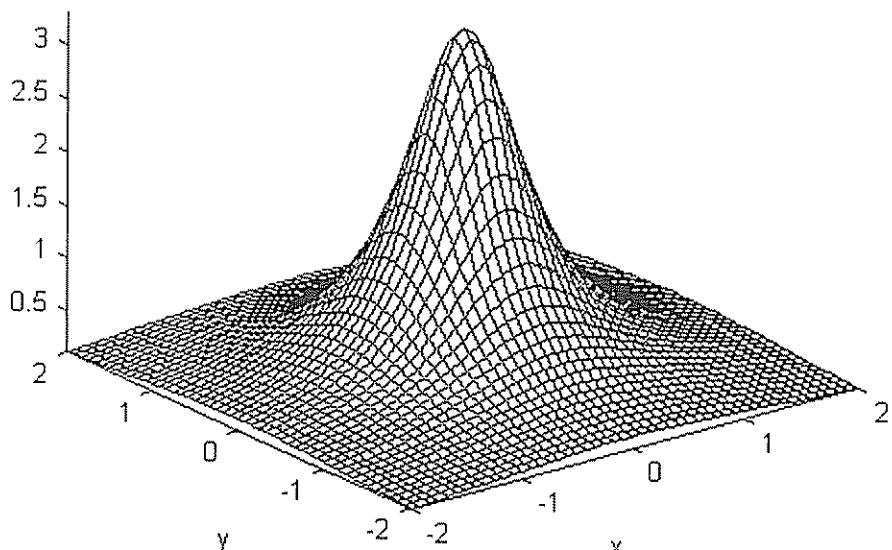


Figura 1-3: Gráfico da função $ff(x)=1/(0.3+x_1^2+x_2^2)$, ideal para métodos determinísticos.

Quanto aos Métodos Enumerativos, a idéia de procura é muito simples. Dentro de um espaço finito de procura, ou um espaço contínuo discretizado, o algoritmo verifica todas as combinações possíveis de soluções. Embora a implementação não seja complicada, o algoritmo torna-se inviável para regiões muito grandes e, consequentemente, a eficiência fica prejudicada.

Já os Métodos Estocásticos têm ganhado popularidade nos últimos anos, devido a sua robustez caracterizada principalmente pela eficácia. Eles buscam a solução a partir de regras de probabilidade. Dessa forma, a busca não é feita somente na vizinhança e, com isso, a chance de se encontrar um ótimo global aumenta. Neste grupo não há necessidade de calcular derivadas, pois os algoritmos usam apenas as informações contidas na função de otimização. Dois algoritmos estocásticos muito usados atualmente são o *Simulated Annealing* e *TABU*. Neles, a procura do ótimo é feita a partir da melhoria do melhor ponto. Enquanto esses métodos trabalham com apenas um ponto e, consequentemente, encontram apenas uma solução, os GAs trabalham com uma população de pontos simultaneamente, selecionando entre eles os melhores, podendo, assim, formar subpopulações que se distribuem não só em torno da solução global, como também em outros picos, como pode ser visto na Figura 1-4.

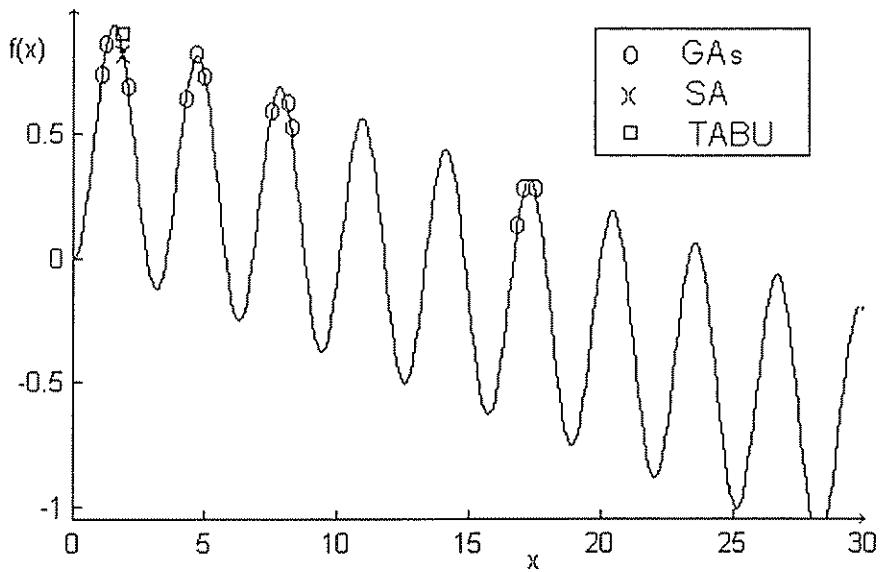


Figura 1-4: Usando subpopulações, a população fica distribuída ao longo dos picos.

Comparando os três conjuntos de algoritmos de otimização, observa-se que, se tempo computacional não for problema, os métodos enumerativos são a melhor opção, pois eles asseguram encontrar a solução global. No entanto, se a solução do problema puder ser qualquer solução factível, e ainda, rapidez de convergência e precisão da resposta forem itens importantes, então o método adequado deve pertencer ao grupo determinístico. Agora, sendo o problema complexo (por exemplo, com muitas variáveis, descontínuo ou de difícil derivação) e necessitando-se da solução global em um tempo computacional razoável, as técnicas estocásticas são as mais indicadas.

1.3 Analogia de Mecanismos de Seleção Natural com Sistemas Artificiais

Na natureza, existe um processo de seleção dos seres vivos. Numa determinada população, quando há escassez de recursos, sejam eles comida, espaço, ou outro recurso essencial, os indivíduos mais preparados para a competição dominam os mais fracos e sobrevivem. Isso acontece porque, dentre todas características imprescindíveis à competição, esses seres possuem algumas mais acentuadamente presentes que os outros. Por herança, essas características provavelmente passarão para seus descendentes, e, assim, eles terão grande chance de se saírem também vencedores. Por outro lado, fortes indivíduos podem surgir da exploração de uma outra característica ainda não desenvolvida na população. Se a natureza tentasse descobrir essas novas características através da seleção dos mais aptos e do cruzamento dentro de um mesmo grupo, certamente não teria sucesso, visto que

depois de muitas gerações, todos os membros compartilhariam praticamente o mesmo código genético. Para contornar o problema, a natureza insere material genético diferente através do processo conhecido como mutação. Se este ser que sofreu mutação, estiver tão capacitado à sobrevivência quanto os atuais, suas chances são grandes no futuro processo de seleção.

Se esse processo funciona tão bem em sistemas naturais, porque não funcionaria em sistemas artificiais? Partindo deste pressuposto, Holland [18] procurou implementar algo semelhante para sistemas artificiais. Nessa comparação, descreve-se o problema (ambiente de sobrevivência) sob forma de uma função matemática, em que as "estruturas" (indivíduos) mais fortes obterão valores mais altos de função. Assim cada indivíduo corresponde a uma possível solução. Então, trabalhando com um grupo de indivíduos simultaneamente, verifica-se a potencialidade de cada um em relação ao grupo, tentando selecionar os mais aptos para o cruzamento. Depois de se efetuar o cruzamento, cada gene de cada indivíduo estará sujeito a uma eventual ação da mutação. Logo, os GAs baseiam-se nos processos naturais de seleção, cruzamento e mutação. Esses processos são conhecidos como operadores genéticos.

Para manter a analogia, são usados nos sistemas artificiais, os termos pertinentes à genética natural nos sistemas artificiais. Dessa forma, um indivíduo ou estrutura corresponde a uma concatenação de variáveis ou *cadeias de caracteres* (cromossomos), onde cada caractere (gene), encontra-se numa dada posição (*locus*) e com seu valor determinado (alelo). Um sinônimo de indivíduo em genética natural é o genótipo e a sua estrutura decodificada é o fenótipo. Em outras palavras, o fenótipo, em sistemas artificiais, significa um conjunto de parâmetros, ou um ponto solução no espaço de procura. A partir do fenótipo, o potencial de sobrevivência pode ser obtido através da avaliação da função desempenho. Termos como reprodução, cruzamento, mutação, população, estão diretamente ligados a indivíduos. Em genética, outro conceito importante é o da epistasia. Epistasia significa a dominância um gene sobre outro gene de par diferente. Em sistemas artificiais, o termo epistasia é utilizado para definir algum tipo de não linearidade do problema tratado.

Para um contato inicial com essa analogia descrita acima, acompanhe as ilustrações a seguir.

Gene:



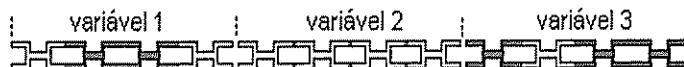
Alelos:



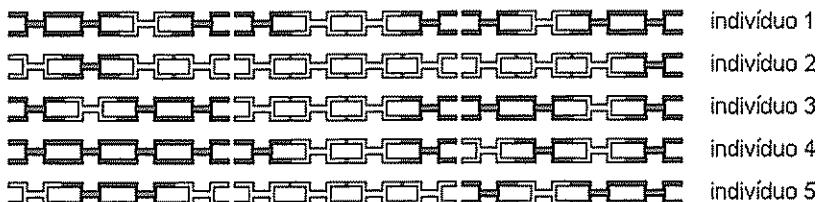
Possível cromossomo de quatro genes:



Para um problema de três variáveis, o indivíduo é a concatenação de três cromossomos, assim um possível indivíduo seria:



Os GAs trabalham com grupos de indivíduos. Então uma típica população com cinco membros poderia ser:



Finalizando, a Tabela 1-1 mostra a relação existente entre as entidades de sistemas naturais e artificiais.

Tabela 1-1
Analogia entre sistemas naturais e artificiais.

Genética Natural	Genética Artificial
gene	caractere
alelo	valor do caractere
cromossomo	cadeia de caracteres
locus	posição do gene na cadeia de caracteres
genótipo	estrutura, indivíduo
fenótipo	conjunto de parâmetros, ponto solução, estrutura decodificada
epistasia	não linearidade

1.4 Um Algoritmo Genético Simples

Na literatura, o Algoritmo Genético descrito por Goldberg em [11] é conhecido como *Simple Genetic Algorithm* ou SGA. Nele, trabalha-se com uma população fixa, cujas cadeias de caracteres estão binariamente codificadas. Após estudar o problema a ser otimizado, deve-se definir qual a quantidade de indivíduos que terá a população, a formação cromossômica do indivíduo e as probabilidades de aplicação dos operadores genéticos.

Após essa etapa de definições, o desempenho de cada indivíduo é avaliado pela função desempenho. A partir do valor do desempenho associado a cada indivíduo, o processo de seleção entra em ação e determina quem poderá reproduzir. Sobre os selecionados, atuam os operadores genéticos, cruzamento e mutação. Os novos indivíduos substituem os anteriores, terminando, assim, uma geração. O algoritmo prossegue ciclicamente a partir dessa nova população e só termina quando algum critério de convergência é alcançado. O algoritmo a seguir mostra o mecanismo completo.

```

Algoritmo Genético Simples {
    Definindo {
        função desempenho
        formação do indivíduo e tamanho da população
        probabilidade dos operadores
    }
    Inicializar população aleatória
    Enquanto não alcançar critério de convergência faça {
        avaliar os indivíduos da população
        executar seleção
        executar cruzamento e mutação
    }
}
  
```

Utilizando a mesma metodologia de Goldberg [11] para entender melhor o funcionamento do SGA, uma resolução de um exemplo foi realizada. O problema encontra-se descrito pela equação 1-1 e tem a forma mostrada na Figura 1-5.

$$\text{maximize } ff(x)=2x \quad \text{onde } 0 \leq x \leq 31 \quad (1-1)$$

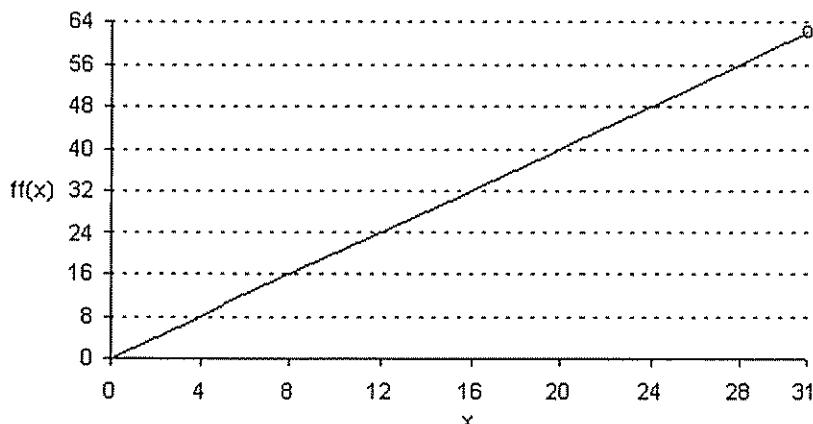


Figura 1-5: $ff(x)=2x$.

Pela Figura 1-5, sabe-se que $ff(x)$ terá o seu maior valor quando $x=31$. Juntamente com a resolução do exemplo, cada item do algoritmo, anteriormente descrito, será explicado.

Função Desempenho: A função a ser otimizada $ff(x)$ recebe o nome de função objetivo. Mas esta, muitas vezes, não é nossa função desempenho $f(x)$. Por exemplo, os GAs são definidos somente em termos de maximização e, muitas vezes, o objetivo é minimizar. Para resolver um problema como este, toma-se por exemplo, a função desempenho como o inverso da função objetivo. Dessa forma, quanto menor a função objetivo, maior o valor de retorno da função desempenho. Adaptação da função objetivo de maneira que a avaliação das variáveis facilite a ação dos GAs é um assunto que será tratado posteriormente (seção 3.2). Mas, nesse exemplo, não existe nenhum empecilho quanto à função objetivo, assim:

$$\max f(x)=2x \quad \text{função desempenho } f(x) = \text{função objetivo } ff(x)$$

Formação do indivíduo e população: Nos GAs, para que seja possível efetuar operações como cruzamento e mutação, o indivíduo deve estar codificado numa estrutura tipo cromossômica, onde cada *cromossomo* representa uma variável. Nesse exemplo, $f(x)$ tem apenas uma variável, e portanto, indivíduo e cromossomo possuem o mesmo significado. Em sistemas artificiais, cada cromossomo é codificado como uma *cadeia de caracteres* de comprimento ℓ . O código mais usado é o binário (em que o gene possui apenas os alelos 1 ou 0), pois há facilidade na implementação do algoritmo, e a busca por semelhanças entre indivíduos torna-se eficiente. Escolhido o código, resta definir o comprimento do indivíduo e o tamanho da população. Nesse exemplo, considere que cada *cadeia de caracteres* tenha comprimento $\ell=5$ e a população formada por quatro indivíduos.

Na literatura, as *cadeias de caracteres* possuem comprimentos diversificados. Em [37], por exemplo, sete funções cujas variáveis estavam binariamente codificadas, foram usadas como teste. O comprimento das *cadeias de caracteres* (cada variável) variou de 3 a 34 bits. O comprimento depende do tipo de função que se quer otimizar e também da precisão que se quer oferecer. Quanto ao tamanho da população, também não há consenso. Nota-se que há variações na quantificação desses parâmetros. Então, a título de ilustração, uma *cadeia de caracteres* curta e uma população pequena são usadas para facilitar o entendimento.

Probabilidade dos operadores Cruzamento e Mutação: Na natureza nem todos os seres cruzam e, quando o fazem, há geralmente envolvimento de um parceiro. Portanto, mantendo a analogia com sistemas naturais, deve ser feita a seleção de dois indivíduos para o cruzamento. Escolhido o par, o cruzamento só se efetivará com uma probabilidade p_c . Em relação à mutação, acontece algo semelhante. Há uma probabilidade de ocorrência sobre cada um dos genes de cada indivíduo. Como

ilustração, suponha um indivíduo composto por 10 caracteres, se a chance de ocorrer mutação for 1/100 para cada gene, cada indivíduo terá 10/100 ou 10% de probabilidade de que um gene seu sofra mutação.

Continuando o exemplo, considerando a probabilidade de cruzamento $p_c=1$, 100% dos indivíduos selecionados cruzarão. Com a probabilidade de mutação $p_m=0.01$, esse indivíduo de comprimento 5 terá 5% de chances de sofrer mutação.

População Inicial: A primeira população é aleatória, podendo-se formá-la atirando uma moeda e anotando cara=1 ou coroa=0 para cada *locus* de cada indivíduo. Assim, a primeira população poderia ser formada pelos membros:

01101, 11000, 10010 e 00101

Seleção: O SGA trabalha com um número fixo de indivíduos na população ao longo das gerações. Então, a cada nova geração, deve-se selecionar quais indivíduos terão cópias e quais desaparecerão. Este processo de duplicar indivíduos é denominado de reprodução. Após o mecanismo seleção/reprodução, todos os indivíduos estão sujeitos à ação dos operadores cruzamento e mutação. No SGA de Goldberg, o esquema de seleção, denominado Roleta, é do tipo roleta de cassinos como pode ser visto na Figura 1-6. Nesse método, cada indivíduo *i* tem uma probabilidade de seleção p_{sel} , de acordo com a sua aptidão f_i . Esse processo se inicia com a soma de todas as aptidões Σf dos indivíduos da atual população e, em seguida, calcula-se a porcentagem $f_i/\Sigma f$ para cada indivíduo *i*. A Tabela 1-2 mostra estes resultados.

Tabela 1-2
Estatística usada pelo método da roleta.

número	indivíduo	x	$f_i=f(x)$	$f_i/\Sigma f(%)$
1	01101	13	26	21.67
2	11000	24	48	40.00
3	10010	18	36	30.00
4	00101	5	10	08.33
Soma	-	-	120	100.00
Média	-	-	30	25.00
Máximo	-	-	48	40.00

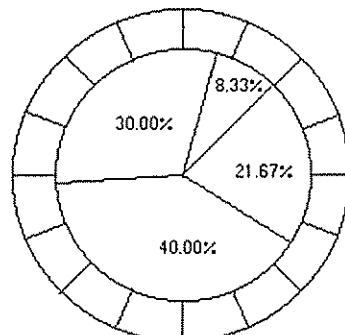


Figura 1-6: Exemplificação da roleta com as probabilidades de seleção dos 4 indivíduos.

Girando a roleta quatro vezes, poder-se-ia ter a seguinte escolha:

1º- giro: indivíduo 2

2º- giro: indivíduo 3

3º- giro: indivíduo 2

4º- giro: indivíduo 1

Assim, o indivíduo 2 teria duas cópias, os indivíduos 3 e 1 uma cópia enquanto que o indivíduo 4 desapareceria. Uma outra forma de verificar a potencialidade da roleta é calculando f_i/f_{med} , onde f_{med} é a média de todas aptidões da população. O resultado é o número provável de cópias no processo de seleção. Veja a Tabela 1-3.

Tabela 1-3
Atualização da Tabela 1-2.

número	indivíduo	x	$f_i=f(x)$	$f_i/\sum f(\%)$	f_i/f_{med}	inteiro(f_i/f_{med})
1	01101	13	26	21.67	0.87	1
2	11000	24	48	40.00	1.60	2
3	10010	18	36	30.00	1.20	1
4	00101	5	10	08.33	0.33	0
Soma	-	-	120	100.00	4.00	4
Média	-	-	30	25.00	1.00	1
Máximo	-	-	48	40.00	1.60	2

A sétima coluna mostra os valores de f_i/f_{med} arredondados. O indivíduo 4 não foi selecionado, então ele foi substituído pelo indivíduo 2, que teve mais de uma cópia. Assim:

número	indivíduo
1	01101
2	11000
3	10010
4	11000

Cruzamento: O cruzamento nada mais é do que um processo que possibilita a troca de material genético entre os indivíduos participantes e, dessa forma, fica fácil entender que ele é um poderoso mecanismo de recombinação de soluções. No SGA, o cruzamento é efetivado cortando-se, num mesmo ponto, a cadeia de

caracteres de cada um dos dois indivíduos participantes e trocando-se as partes posteriores ao corte, como mostrado na coluna "indivíduo" da Tabela 1-4.

Como nesse exemplo a probabilidade p_c é 100%, resta saber quem formam os pares de indivíduos, e entre quais *locus* ocorrerá o corte. A escolha do casal, assim como a posição de corte, é aleatória.

Após o sorteio, considere que os indivíduos 1 e 4 foram escolhidos para formarem o primeiro casal, enquanto que os indivíduos 2 e 3 o segundo. Para as duas duplas, considere ainda a posição de corte situada entre os *locus* 2 e 3. A Tabela 1-4 mostra os resultados. Depois do cruzamento, todos os novos membros da população foram submetidos à mutação.

Tabela 1-4 População Inicial após o cruzamento.			
casal	corte entre os locus	indivíduo	novos indivíduos
1	2 e 3	01101	01000
		11000	11101
2	2 e 3	11000	11010
		10010	10000

Mutação: O papel desenvolvido pela mutação é bem diferente do descrito para o cruzamento. As principais funções da mutação sobre a população são a inserção de novas características e a restauração de material genético perdido nos processos de seleção, cruzamento e até mesmo na própria mutação. A implementação da mutação, quando se usa o código binário, consiste apenas na mudança do valor do bit escolhido.

Como a probabilidade de mutação escolhida foi $p_m=0.01$ e cada indivíduo possui somente 5 bits, então há poucas chances de ocorrer mutação nessa geração.

Critério de Convergência: O próximo passo é observar se algum critério de convergência foi satisfeito. Nos GAs, o critério de término mais simples é o critério por número máximo de gerações, o qual foi adotado para este exemplo, com valor igual a 10. O algoritmo se encontra iniciando a segunda geração conforme mostrado pela Tabela 1-5, que atualiza a Tabela 1-2. Observe que o desempenho médio dos novos indivíduos que surgiram depois da ação dos operadores genéticos é superior àquele apresentado pela população inicial.

Tabela 1-5
Fim da geração inicial.

índice	indivíduo	x	$f_i = f(x)$	$f_i / \sum f(\%)$	f_i / f_{med}	inteiro(f_i / f_{med})
1	01000	8	16	10.13	0.41	0
2	11101	29	58	36.71	1.47	1
3	11010	26	52	32.91	1.32	1
4	10000	16	32	20.25	0.81	1
Soma	-	-	158	100.00	4.01	3
Média	-	-	39.5	25.00	1.00	0.75
Máximo	-	-	58	36.71	1.47	1

1.5 Avaliando o SGA

O SGA começa com uma população inicial aleatória. Através da seleção, ele tenta extrair quais indivíduos poderão contribuir mais significativamente. Este processo é executado tomando-se como base a média de aptidão de todos os componentes da população. Durante a seleção, os indivíduos com desempenho acima da média poderão ter mais cópias enquanto que aqueles com fraco desempenho podem desaparecer completamente. Entre os indivíduos selecionados da população corrente, sorteiam-se os pares para o cruzamento. O cruzamento proporcionará troca de material genético entre os pais, o que pode possibilitar a pesquisa de novos pontos no espaço de otimização. O efeito destas operações é fácil de ser observado, analisando-se o exemplo anterior, onde a média dos valores dos desempenhos da população passou de 30 para 39.5 e o valor máximo de desempenho de 48 para 58 em apenas uma geração. No decorrer das gerações, o valor do desempenho médio aproxima-se do valor máximo. A Figura 1-7 mostra um comportamento típico do desenvolvimento dos valores de desempenhos médio e máximo ao longo das gerações.

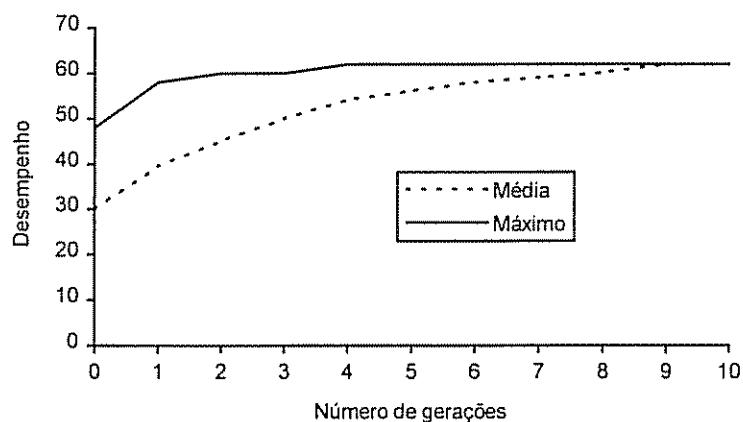


Figura 1-7: Gráfico da evolução dos valores de desempenhos médio e máximo durante dez gerações.

Após o cruzamento, cada gene de cada membro tem probabilidade $p_m=0.01$ de sofrer mutação. Como a cada geração têm-se 20 genes, e são ao todo 10 gerações (200 genes), teoricamente podem ocorrer apenas duas mutações.

1.6 Pesquisa sobre GAs

Um trabalho realizado por Goldberg & Outros [13] em 1992, consistiu em tentar coletar todas as referências bibliográficas sobre GAs. Eles conseguiram relacionar aproximadamente 1200 referências e observaram um crescimento do número de trabalhos sobre os GAs na faixa de 37% ao ano, desde 1986. O trabalho está apresentado por ordem alfabética de autores. Uma outra referência importante é a de Davis [4] em 91, onde ele divide em três, as linhas de pesquisa sobre GAs:

1. *Sistemas Classificadores*: Os *Sistemas Classificadores* são mecanismos que produzem como saída, algum tipo de informação. Essa informação será escolhida de acordo com algum critério de classificação dos dados de entrada. Os critérios de classificação nada mais são do que regras booleanas simples e, a busca pela melhor informação de saída é baseada em genética (*Genetics-Based Machine Learning - GBML*). Uma referência inicial é Goldberg [11].

2. *Análise de Performance*: Este tem sido o tema principal da maioria dos pesquisadores de GAs. O objetivo é desenvolver mecanismos que auxiliem um GA durante seu processo evolutivo. Portanto, o assunto *Análise de Performance* busca métodos de interação e manuseio de indivíduos (como tipos alternativos de cruzamento e seleção) e, entre outros aspectos, a análise comprehende também a procura por uma boa relação entre parâmetros (como tamanho da população e probabilidade dos operadores genéticos). Como são muitos os mecanismos e parâmetros associados aos GAs, ainda existem muitas questões e aspectos que necessitam de maiores pesquisas.

3. *Algoritmo Genético Paralelo*: Devido à independência das operações de seleção, cruzamento, mutação e principalmente de avaliação da função desempenho para cada possível solução (indivíduo), pode-se implementar GAs utilizando-se as vantagens da programação paralela. Segundo Ribeiro & Outros em [29], existem duas classes de *GA Paralelo*: centralizada e distribuída. Observando essas duas classes sob o ponto de vista do método de seleção, a classe centralizada possui um único mecanismo de seleção que trabalha de forma síncrona sobre a população global, a qual se encontra dividida para melhor ação dos processadores. Portanto, esta classe utiliza o

paralelismo apenas para diminuir o tempo computacional. Por outro lado, a classe distribuída utiliza cada processador para desenvolver suas subpopulações independentemente. A comunicação dos melhores indivíduos entre as subpopulações é feita de maneira assíncrona. Veja por exemplo [27].

O desempenho dos itens 1 e 3 é claramente dependente do conhecimento e estudo do item 2.

Muitos trabalhos têm surgido da aplicação de GAs como técnica de otimização (veja Apêndice A) a problemas reais. Esses trabalhos contribuem apenas na divulgação e solidificação dos GAs como algoritmo de otimização robusto e versátil. Esta dissertação se concentra no tópico *Análise de Performance*.

1.7 Conclusão

A otimização é uma ferramenta imprescindível para ajudar na resolução de problemas complexos. Vários métodos foram desenvolvidos para buscar a melhor solução. Eles se agrupam em três conjuntos: Determinísticos, Enumerativos e Estocásticos.

Dentre os três conjuntos de métodos de otimização, os estocásticos se destacam por sua característica de buscar sempre a solução global (eficácia) sem fazer uso de todo o domínio de soluções candidatas. Isso é possível por causa do uso de técnicas probabilísticas para guiar a amostragem por todo espaço viável. Dessa forma, a pesquisa pela solução global é feita em mais regiões. Dentre os métodos estocásticos, os GAs vêm ganhando espaço devido à sua robustez.

Várias diferenças podem ser notadas entre os GAs e os métodos de procura convencionais. De acordo com Goldberg [11], elas são:

1. *Manipulação de código*: Os GAs exploram a semelhança entre boas soluções através da sua codificação, enquanto que outros métodos controlam as variáveis diretamente.
2. *A procura pelo ótimo é feita a partir de uma população de pontos e não de um único ponto*: Com mais pontos para basear a pesquisa, a probabilidade de cair numa solução local se reduz.
3. *A procura é cega e feita por amostragem*: A única informação necessária é a da função de otimização, não precisando, portanto, do uso de derivadas

de qualquer ordem. A busca é guiada por soluções parciais, com a pesquisa sendo feita por amostragem e não por todo o espectro possível.

4. Os GAs *usam operadores estocásticos e não regras determinísticas*: Os operadores genéticos agem com certa probabilidade (podem ocorrer ou não), e não com regras bem definidas. Isso difere os GAs dos mecanismos de busca aleatória.

Como foi visto no decorrer do capítulo, os Algoritmos Genéticos são mecanismos de busca estocástica, direcionado pelas melhores soluções parciais, baseado nos processos de genética e seleção natural. Nessa analogia as possíveis soluções são indivíduos, e a função de otimização simula um ambiente de sobrevivência. Uma função de seleção escolhe quem vive para interagir com os operadores genéticos, resultando em novas soluções parciais.

2. Teoria do Processo Evolutivo num GA

2.1 Introdução

Através da procura por semelhanças na codificação dos indivíduos de bom desempenho, os GAs caminham na direção do ponto ótimo. Para entender o mecanismo de procura, é necessário analisar (análise feita por Goldberg [11] e Holland [18]) o indivíduo de um ponto de vista mais amplo e, para isso, insere-se na codificação binária um caractere coringa (*don't care*), o “*”, que pode ser tanto 0 ou 1. Descrito dessa forma, a estrutura formada por {0,1,*} é denominada de *esquema*. Com uma estrutura de comprimento ℓ pode-se obter, no código binário, 2^ℓ cadeias de caracteres e $(2+1)^\ell$ esquemas distintos.

Um esquema H descreve um conjunto de *cadeias de caracteres* que possuem semelhanças em certas posições. Por exemplo, suponha que uma *cadeia de caracteres* binária tenha comprimento $\ell=4$. Dessa forma, um esquema $H1$ poderia ser ****11**. Esse esquema representaria as *cadeias de caracteres* 0011, 0111, 1011 e 1111. Cada *cadeia de caracteres* possível significa uma *instância* do esquema.

Algumas características relativas às posições fixas são muito importantes na análise da performance desses esquemas. A primeira característica em relação aos esquemas é a *ordem* $o(H)$, que denota simplesmente a quantidade de posições fixas. Dessa forma, em relação ao exemplo anterior, o esquema ****11** possui *ordem* 2. A outra característica é o *comprimento de definição* $\delta(H)$, que é a distância entre as posições fixas extremas. Assim, o esquema $H2=**11*$ tem $\delta(H2)=4-3=1$, o esquema $H3=****1$ tem $\delta(H3)=0$, enquanto que $H4=1**00$ tem $\delta(H4)=4$. Um outro fator relevante ao esquema é o seu desempenho $f(H)$, sendo que este é medido pela média aritmética da aptidão entre todas as suas instâncias, para uma dada população e geração.

Definido o que é esquema, duas considerações teóricas são importantes para analisar o mecanismo de pesquisa dos GAs: A *Hipótese dos Blocos de Construção* e

o *Teorema Fundamental dos GAs*, os quais são descritos por Goldberg [11] e Holland [18]. Esses dois temas abordam a viabilidade futura de um esquema (e, por conseguinte, um conjunto de *cadeias de caracteres*), quando sujeito à ação da reprodução, do cruzamento, e da mutação. Estes assuntos serão tratados nas seções a seguir.

2.2 Hipótese dos Blocos de Construção

A descrição sobre a performance dos GAs fica mais clara quando a análise da população é feita pela perspectiva de esquemas. Esquemas de pequeno *comprimento de definição*, de baixa *ordem* e com alto desempenho são amostrados e recombinados, formando *cadeias de caracteres* de mais alto valor de aptidão. Neste sentido, reduz-se a complexidade do problema. A intenção é construir indivíduos fortes a partir dos melhores existentes, ao invés de se tentar combinar quaisquer *cadeias de caracteres*. Esquemas curtos, de baixa *ordem* e desempenho acima da média, se juntam como *blocos de construção* (*building blocks*, termo usado de agora em diante), formando uma estrutura maior.

Os esquemas vasculham regiões ao invés de um único ponto. Assim, é interessante entender a representatividade de determinado esquema no âmbito do espaço de procura. Sabendo que o número máximo de *cadeias de caracteres* diferentes no código binário é 2^t , considere então um esquema qualquer com uma posição fixa, ou seja, $o(H)=1$. O espaço atingido é $2^{t-1}/2^t$, ou seja, 50%. Com o aumento da *ordem* para 2, a região abrangida se reduz a 25% e assim por diante. Generalizando, um esquema varre $2^{t-o(H)}/2^t$ do espaço total de procura. A Figura 2-1 e a Figura 2-2, que aparecem a seguir, exemplificam a amostragem obtida por dois esquemas em relação à função exemplo do Capítulo 1 (veja a equação 1-1 e a Figura 1-5).

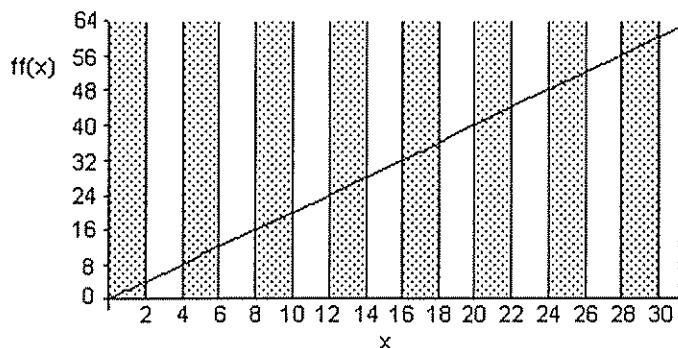
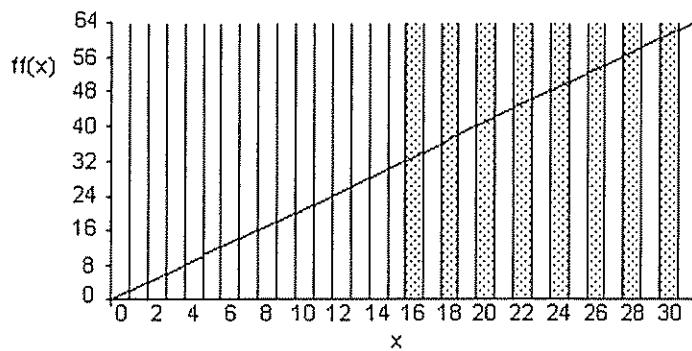


Figura 2-1: Croqui do esquema **0*, $o(H)=1$, representatividade de 50%

Figura 2-2: Croqui do esquema $1^{***}0$, $\sigma(H)=2$, representatividade de 25%

Os *building blocks* são muito importantes na fase inicial dos GAs, cujo objetivo é encontrar a região onde se encontra a solução global. Numa outra fase, a de convergência final, as *cadeias de caracteres* compartilham praticamente o mesmo código genético e, consequentemente, os mesmos esquemas de alto desempenho. Nesse ponto, a contribuição dos esquemas nos *building blocks* é bem menos significativa.

2.3 Teorema Fundamental dos Algoritmos Genéticos

O *Teorema Fundamental dos Algoritmos Genéticos* será desenvolvido apenas para o SGA, em que o método de seleção é o da Roleta, o cruzamento é feito com um ponto de corte e a mutação é feita verificando-se a probabilidade para cada alelo de cada indivíduo. Quando se tratar de outros métodos de seleção, cruzamento e mutação, a formulação do teorema pode ser conseguida de forma análoga.

Seguindo o algoritmo descrito no capítulo anterior, no princípio do algoritmo gera-se uma população aleatória, a seguir os membros desta população são submetidos à reprodução e à ação dos operadores cruzamento e mutação. Em relação aos indivíduos avaliados para participarem da próxima geração, quais são as chances destes sobreviverem ou desaparecerem? A resposta pode ser dada considerando-se um esquema H , conforme apresentado por Goldberg [11] e Holland [18]. As propriedades dos esquemas proporcionam mecanismos interessantes para discutir e classificar as semelhanças entre as *cadeias de caracteres*. Conceitos como *comprimento de definição* e *ordem* de um esquema facilitam a análise da rede de efeitos da reprodução e dos operadores genéticos sobre os *building blocks*, dentro da população.

Para começar, suponha inicialmente que, numa dada geração t , um esquema particular H , possua m representantes, essa situação será denotada por $m=m(H,t)$. Quantos representantes este esquema provavelmente existirão na geração $t+1$?

Durante a reprodução, uma *cadeia de caracteres* é copiada de acordo com a probabilidade de seleção p_{sel} proporcionada pelo seu desempenho f_i , perante todos os desempenhos encontrados na população Σf , ou seja, $p_{sel}=f_i/\Sigma f$. De maneira análoga, para esquemas, $p_{sel}=f(H)/\Sigma f$, em que $f(H)$ é o desempenho médio das instâncias do esquema. Como na geração $t+1$, n_{pop} (número total de indivíduos da população) indivíduos são selecionados, espera-se ter $m(H,t+1)=m(H,t)*n_{pop}*f(H)/\Sigma f$ representantes de H . Sabendo que a média de desempenhos da população f_{med} é definida como $f_{med}=\Sigma f/n_{pop}$, o valor $m(H,t+1)$ pode ser obtido da equação 2-1.

$$m(H,t+1) = m(H,t) \frac{f(H)}{f_{med}} \quad (2-1)$$

Analizando a equação 2-1, pode-se perguntar se é possível estimar o crescimento ou decaimento de um esquema particular após um certo número de gerações. Para responder a essa questão, considere, por exemplo, que um esquema H possua desempenho $f(H)=f_{med}+c*f_{med}$, onde c é uma constante durante t gerações. Assim, a equação pode ser rescrita como:

$$m(H,t+1) = m(H,t) \frac{f_{med} + c * f_{med}}{f_{med}} = m(H,t) * (1 + c) \quad (2-2)$$

Começando de $t=0$ e observando o que acontece com a equação 2-2 na geração t , a ação da reprodução sobre os esquemas tem forma exponencial, como está mostrado na equação 2-3.

$$m(H,t) = m(H,0) * (1 + c)^t \quad (2-3)$$

Da equação 2-1 nota-se que o número de esquemas aumentará dependendo da razão entre os desempenhos médios de suas instâncias e da população. Lembre-se de que somente o efeito da reprodução foi levado em consideração.

A reprodução simplesmente determina quem vive (e com quantas cópias) e quem desaparece. Somente com a reprodução não há exploração de novas regiões, já que nenhum indivíduo novo apareceu. Também não há troca de informação entre os indivíduos, visto que a reprodução não proporciona interação entre os mesmos. O operador que promove interação entre os membros da população é o cruzamento. Para entender como é afetado um esquema pelo cruzamento, suponha uma *cadeia de caracteres* S , com $\ell=10$, e dois esquemas $H1$ e $H2$, ambos com *ordem* igual a dois, mas com *comprimentos de definição* $\delta(H1)=7$ e $\delta(H2)=1$, como mostrado a seguir:

$$\begin{array}{l}
 S = 1 \ 0 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 0 \\
 H1 = * \ * \ 1 \ * \ * \ * \ * \ * \ 0 \\
 H2 = * \ 0 \ 1 \ * \ * \ * \ * \ * \ *
 \end{array}$$

O ponto de corte no cruzamento é simplesmente uma escolha aleatória entre 1 e $\ell-1$. É claro que, se a *cadeia de caracteres* S for escolhida para o cruzamento, o esquema H2 tem maiores chances de sobreviver ao cruzamento, pois esse esquema só será desfeito se o corte acontecer entre as posições 2 e 3, enquanto que H1 pode ser destruída se o corte ocorrer entre as posições 3 e 10. Logo, a probabilidade de destruição p_d de H2 é 1/9, enquanto que H1 tem $p_d=7/9$. Generalizando, $p_d=\delta(H)/(\ell-1)$. É interessante trabalhar com a probabilidade de sobrevivência p_s ao invés de p_d , assim $p_s=1-p_d$ ou $p_s=1-\delta(H)/(\ell-1)$. Mas, na realidade, o cruzamento só ocorre a uma dada probabilidade p_c (com valores entre 0 e 1), que deve ser levada em conta. Se o cruzamento é feito entre duas *cadeias de caracteres* representantes do mesmo esquema H, esse esquema será jamais destruído. Matematicamente, $p_d=p_c*\delta(H)/(\ell-1)$ e, como o cruzamento pode ser realizado entre representantes de mesmo esquema, $p_d\leq p_c*\delta(H)/(\ell-1)$, o que conduz a $p_s\geq 1-p_c*\delta(H)/(\ell-1)$.

Como a reprodução e cruzamento são eventos independentes, a ação conjunta dos dois processos é obtida simplesmente multiplicando-se os dois eventos, desta forma:

$$m(H, t+1) \geq m(H, t) \frac{f(H)}{f_{\text{med}}} \left[1 - p_c \frac{\delta(H)}{(\ell-1)} \right] \quad (2-4)$$

Essa equação mostra claramente que, se um esquema tem desempenho acima do desempenho médio da população e pequeno *comprimento de definição*, provavelmente ele proliferará. Na reprodução, os indivíduos mais aptos têm maior probabilidade de serem selecionados, enquanto que no cruzamento realiza-se a troca de material genético entre os membros da população. Com o passar das gerações, os indivíduos poderão conter o mesmo código genético, o que impossibilitaria a exploração de novas regiões ou mesmo restaurar material genético perdido em alguma operação. A mutação surge para cobrir essa lacuna e será tratada a seguir.

O último operador a ser considerado é a mutação. Do capítulo anterior, viu-se que cada bit da *cadeia de caracteres* de cada indivíduo da população está sujeito à mutação de acordo com uma probabilidade p_m . Cada alelo simples tem

probabilidade $p_s=1-p_m$, e como cada mutação é estatisticamente independente, a probabilidade de sobrevivência p_s para um particular esquema será o produto das probabilidades para cada posição fixa, ou seja a probabilidade de sobrevivência de um esquema com *ordem* $o(H)$ é $p_s=(1-p_m)^{o(H)}$. Para $p_m << 1$, o que freqüentemente ocorre, pode-se aproximar a relação anterior para $p_s=1-o(H)*p_m$. Assim, tem-se a equação 2-5:

$$m(H, t+1) \geq m(H, t) \frac{f(H)}{f_{\text{med}}} \left[1 - p_c \frac{\delta(H)}{(\ell - 1)} - p_m * o(H) \right] \quad (2-5)$$

Com esse último resultado, pode-se finalmente formular os efeitos da seleção/reprodução, cruzamento e mutação sobre o número de esquemas esperados para próxima geração.

Agora a hipótese *building blocks* está concretizada, ou seja, esquemas de desempenho alto, *comprimento de definição* pequeno e *ordem* baixa são os melhores candidatos a perseverarem. Essa formulação (equação 2-5) recebe o nome especial de *Teorema de Esquema* ou *Teorema Fundamental dos GAs*. Para se verificar a validade, analisaremos o exemplo do capítulo anterior na próxima seção.

2.4 Validação do Teorema Fundamental dos GAs

A formulação matemática do *Teorema Fundamental dos Algoritmos Genéticos* demonstra claramente quais esquemas terão mais chances de passar ou não para a próxima geração. Com o intuito de validar a equação, o exemplo do Capítulo 1 será revisado do ponto de vista de esquemas. Serão observados 5 esquemas. São eles:

H1	=	1	*	*	*	*
H2	=	1	*	0	*	*
H3	=	*	1	*	*	0
H4	=	*	*	*	0	1
H5	=	*	1	*	0	*

Para facilitar a verificação do desempenho desses esquemas em relação aos membros da população, a Tabela 1-3 foi reproduzida e revisada, acrescentando-se a análise dos esquemas escolhidos, tornando-se a Tabela 2-1 que vem a seguir.

Tabela 2-1
Análise da performance dos esquemas H1-H5 na geração inicial ($t=0$).

número	indivíduo	x	$f=f(x)$	$f/\sum f$ (%)	f/f_{med}	inteiro(f/f_{med})
1	01101	13	26	21.67	0.87	1
2	11000	24	48	40.00	1.60	2
3	10010	18	36	30.00	1.20	1
4	00101	5	10	08.33	0.33	0
Soma	-	-	120	100.00	4.00	4
Média	-	-	30	25.00	1.00	1
Máximo	-	-	48	40.00	1.60	2
Esquema	$m(H, t=0)$		$f(H)$	$\delta(H)$	$o(H)$	
1***	2		42	0	1	
1*0**	2		42	2	2	
*1**0	1		48	3	2	
***01	2		18	1	2	
*1*0*	2		37	2	2	

No exemplo, $p_c=1$, $p_m=0.01$, $f_{med}=30$ e $\ell=5$. Assim, o *Teorema Fundamental dos Algoritmos Genéticos* descrito na equação 2-5, torna-se:

$$m(H, t+1) = m(H, t) \frac{f(H)}{30} \left[1 - \frac{\delta(H)}{4} - 0.01 * o(H) \right] \quad (2-6)$$

Substituindo os valores de $m(H, t)$, $f(H)$, $\delta(H)$ e $o(H)$ da Tabela 2-1, obtém-se as estimativas de quantidade de cada esquema para a próxima geração. O resultado é acoplado à Tabela 1-5, onde já se encontravam as *cadeias de caracteres* (indivíduos) para a segunda geração, formando a Tabela 2-2.

Observando a Tabela 2-2, nota-se que o esquema H1 possui alto desempenho médio, tem comprimento $\delta(H)=0$ e $o(H)=1$, ou seja, todas as condições de aumentar o número de cópias. O esquema H2 possui o mesmo desempenho de H1, mas a probabilidade de que haja separação de suas posições fixas é $\delta(H2)/(\ell-1)*100=50\%$. Nesse exemplo, durante o cruzamento, o corte ocorreu entre o *locus* 2 e 3, separando o esquema, mas as *cadeias de caracteres* escolhidas para formarem casal para o cruzamento eram representantes desse esquema, evitando-se a sua destruição, mantendo, portanto, as características originais. O esquema H3 obteve duas cópias devido à reprodução da *cadeia de caracteres* (indivíduo) 2. Quando foi conduzida ao cruzamento, rompeu-se, mas, por sorte, o mesmo cruzamento que a desfez a reconstruiu. Por isso, vale lembrar que o cruzamento não adiciona características novas, ele simplesmente explora as existentes. O esquema H4 tem baixo desempenho médio, mas como ponto positivo tem $\delta(H4)=1$ e isso proporcionou a sua sobrevivência para a próxima geração. Enfim, o esquema H5, com desempenho mediano e comprimento de definição longo, obteve cópias pelo mesmo motivo que no caso H2.

Tabela 2-2
Avaliação da estimativa de alguns esquemas para a geração t=1.

número	individuo	x	$f_i = f(x)$	$f_i / \sum f_i (\%)$	f_i / f_{med}	inteiro(f_i / f_{med})
1	01000	8	16	10.13	0.41	0
2	11101	29	58	36.71	1.47	1
3	11010	26	52	32.91	1.32	1
4	10000	16	32	20.25	0.81	1
Soma	-	-	158	100.00	4.01	3
Média	-	-	39.5	25.00	1.00	0.75
Máximo	-	-	58	36.71	1.47	1
Esquema	Individuos que contiam H (t=0)	individuos que contêm H (t=1)		$m(H, t=1)$	quantidade real em t=1	
1***	2;3	2;3;4		2.77	3	
1*0**	2;3	3;4		1.34	2	
*1**0	2	1;3		0.37	2	
***01	1;4	2		0.88	1	
*1*0*	1;2	1;2		1.18	2	

Revisando o exemplo do Capítulo 1, constatou-se a validade do *Teorema Fundamental dos GAs*. A princípio, o erro entre os valores esperados e os realmente obtidos parece ser grande. Essas diferenças entre valores estimados e valores reais diminuirão quando a amostragem for significativa, ou seja, população grande e longo comprimento ℓ das *cadeias de caracteres*.

2.5 Paralelismo Implícito

O GA, a cada geração, processa n_{pop} indivíduos, mas o que faz dele um poderoso método de procura é que, na análise dos n_{pop} indivíduos, ele verifica paralelamente n_{pop}^3 esquemas. Esse resultado importante recebeu o nome de *paralelismo implícito*. Esta estimativa de n_{pop}^3 esquemas por n_{pop} indivíduos foi desenvolvida por Goldberg em [11] e [12] e será reproduzida nesta seção.

É interessante iniciar a estimativa do número de esquemas processados em paralelo conhecendo a capacidade de formação de indivíduos e esquemas de qualquer código. Para qualquer tipo de codificação, seja k o número de alelos diferentes e ℓ o comprimento da *cadeia de caracteres*, existem pois k' possibilidades diferentes de formação de *cadeias de caracteres* e $(k+1)'$ de esquemas. Cada *cadeia de caracteres* específica é representante de k' esquemas diferentes. Por exemplo, a *cadeia de caracteres* 111 é representante de todos os esquemas possíveis formados pelos caracteres 1 e *. Então, numa população de n_{pop} membros existem de k' a $n_{pop} * k'$ esquemas competindo.

Considere uma população de n_{pop} *cadeias de caracteres* binárias de comprimento ℓ . Considere também somente os esquemas que sobrevivem com uma probabilidade maior que ps , onde ps é uma constante. Logo, admite-se somente

esquemas com uma taxa de destruição $p_d < 1 - ps$. Assumindo a operação cruzamento com um único ponto de corte e uma pequena taxa de mutação, trata-se somente daqueles esquemas com comprimento $\ell_s < p_d(\ell-1)+1$. Isso pode ser facilmente verificado considerando-se, inicialmente, a probabilidade de cruzamento igual à unidade. Assim, a probabilidade de destruição de um esquema devido à ação do cruzamento é $p_d < \delta(H)/(\ell-1)$. Sendo $\delta(H) = \ell_s - 1$, tem-se que valores de ℓ_s , tais que, $\ell_s > (\ell-1)p_d + 1$ aumentam a probabilidade de destruição. Logo, para se obter $p_d < 1 - ps$, deve-se levar em conta somente os esquemas cujo comprimento seja $\ell_s < p_d(\ell-1)+1$.

Com o comprimento de um esquema particular, consegue-se estimar um limite inferior para o número de esquemas processado por uma população inicial de indivíduos gerada aleatoriamente. Para essa estimativa, primeiro conta-se o número de esquemas de comprimento ℓ_s ou menor. A seguir, multiplica-se esse número pelo tamanho da população.

Para o primeiro passo, suponha que se deseja contar todos os esquemas de comprimento de definição $\ell_s=5$ ou menor no seguinte indivíduo de comprimento $\ell=10$.

1 0 1 1 1 0 0 0 1 0

Primeiramente, calcula-se o número de esquemas na primeira célula considerando o quinto bit fixo, isto é, deseja-se todos esquemas da forma:

1 0 1 1 1 0 0 0 1 0 = = = > % % % % 1 * * * *

onde * é um símbolo para representar em seu lugar os caracteres 0 ou 1. O símbolo % representa valores fixos 0 ou 1, ou *. Evidentemente há $2^{(\ell_s-1)}$ diferentes representantes destes esquemas porque $\ell_s-1=4$ posições podem ser bits fixos ou *. Para contar o número total, basta deslizar a região selecionada uma posição por vez.

1 0 1 1 1 0 0 0 1 0

O número total de deslizamentos é $\ell - \ell_s + 1$ e, portanto, o número de esquemas de comprimento igual ou menor a ℓ_s numa única cadeia de caracteres é $(2^{\ell_s-1})^*(\ell - \ell_s + 1)$.

O segundo passo dessa estimativa é obtido, multiplicando-se $(2^{\ell_s-1})^*(\ell - \ell_s + 1)$ pelo número de indivíduos da população n_{pop} , assim, obtém-se a quantia $n_{pop} * (2^{\ell_s-1})^*(\ell - \ell_s + 1)$. Este resultado é, sem dúvida, superestimado, tendo em vista que certamente haverá muitas duplicatas de esquemas de baixa ordem em populações grandes.

Então, para refinar este resultado, considere $n_{\text{pop}}=2^{\ell_s/2}$. O número de esquemas é distribuído binomialmente, e como isso, conclui-se que, de todos esquemas com comprimento menor ou igual a ℓ_s , metade possui *ordem* entre $\ell_s/2$ e ℓ_s e metade com *ordem* menor que $\ell_s/2$. Contando somente os esquemas de *ordem* mais elevada, estima-se um limite inferior do número de esquemas como segue:

$$n_s \geq n_{\text{pop}} * 2^{\ell_s-2} (\ell - \ell_s + 1) \quad (2-7)$$

Isso difere do valor superestimado por um fator de 1/2. Além disto, a restrição do tamanho da população para o valor particular $2^{\ell_s/2}$ resulta na seguinte expressão:

$$n_s \geq \frac{n_{\text{pop}}^3 (\ell - \ell_s + 1)}{4} \quad (2-8)$$

Desde que $n_s = \text{constante} * n_{\text{pop}}^3$, deduz-se que o número de esquemas é proporcional ao cubo do tamanho da população. Portanto, na geração inicial, quando processam-se os n_{pop} indivíduos, processam-se paralelamente n_{pop}^3 esquemas diferentes.

2.6 Esquemas - Visão Geométrica

Mesmo com a equação do *Teorema Fundamental dos GAs* aplicada para o exemplo do Capítulo 1, e com a explicação dada na seção anterior, ainda não é fácil enxergar o *paralelismo implícito* com os qual os GAs trabalham. Uma outra forma de verificar este paralelismo é uma visão geométrica de esquemas num espaço de procura. Dessa forma, considera-se esquemas de comprimento $\ell=3$ e um espaço de procura tridimensional como mostrado na Figura 2-3.

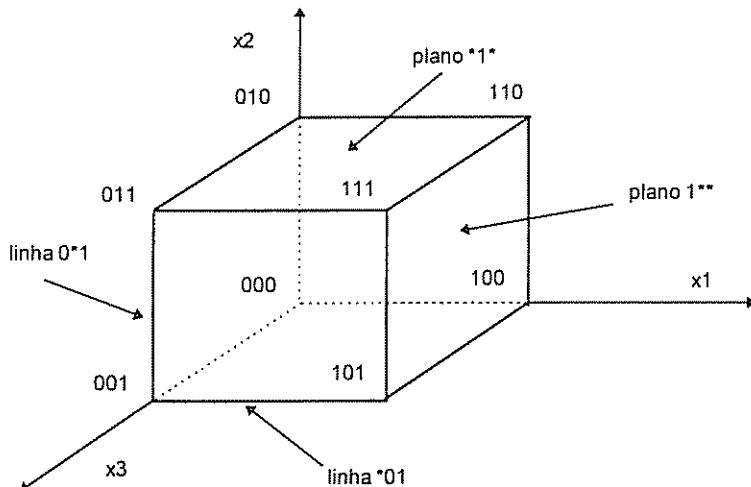


Figura 2-3: Visão geométrica de hiperplanos tridimensionais.

Esquemas com *ordem* 3 definem pontos específicos no espaço, esquemas com *ordem* 2 definem linhas de possíveis soluções. Os planos no espaço são esquemas de *ordem* 1, enquanto que todo espaço é coberto por esquema de *ordem* 0, ou seja, ***.

Infelizmente, é impossível ter esta visão geométrica para espaços superiores a 3D. Mas, generalizando para n dimensões, estes esquemas formam hiperplanos. É interessante imaginar os GAs, cortando hiperplanos na procura do melhor desempenho.

2.7 Conclusão

Analizando a *cadeia de caracteres* sob o ponto de vista de esquemas, fica fácil entender como os GAs fazem a busca do ponto ótimo explorando as semelhanças entre os diversos esquemas. A Figura 2-3 mostra graficamente como as similaridades conduzem às regiões mais diversas.

Usando o cruzamento com um único ponto de corte, a mutação com um pequeno valor de probabilidade e o método da Roleta como mecanismo de seleção, o *Teorema Fundamental dos GAs*, representado pela equação 2-5, foi desenvolvido sob a perspectiva de esquemas. Da equação 2-5 observou-se que esquemas curtos, de baixa *ordem* e alto valor de aptidão, são fortes soluções parciais do problema e, com isso, terão grandes chances de prosseguirem nas gerações seguintes. A partir da combinação dessas soluções, os GAs descobrem novas soluções (*building blocks*). Portanto, os GAs trabalham com “montagem” de soluções parciais e, assim, reduz-se a complexidade de problema como um todo.

Os Algoritmos Genéticos avaliam o desempenho de todos os n_{pop} membros da população a cada geração, mas analisam implicitamente n_{pop}^3 esquemas (*parallelismo implícito*).

3. Técnicas, Mecanismos e Parâmetros Utilizados pelos GAs

3.1 *Introdução*

O GA apresentado até agora (SGA), com o método de seleção e apenas dois operadores (cruzamento e mutação) e descrito por Goldberg [11], teve um papel muito importante na introdução dos GAs como ferramentas de otimização. Mas, com o passar do tempo, técnicas novas (como técnicas de nicho) e alternativas (como outros métodos de seleção) apareceram. Assim, com o objetivo de descrever essas técnicas, é feito neste capítulo, uma revisão bibliográfica de temas sobre os GAs. Também são apresentadas as contribuições desta dissertação. Os itens tratados vão desde codificação da estrutura cromossômica até técnicas mais avançadas, como formação de subpopulações e redução de intervalo.

3.2 *Transformação do Problema de Otimização na Forma Adequada a Ação dos GAs*

Para se resolver um problema real através da otimização, deve-se inicialmente modelá-lo matematicamente através de equações que contenham seus parâmetros (função objetivo). Mas, muitas vezes, a função objetivo apresenta problemas que dificultam ou até mesmo impossibilitam a execução dos GAs. Por exemplo, sabe-se que os GAs trabalham tentando propiciar o desenvolvimento de indivíduos, cujo valor de aptidão esteja acima da média, até chegar àqueles de mais alto desempenho. Portanto, é fácil perceber que os GAs trabalham em termos de maximização. No entanto, muitas vezes, o objetivo é minimizar uma função. Em casos como esse, é necessário transformar a função objetivo em uma outra função (função desempenho), onde esse problema esteja corrigido. Outros cuidados com a função objetivo incluem evitar que ela retorne valores negativos, e também respeitar

as restrições impostas pelo problema de otimização. Esses dois itens, juntamente com a transformação de um problema de minimização em maximização, são explicados com detalhes adiante. Os métodos de mapeamento da função objetivo para função desempenho devem ser aplicados com cuidado, para que a função objetivo não perca as características de sua solução ótima. Inicialmente foram tratados os problemas de restrições à função objetivo.

3.2.1 Introduzindo Restrições

Um tipo de restrição muito comum é a limitação da variável a um determinado intervalo. Como essa restrição é imprescindível à delimitação da região de procura para os GAs, denominam-se de irrestritos, os problemas que contêm apenas este tipo restrição. Além da restrição anterior, há, no entanto, problemas em que algumas outras condições devem ser satisfeitas para que a solução tenha validade. Nesses casos, para contornar a situação, um procedimento seria executar a avaliação de cada um dos integrantes da população e, se algum não for viável, então procura-se outro para substitui-lo e assim por diante. Este procedimento de resolver problemas com restrições é muito caro computacionalmente, agravando-se nas situações, em que achar um ponto viável é tão difícil quanto encontrar a melhor solução. Assim, uma segunda e, mais adequada maneira, seria tentar acoplar as funções de restrição à função objetivo, transformando um problema restrito em irrestrito, como por exemplo, as transformações dos métodos de penalidade.

Nos métodos de penalidade, um problema com restrições é transformado em outro sem restrições pela associação de uma *função de penalidade*, que contém todas as restrições violadas. Equacionadas, as restrições podem ser de igualdade e desigualdade, resultando numa *função pseudo-objetivo*. Para exemplificar, veja o problema de minimização com restrição a seguir:

$$\begin{aligned} \min \quad & ff(X) \\ \text{sujeito a:} \quad & \\ & g_j(X) \leq 0 \quad j = 1, \dots, m \\ & h_k(X) = 0 \quad k = 1, \dots, l \end{aligned} \tag{3-1}$$

onde $ff(X)$ é a função objetivo e, $g_j(X)$ e $h_k(X)$ representam todas as funções de restrições de desigualdade e igualdade respectivamente. O problema restrito acima pode ser transformado no seguinte problema de minimização irrestrita:

$$\min \phi(X, r) = ff(X) + r * P(X) \tag{3-2}$$

onde $\phi(X, r)$ denota a função pseudo-objetivo, $P(X)$ é a função de penalidade e r é um parâmetro de penalidade.

Há três definições tradicionais para $P(X)$. A primeira penaliza a função objetivo quando as restrições são violadas, esse é o *Método de Penalidade Exterior* [1], [24] e [41] (estas referências tratam também dos métodos de penalidade apresentados a seguir). A segunda, conhecida como *Método de Penalidade interior*, penaliza a função objetivo de forma que as restrições nunca sejam violadas. Cada um destes algoritmos tem vantagens, e estas foram acopladas num terceiro método, denominado de *Método de Penalidade Interior Estendida*.

Não é objetivo deste trabalho descrever detalhes destas funções de penalidade. Sugere-se a consulta às referências e [1] e [24] e [41]. Mas, a título de ilustração, a função de penalidade exterior tem a forma:

$$P(X) = \sum_{j=1}^m \{\max[0, g_j(X)]\}^2 + \sum_{k=1}^l [h_k(X)]^2 \quad (3-3)$$

Sendo que o primeiro somatório significa que será tomado o maior valor entre 0 e $g_j(X)$.

3.2.2 Evitando a Negatividade da Função Desempenho

Os GAs verificam a performance dos integrantes da população submetendo-os a uma função de desempenho. O processo de seleção utiliza os resultados dessa função para compor a população, que ficará sujeita à ação dos operadores genéticos. Como a grande maioria dos métodos de seleção faz uso de dados estatísticos, dependentes do valor do desempenho f_i de cada indivíduo i , deve-se evitar que a função, que se quer otimizar, retorne valores menores que zero, pois isso implicaria valores negativos para probabilidade, o que não existe.

Uma maneira de garantir a não negatividade da função desempenho $f(x)$ seria simplesmente tomar $f(x)$ como o módulo da função pseudo-objetivo $\phi(X, r)$ (que, nessa subseção, considerou-se que já encontra-se descrita sob o formato de maximização), como é mostrado a seguir:

$$f(x) = |\phi(X, r)| \quad (3-4)$$

Se $\phi(X, r)$ for um tipo de função desconhecida, não se aconselha o uso desse artifício, pois pode-se ocasionar um erro grave, onde um mínimo acentuado seria um máximo falso. A Figura 3-1 a seguir ilustra o problema.

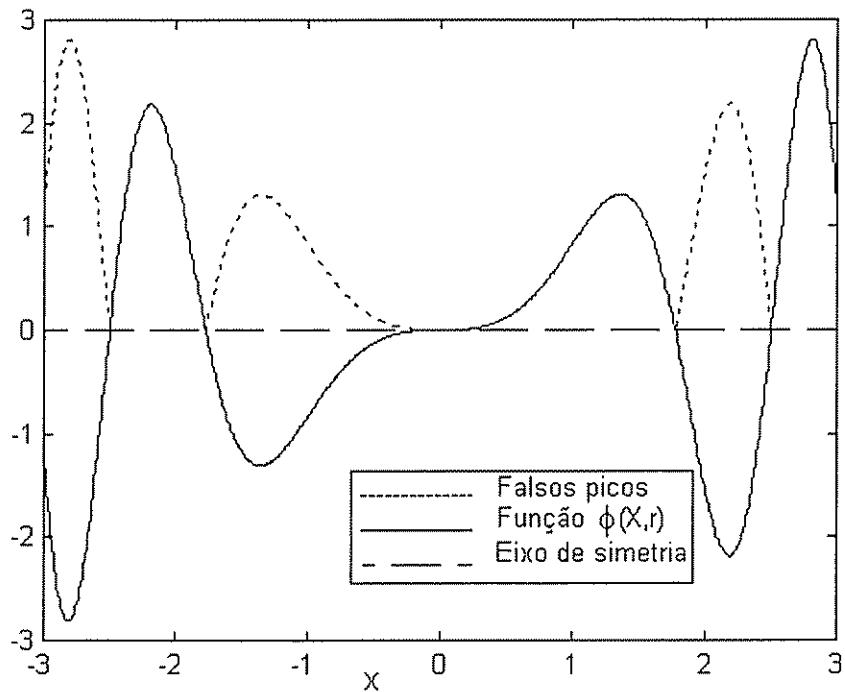


Figura 3-1: Função $\phi(X,r)=x \cdot \sin(x^2)$ no intervalo $[-3,3]$. O módulo de $\phi(X,r)$ causa falsos picos.

Outra forma mais segura de evitar a negatividade da função de desempenho é a seguinte transformação:

$$\begin{cases} f(X) = \phi(X,r) + C_{\min} & \text{se } \phi(X,r) + C_{\min} > 0 \\ f(X) = 0 & \text{se } \phi(X,r) + C_{\min} \leq 0 \end{cases} \quad (3-5)$$

O parâmetro C_{\min} pode ser uma constante, como também pode variar ao longo das gerações dependendo dos valores da média e desvio padrão da população. Inicialmente, C_{\min} é escolhido após a verificação do retorno de $\phi(X,r)$ para vários pontos testes. A Figura 3-2 mostra o gráfico da mesma função anterior adaptada à esse procedimento e adotando $C_{\min}=4$.

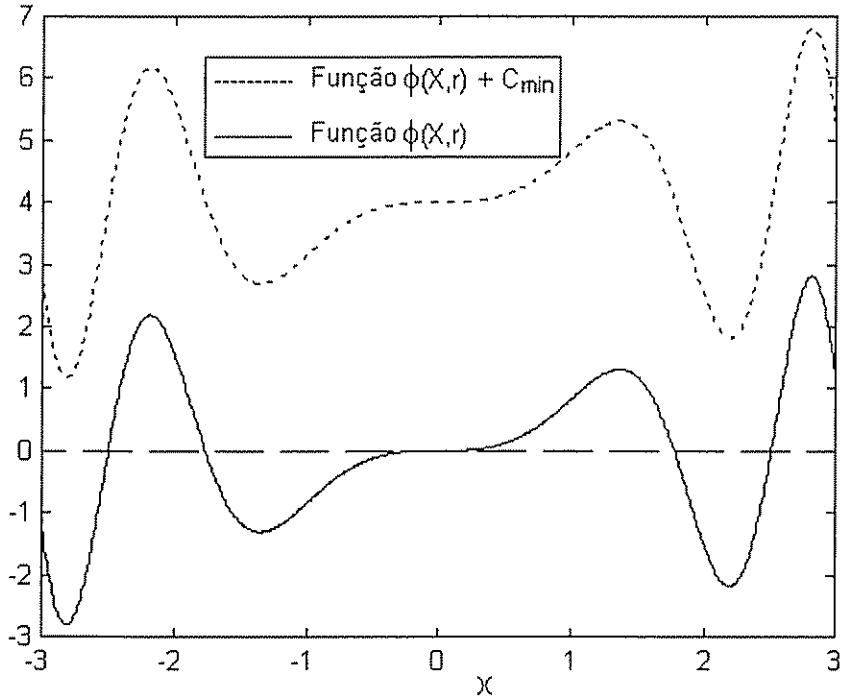


Figura 3-2: Acréscimo de C_{\min} para contornar o problema da negatividade de $\phi(X, r)$.

3.2.3 Transformando um Problema de Minimização em Maximização

Sabendo-se que os GAs trabalham em termos de maximização, outro problema aparece quando se quer encontrar o mínimo da função objetivo. Para contornar essa situação, são apresentadas duas maneiras de transformar problemas minimização em maximização. A primeira solução apresentada, descrita em [42] e [44], seria inverter a função objetivo como é mostrado a seguir:

$$f(x) = \frac{1}{\phi(X, r) + \varepsilon} \quad (3-6)$$

onde constante ε , em [44], tem o valor definido como sendo ligeiramente superior ao módulo do valor mínimo de $\phi(X, r)$, caso $\phi(X, r) \leq 0$, ou ligeiramente superior ao negativo do mínimo de $\phi(X, r)$, quando $\phi(X, r) > 0$. Outra maneira interessante de como utilizar a constante ε , é definindo-a como sendo um valor positivo de pequena magnitude (sugere-se $\varepsilon \in [10^{-3}, 10^{-8}]$) para todo $\phi(X, r) \geq 0$. Lembrando que, se $\phi(X, r)$ puder retornar valores negativos, deve-se modificá-la previamente com os métodos do início da seção. Com esse mapeamento, quanto menor for $\phi(X, r)$, maior será $f(x)$.

A segunda solução seria redimensionar $\phi(X, r)$ em relação a um valor C_{\max} , como pode ser mostrado pelo sistema:

$$\begin{cases} f(x) = C_{\max} - \phi(X, r) & \text{se } \phi(X, r) < C_{\max} \\ f(x) = 0 & \text{se } \phi(X, r) \geq C_{\max} \end{cases} \quad (3-7)$$

O parâmetro C_{\max} pode ser constante, ou variar ao longo das gerações. O valor inicial de C_{\max} é escolhido após vários testes de retorno de $\phi(X, r)$. Exemplificando os dois procedimentos, suponha agora que se quer minimizar $f(x) = x^2$, usando-se $\varepsilon=10^{-1}$ (esta magnitude ε tem apenas finalidade de facilitar graficamente a comparação com o outro procedimento) para a primeira transformação e $C_{\max}=5$ para a segunda. O resultado é o gráfico da Figura 3-3.

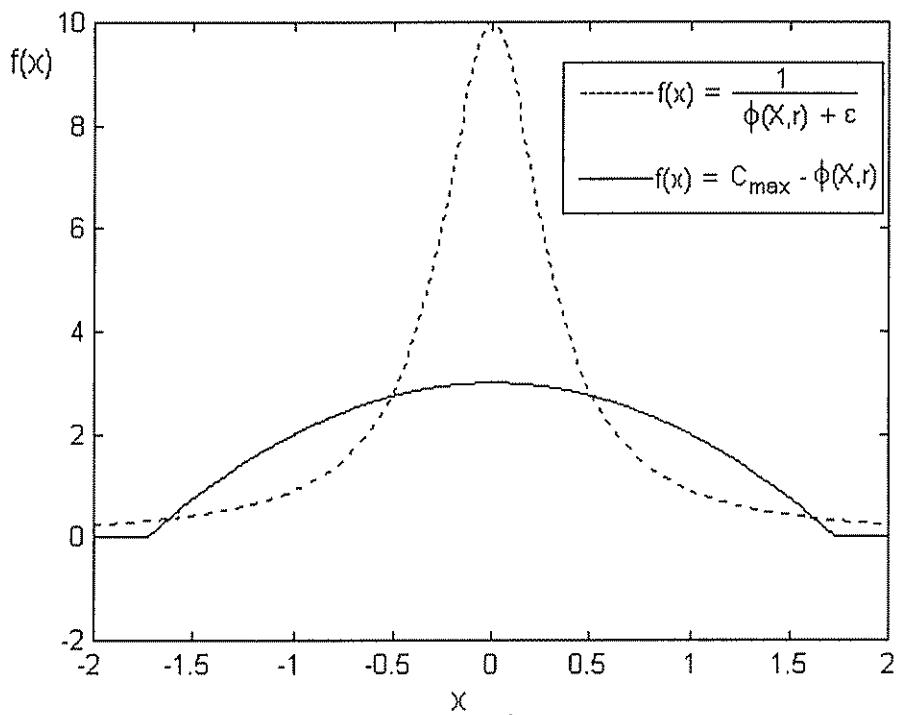


Figura 3-3: Transformação de $\phi(X, r)=x^2$, em função de maximização.

Transformar a função objetivo em uma função desempenho, mantendo-se as características da função original, é um fator importante para a execução e convergência dos GAs. Dependendo do problema tratado, mais de um tipo de transformação é requerido. No entanto, observando separadamente os métodos apresentados aqui, sugere-se a utilização das funções de penalidade para considerar as restrições, o do acréscimo de C_{\min} para evitar a negatividade de $\phi(X, r)$ e, para transformar um problema de minimização em maximização, sugere-se o método da inversão de $\phi(X, r)$.

3.3 Sistemas de Representação

Em sistemas artificiais, avalia-se a performance dos indivíduos submetendo-os a uma função matemática. Para serem apreciados por essa função, os indivíduos são codificados numa estrutura tipo cromossômica, formada por uma *cadeia de caracteres*. Nos GAs, cada *cromossomo* (ou *cadeia de caracteres*) representa uma variável. Dessa maneira, em problemas com múltiplas variáveis, deve-se, então, avaliar a ação conjunta do grupo de cromossomos, ou seja, o seu genótipo. Por exemplo, seja um problema multimodal, cuja solução é um vetor $X=[x_1, x_2, x_3]$. Cada x_i é um cromossomo e seu agrupamento X , é um indivíduo. Vale a observação de que quando o problema tem uma única variável, o cromossomo e o indivíduo são iguais.

O código mais usado é o binário, pois há facilidade na busca de similaridades e na implementação. Para ilustrá-lo, suponha que cada cromossomo tenha comprimento $\ell=5$. Dessa forma, um possível indivíduo para um problema de 3 variáveis poderia ser representado como a seguir:

$$\begin{array}{ccccccc} & x_1 & & x_2 & & x_3 & \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ & 0 & 1 & 0 & 1 & 1 & 0 \\ & 0 & 0 & 0 & 0 & 1 & 1 \end{array}$$

O código binário possui um problema de representação que é a presença dos *Hamming cliffs*, ou seja, grandes diferenças de bits nas *cadeias de caracteres* que codificam valores inteiros adjacentes. Por exemplo, as *cadeias de caracteres* 0111 e 1000, que representam os valores 7 e 8, são muito distantes no código binário. A distância *Hamming* denota a quantidade de bits diferentes que, no exemplo, são todos os quatro.

Com o intuito de resolver esse problema, um outro tipo de código binário, denominado de código Gray (criado por Frank Gray em 1953), foi aplicado. A finalidade era fornecer um código que diferenciasse de um bit para cada dois números inteiros em seqüência. Esse objetivo quase foi conseguido (excetuando casos, como o das *cadeias de caracteres* respectivas a 0 e 7 que são bem semelhantes). Para ilustrar a situação, veja o exemplo:

0	1	2	3	4	5	6	7	Valor binário
000	001	010	011	100	101	110	111	Gray
000	001	011	010	110	111	101	100	

Mesmo com as exceções, o código Gray possui mais adjacência (propriedade de semelhança na cadeia codificada para representação de inteiros em seqüência) na codificação que o código binário. Com adjacência, uma pequena taxa de mutação apenas ajuda na convergência final dos GAs, enquanto que no código binário uma

pequena taxa favorece a exploração de novas regiões, que é um fator importante para a globalidade da pesquisa. A transformação de uma *cadeia de caracteres* codificada binariamente para uma outra que utiliza o código Gray, ou vice-versa, é obtida através da aplicação dos seguintes algoritmos:

```

//Apresentação das variáveis

n          //Número de bits da cadeia de caracteres
           //Identifica também a posição do bit mais
           //significativo da cadeias de caracteres
i          //Contador
G[n]       //Vetor que contém a cadeia de caracteres
           //no código Gray
b[n]       //Vetor que contém a cadeia de caracteres
           //no código binário
XOR(bit, bit) //Função Ou exclusivo
               //Exemplo:      XOR(0,0) → 0
               //                  XOR(1,1) → 0
               //                  XOR(1,0) → 1
               //                  XOR(0,1) → 1

//Apresentação das funções

Função_Binário_Para_Gray(){
    i=n;
    G[i]=b[i];
    Enquanto (i≠0) {
        i=i-1;
        G[i]=XOR(b[i+1],b[i]);
    }
}

Função_Gray_Para_Binário(){
    i=n;
    b[i]=G[i];
    Enquanto (i≠0) {
        i=i-1;
        b[i]=XOR(b[i+1],G[i]);
    }
}

```

No código binário é importante enfatizar que, devido à presença dos *Hamming cliffs*, os operadores genéticos, tais como cruzamento e mutação, poderão ter dificuldade em vencer uma grande distância *Hamming*. A princípio, esse seria um fator de motivação para uso do código Gray, mas, mesmo assim, o código mais empregado é o binário. Logo, observa-se que não há consenso sobre qual dos dois códigos é melhor. Entretanto, para problemas que envolvem números inteiros, como análise combinatória, arranjo e permutação, pode ser bem mais adequada a aplicação de um outro tipo de código. Um exemplo seria o TSP (*Travelling Salesman*

Problem), em que um vendedor deve viajar por uma série de cidades e retornar à cidade de origem. Nesse problema, o objetivo é encontrar o menor percurso, conhecendo-se, a princípio, as distâncias existentes entre as cidades. Para resolver o TSP, um código inteiro formado pelos conjunto dos números naturais (0,1,2, ..., n) tem sido usado, por exemplo em [15] e [28].

Nos casos em que os códigos existentes não sejam os mais adequados na resolução de um problema específico a saída deve ser a criação de um novo. Mas, antes da implementação, deve-se observar sua viabilidade. Para guiar a implementação de novos códigos para os GAs, ou mesmo a escolha de um já existente, deve-se observar dois princípios básicos, segundo Goldberg [11]: o princípio do significado dos *building blocks* e o princípio dos *alfabetos mínimos*.

O primeiro princípio sugere que o código deve dar importância aos esquemas curtos e de baixa *ordem*, pois o mecanismo de procura pela melhor solução utilizará a sobreposição destes esquemas. Outro ponto é que as posições fixas sejam gradualmente diferenciadas uma das outras. Com isso, pode-se observar mais facilmente as similaridades.

O segundo princípio diz que o usuário deve selecionar o menor alfabeto que permita a expressão natural do problema. Como exemplo suponha uma *cadeia de caracteres* genérica que tenha comprimento ℓ e que possua k tipos diferentes de alelos. Haverá então k^{ℓ} diferentes *cadeias de caracteres*. Fixando o comprimento em $\ell=5$, o código binário teria $2^5=32$ diferentes *cadeias de caracteres*. Um outro código formado pelas letras do alfabeto inglês (a-z) teria 26 alelos, então para o mesmo ℓ , esse código teria $26^5=11.881.376$ *cadeias de caracteres*! O que se deseja em um problema real é encontrar uma solução com boa precisão, dentro de um grande espaço de procura e o mais rápido possível. Seria interessante uma boa quantidade, mas não exagerada, de indivíduos, em que se pudesse verificar facilmente as características comuns (similaridades). Mas, para códigos com grande diversidade de alelos, torna-se um trabalho duro pesquisar essas semelhanças, pois, num mesmo *locus*, pode-se ter k variações da característica genética.

Como o código escolhido neste trabalho é o binário, deve-se, então, saber como ele se enquadra nesses dois princípios. Nas ilustrações da Figura 2-1 e Figura 2-2, nota-se que esquemas curtos de baixa *ordem* varrem o espaço de procura e ajudam nos *building blocks*. O código binário com seus 0's e 1's facilitam a busca por semelhanças. E desde que esta é a essência do mecanismo de procura dos GAs,

quando se projeta um novo código, ele deve tentar propiciar o máximo de esquemas. Para um mesmo número de pontos no espaço, pode-se demonstrar que o código binário possui maior número de esquemas por bit que qualquer outro código. Para verificar esta afirmação, acompanhe o desenvolvimento a seguir.

Considere que o número de pontos possíveis no espaço de procura seja o mesmo para os códigos binário e um outro com $k > 2$. Então:

$$2^{\ell} = k^{\ell'}$$

o que resulta em:

$$\ell * \log 2 = \ell' * \log k \quad \rightarrow \quad \ell' = \ell * \log 2 / \log k \quad (3-8)$$

Como o número máximo de esquemas por *cadeia de caracteres* é 3^{ℓ} para o código binário e, $(k+1)^{\ell'}$ para os demais, nota-se que o código binário conduz a maior número de esquemas que qualquer outro. Este fato é ilustrado pela Figura 3-4, considerando $\ell=7$.

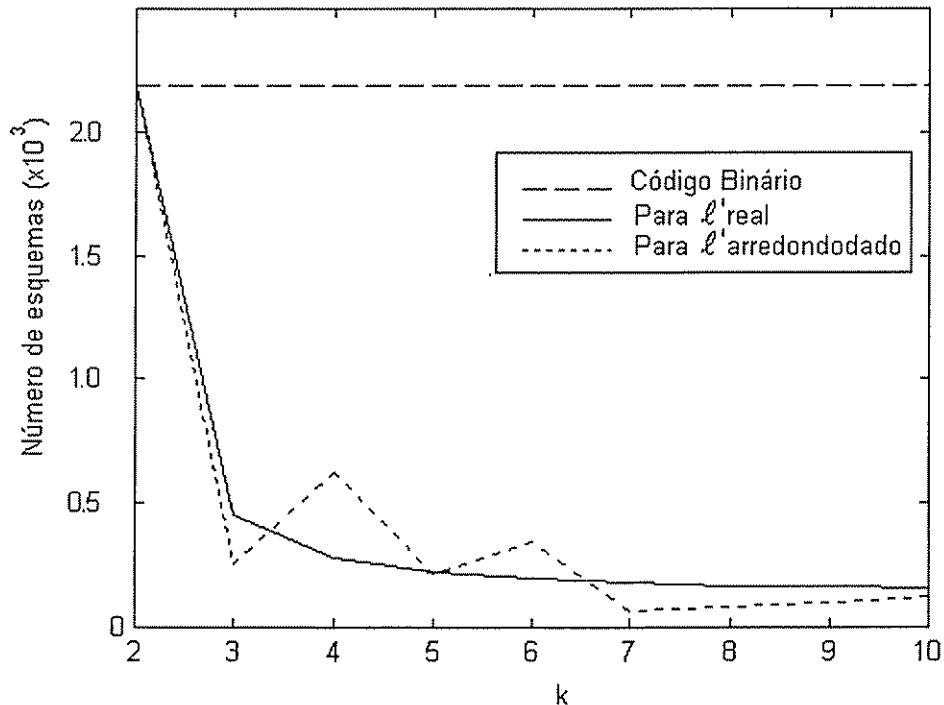


Figura 3-4:Número de esquemas possíveis para o mesmo número de pontos no domínio, em códigos diferentes.

3.4 Mapeando Variáveis

No exemplo do Capítulo 1, a variável x representa todos os valores inteiros compreendidos na faixa $[0, 2^{\ell}-1]$. Mas, as variáveis reais são bem mais adequadas

para modelar a grande maioria dos problemas. Então, deve-se fazer o mapeamento delas para o código com o qual os GAs trabalham. A relação existente entre o código binário (supondo que o cromossomo tenha comprimento $\ell=5$) e as variáveis reais limitadas por $[X_{\min}, X_{\max}]$ é a seguinte:

$$00000 \Rightarrow X_{\min}$$

$$11111 \Rightarrow X_{\max}$$

O mapeamento é feito linearmente, dependendo da resolução oferecida pelo número de bits usado em relação ao intervalo $[X_{\min}, X_{\max}]$. A precisão p_r , proveniente dessa resolução, pode ser calculada pela equação:

$$p_r = \frac{X_{\max} - X_{\min}}{2^\ell - 1} \quad (3-9)$$

O comprimento da *cadeia de caracteres*, para uma certa precisão requerida, pode ser determinado explicitando-se o ℓ da equação 3-9. Assim:

$$\ell = 1 + \text{inteiro} \left(\frac{\log \left(\frac{X_{\max} - X_{\min}}{p_r} + 1 \right)}{\log(2)} \right) \quad (3-10)$$

Quando o problema é multi-variável, concatenam-se os cromossomos para formação do indivíduo, como é mostrado a seguir:

1	1	x_1	0	0	1	0	x_2	1	1	0	0	x_3	1	1	1	x_4	1	1	1	1	x_5	0	0	1
---	---	-------	---	---	---	---	-------	---	---	---	---	-------	---	---	---	-------	---	---	---	---	-------	---	---	---

Em todas as referências analisadas, o comprimento ℓ da *cadeia de caracteres* não variou de parâmetro a parâmetro num mesmo problema de otimização.

3.5 Parâmetros dos GAs

Os GAs, geração após geração, tentam encontrar indivíduos com melhor desempenho que os anteriores. Mas, problemas intrínsecos do algoritmo podem dificultar a procura ou mesmo conduzir a pesquisa a uma direção errada. Nesses casos, provavelmente, está ocorrendo uma falha quantitativa ou qualitativa na ação dos operadores.

Quantitativamente pode-se listar o tamanho da população, o comprimento do cromossomo e as probabilidades de cruzamento e de mutação. Qualitativamente,

são os métodos de cruzamento, de escalonamento, as estratégias de seleção, entre outros. No Capítulo 4, uma análise dos resultados experimentais entre estes métodos é apresentada. Por ora, é interessante analisar uma faixa de valores para alguns desses parâmetros, tal que viabilize uma performance razoável dos GAs.

Comprimento do cromossomo: Sugere-se que o comprimento ℓ do cromossomo seja escolhido conforme a precisão requerida, o valor de ℓ pode ser determinado pela equação 3-10. Quanto maior o comprimento ℓ , maior o domínio de otimização, porém mais precisa será a solução.

Número de indivíduos da população (n_{pop}): Uma população pequena possui amostragem insuficiente da maioria dos hiperplanos, podendo conduzir o algoritmo na direção de um mínimo local. Uma população grande contém quantidade bem representativa do total de hiperplanos, e também, a homogeneização da população ocorrerá mais lentamente, possibilitando os GAs explorarem mais a informação existente. Entretanto, o número de cálculos de função desempenho por geração pode resultar num tempo computacional inaceitável.

Probabilidade de Cruzamento (p_c): Esse parâmetro controla a freqüência com a qual o operador de cruzamento é aplicado. A cada nova geração provavelmente $p_c * n_{pop}$ cruzamentos serão realizados. Baixo p_c significa pouco aproveitamento da informação existente e um alto valor de p_c pode provocar *convergência prematura* (homogeneização rápida da população), pois o cruzamento associado à reprodução ajuda a uniformizar a população.

Probabilidade de Mutação (p_m): A mutação tem um papel diferente, mas não menos importante que o cruzamento. As funções da mutação são inserir material genético novo e restaurar alelos perdidos no cruzamento ou mesmo na mutação. A mutação é realizada bit a bit segundo uma probabilidade p_m . Numa população onde o comprimento total da cadeia cromossômica ((número de variáveis)* ℓ) de um indivíduo é L , provavelmente acontecerão $p_m * n_{pop} * L$ mutações por geração. Pequenos valores de p_m não proporcionam a satisfação dos objetivos da mutação, levando o algoritmo a ficar estagnado nos hiperplanos existentes. Por outro lado, alto valor de p_m conduz os GAs à procura aleatória.

A escolha ideal dos parâmetros é um problema não linear, e dependente do tipo de problema tratado. Isso dificulta encontrar uma boa configuração para generalizar a execução de qualquer tipo de problema. De Jong [7] sugere $n_{pop}=50$, $p_c=0.60$ $p_m=0.001$. Grefenstette testou vários conjuntos de valores de parâmetros em [14] e

encontrou dois que melhor satisfizeram os medidores de desempenho conhecidos como *on-line* e *off-line performance* (veja seção 3.13), os quais foram apresentados por De Jong em [6] e [7]. Relativamente ao tipo *on-line*, Grefenstette encontrou $n_{pop}=30$, $p_c=0.95$, $p_m=0.01$ e para a *off-line performance*, $n_{pop}=80$, $p_c=0.45$, $p_m=0.01$, dando resultados ligeiramente superiores ao de De Jong. É importante salientar que, no grupo de parâmetros pesquisados por De Jong, a mutação é um operador secundário, pois tem valor bem reduzido. Grefenstette concluiu que, provavelmente, uma variação dinâmica de alguns dos parâmetros seria superior ao seu resultado. Essa afirmação motivou um estudo mais aprofundado em relação ao tema e conduziu esta dissertação a uma contribuição científica, a qual será descrita no item *Variação Dinâmica das Probabilidades dos Operadores Cruzamento e Mutação*.

3.6 A Mutação

A mutação é um operador genético que tem a função de introduzir características novas ao indivíduo ou mesmo restaurar características que se perderam em operações, como, por exemplo, de cruzamento.

Na teoria, a mutação é um evento que possui uma probabilidade p_m para cada bit da cadeia de caracteres de todos os indivíduos da população. Se a implementação da mutação for realizada como descrito acima, então a mutação pode se tornar um operador de processamento caro. Por exemplo, considere que um indivíduo tenha 10 variáveis e cada variável possua uma cadeia de caracteres de $\ell=20$ bits. O comprimento total L do indivíduo é $L=10*20=200$ e, portanto, esse será o número de verificações de ocorrência da mutação. Considerando a probabilidade de mutação $p_m=0.01$, provavelmente só se efetivarão $0.01*200=2$ mutações em cada indivíduo. Nesse caso, essa implementação de mutação é dispendiosa computacionalmente, pois apenas devem acontecer 2 mutações em 200 bits candidatos em cada indivíduo. O problema se agrava quando se leva em conta o tamanho da população. Entretanto, há situações em que essa implementação é viável. Por exemplo, quando $p_m*L < 1$. O algoritmo a seguir mostra uma boa implementação de mutação.

```
//função mutação

mutação(indivíduo, L, pm) {
    se pm*L<1 {
        verificar_mutação_bit_a_bit(indivíduo);
    }
    senão {
        número_de_mutações=inteiro_arredondado(pm*L);
        Enquanto número_de_mutações>0 {
            mutar_bit(indivíduo, random(1,L));
            número_de_mutações=número_de_mutações-1;
        }
    }
}
```

Essa implementação é interessante quando o argumento utilizado para efetuar a mutação é o indivíduo. No entanto, o desempenho de processamento desse operador genético pode ainda ser melhorado se o argumento for a população. O algoritmo a seguir mostra a implementação.

```
//função mutação

mutação(população, npop, L, pm) {
    se npop*pm*L<1 {
        verificar_mutação_bit_a_bit_de_cada_indivíduo();
    }
    senão {
        número_de_mutações=inteiro_arredondado(npop*pm*L);
        Enquanto número_de_mutações>0 {
            escolhido=escolher_indivíduo(população);
            mutar_bit(escolhido, random(1,L));
            número_de_mutações=número_de_mutações-1;
        }
    }
}
```

As implementações de mutação descritas são válidas quando a codificação utilizada é a binária. No entanto, para problemas que envolvem ordenação e permutação (por exemplo o TSP) o operador mutação deve evitar que haja repetição de variáveis num mesmo indivíduo. A ilustração abaixo representa um possível indivíduo para o problema TSP de 10 cidades.

1 2 3 4 5 6 7 8 9 0

Se esse indivíduo sofrer mutação, por exemplo na terceira variável, para qual valor a variável mudará? Em [28] os autores utilizaram uma mutação chamada *troca* (*swap*), que ocorre numa probabilidade p_{swap} . Nessa implementação, escolhe-se duas variáveis e estas trocam de posição.

O operador mutação tem papel fundamental na evolução da população (introdução e restauração de características). Mas deve-se tomar cuidado com a implementação, pois pode-se ter um custo computacional elevado.

3.7 Métodos de Cruzamento

O cruzamento é o principal mecanismo na geração de novos pontos no espaço de otimização, o qual proporciona troca de informação genética entre indivíduos. Holland, em [18], propôs o cruzamento com um único ponto de corte. Esse tipo é muito importante na teoria dos *building blocks*. Depois disto, aparecem outras formas de cruzamento, como de 2 a n pontos de corte e o cruzamento Uniforme. Além desses tipos de cruzamento já bem conhecidos na literatura, descreve-se o cruzamento entre vários indivíduos simultaneamente (CEVI), o qual foi apresentado primeiramente em [43], e também o cruzamento por variável (CPV), que foi proposto nesta dissertação. A seguir, apresenta-se cada método e, para ajudar em sua compreensão, serão usados, como casal de cruzamento, os seguintes indivíduos com três variáveis:

$$\begin{array}{cccccccccccccc} 1 & - & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 2 & - & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{array}$$

Cruzamento com n pontos de corte: Similarmente ao cruzamento com um ponto, os locais são escolhidos aleatoriamente. Se algum ponto de corte foi sorteado mais de uma vez, não se procura por outro. Os locais de corte do indivíduo acima variam de 1 a L-1, sendo L o comprimento total do indivíduo. Então, supondo que se esteja tratando do tipo com 4 pontos de corte, e que os escolhidos sejam 4, 6, 9 e 12, tem-se:

$$\begin{array}{cccccccc|cc|cccc|cc|cc} 1 & - & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 2 & - & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{array}$$

Resultando:

$$\begin{array}{cccccccc|cc|cccc|cc|cc} 1 & - & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 2 & - & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{array}$$

Cruzamento Uniforme: Seguindo cada bit do primeiro indivíduo, verifica-se se ocorreu um evento com probabilidade de 50%. Caso afirmativo, ali é um ponto de corte, caso contrário, repete-se o procedimento para o bit posterior. Espera-se, neste método, por $(L-1)/2$ pontos de corte.

Cruzamento Por Variável (CPV): Diferenciando-se dos métodos descritos anteriormente, aqui o cruzamento terá um ponto de corte por variável. A idéia de

realizar o cruzamento por variável parece ser bem natural, no entanto, não encontrou-se na literatura nada a respeito, e portanto, considerou-se esse tipo de cruzamento como contribuição desta dissertação. O cruzamento CPV é ilustrado abaixo:

1 - 1 0 1 0 0 0 0 0 1 0 0 0 1 1 1 0
2 - 1 1 0 1 1 1 0 1 1 0 0 0 1 0 0 0

Como são três variáveis, o cruzamento precisa de três pontos de corte. Considere que os pontos de corte sejam 2, 2 e 3 para as variáveis respectivamente apresentadas. Assim:

1 - 1 0 1 0 0 0 0 0 1 0 0 1 1 1 0
2 - 1 1 0 1 1 1 1 0 1 1 0 0 1 0 0 0

Resultando:

1 - 1 0 0 1 1 0 0 0 1 1 0 0 1 1 0 0
2 - 1 1 1 0 0 1 0 0 1 0 0 0 1 0 1 0

Cruzamento Entre Vários Indivíduos (CEVI): Inicialmente, seleciona-se o indivíduo base e depois, para cada variável, escolhe-se aleatoriamente um parceiro e um ponto de corte. Com isso, tenta-se aproveitar a riqueza da informação mais rapidamente. Para mostrar seu funcionamento, suponha que o indivíduo base seja o 1. Como ele é composto por três variáveis, serão necessários mais dois indivíduos para compor o grupo de cruzamento. É importante salientar que o resultado é apenas um indivíduo proveniente do base. Os pontos de corte para as variáveis são 2, 3 e 1 respectivamente aos indivíduos 2, 3 e 4.

1 - 1 0 1 0 0 0 0 1 0 0 1 1 1 0
2 - 1 1 0 1 1 1 1 0 1 1 0 0 1 0 0 0
3 - 0 1 0 1 0 1 1 0 0 1 1 1 0 1 1
4 - 1 0 0 1 1 1 1 0 0 1 0 1 1 0

Resultando:

1 - 1 0 0 1 1 0 0 0 0 1 0 0 1 1 0

Os métodos de cruzamento descritos abaixo, também conhecidos como operadores de recombinação, foram projetados para problemas de ordenação, como arranjos e permutações. Nesses casos, deve-se evitar que o cruzamento introduza repetição de variável num mesmo indivíduo. Essa restrição inviabiliza os procedimentos de cruzamentos descritos anteriormente. Os cruzamentos aqui apresentados foram descritos em [11] e [28]. Em [28] foi comparada a performance dos três tipos de cruzamento (OX, CX e PMX).

Cruzamento parcial (PMX - Partially Matched Crossover): Desse tipo de cruzamento participam dois indivíduos e geram outros dois. Para executá-lo, há necessidade de dois pontos de corte que, como nos casos anteriores, são escolhidos aleatoriamente. Esses dois pontos definem uma seção em que os alelos pertencentes a ela trocarão de posição com os provenientes do parceiro. O casal apresentado a seguir mostra o procedimento.

1	-	1	2	3	4	5	6	7	8	9	0
2	-	8	1	6	5	0	3	9	2	7	4

Com o corte ocorrendo nas posições 4 e 7, troca-se o conteúdo dentro da faixa:

1	-	1	2	3	4	0	3	9	8	9	0
2	-	8	1	6	5	5	6	7	2	7	4

Os alelos fora da faixa repetidos, são substituídos pelos respectivos alelos fornecidos ao outro indivíduo, finalizando o processo. Assim:

1	-	1	2	6	4	0	3	9	8	7	5
2	-	8	1	3	0	5	6	7	2	9	4

Cruzamento seqüencial (OX - Order Crossover): Para descrever o cruzamento OX, considere os indivíduos anteriormente descritos e com os mesmos dois pontos de corte utilizados para o cruzamento PMX.

1	-	1	2	3	4	5	6	7	8	9	0
2	-	8	1	6	5	0	3	9	2	7	4

Da ilustração, observa-se que os indivíduos 1 e 2 trocarão os alelos dentro da faixa. Do ponto de vista de cada indivíduo, os alelos a serem recebidos são marcados por uma letra H, como pode ser observado a seguir:

1	-	1	2	H	4	5	6	7	8	H	H
2	-	8	1	H	H	0	3	9	2	H	4

Agora, a partir do segundo ponto de corte, os alelos não marcados com H, deslizam preenchendo os espaços dos alelos marcados com H. Para facilitar o entendimento, os próximos passos serão desenvolvidos somente para o indivíduo 1. Assim:

1	-	2	H	4	5	6	7	H	8	H	1
1	-	H	4	5	6	7	H	H	8	1	2
1	-	4	5	6	7	H	H	H	8	1	2

Finalmente, substituem-se os Hs pelos alelos recebidos, então:

1	-	4	5	6	7		0	3	9		8	1	2
---	---	---	---	---	---	--	---	---	---	--	---	---	---

Procedendo da mesma forma, obtém-se o segundo indivíduo do cruzamento. O resultado final do cruzamento OX é mostrado a seguir:

1	-	4	5	6	7		0	3	9		8	1	2
2	-	1	0	3	9		5	6	7		2	4	8

Embora os procedimentos dos cruzamentos PMX e OX sejam semelhantes, eles possuem objetivos diferentes. Enquanto que o PMX tende a respeitar a posição absoluta dos alelos, o cruzamento OX busca respeitar a posição relativa.

Cruzamento Cíclico (CX - Cycle Crossover): Esse tipo de cruzamento não necessita de pontos de corte, já que ele é dependente apenas das posições dos integrantes do cruzamento. Para explicar o procedimento, considere o casal previamente descrito:

1	-	1	2	3	4	5	6	7	8	9	0
2	-	8	1	6	5	0	3	9	2	7	4

O processo inicia-se considerando-se um indivíduo, por exemplo o 1, como a estrutura básica para o cruzamento. A seguir fixa-se o seu primeiro alelo.

1 - 1

Agora, verificando-se o segundo indivíduo do cruzamento, nota-se que o alelo correspondente ao primeiro é o alelo 8. Esse será o próximo a ser fixo, porém, no segundo indivíduo resultante. Depois disso fixa-se esse alelo no indivíduo resultante 1, como mostrado a seguir:

1	-	1									8
			↓								↑
2	-	8	→	→	→	→	→	→	→	→	

A partir desse ponto, o processo se repete até que o alelo respectivo no indivíduo 2 já tenha sido usado. Assim:

1	-	1	2	←	←	←					8
				↓			↑				↓
2	-	8	1		←	←	←	2			

Para concluir o mecanismo, trocam-se as partes não fixas entre os indivíduos 1 e 2. Assim:

1	-	1	2	6	5	0	3	9	8	7	4
2	-	8	1	3	4	5	6	7	2	9	0

Operador de Recombinação de Fronteiras (ERO - Edge Recombination Operator): Esse operador, que foi criado por Whitley & Outros (Capítulo 22 de [4]),

tem o objetivo de desenvolver uma codificação que permita a operação de recombinação utilizar as melhores fronteiras atuais dos indivíduos pais. Segundo os autores, os operadores de recombinação que quebram as ligações já existentes, introduzem uma mutação indesejada. Para melhor explicação do processo, acompanhe o exemplo de TSP (6 cidades), que vem a seguir.

Inicialmente, o método de seleção escolhe dois pais para se realizar a operação de recombinação. Suponha então que os pais sejam:

1	-	1	2	3	4	5	6
2	-	5	1	6	2	4	3

A seguir, é necessário construir algum tipo de banco de dados que armazene todas as conexões existentes entre as cidades de cada indivíduo pai. Para esse exemplo a informação armazenada deve ser:

cidade	fronteiras				cidade	fronteiras		
1	2	6	5		4	3	5	2
2	1	3	6	4	5	4	6	1
3	2	4	5		6	5	1	2

O processo segue escolhendo-se a cidade inicial de um dos pais, a qual será denominada como *cidade corrente*. A escolha pode ser realizada a partir de algum critério (por exemplo, a cidade inicial que tem menor quantidade de fronteiras) ou simplesmente pode ser aleatória. Considere o indivíduo 2 escolhido. Então a *cidade corrente* deve ser a cidade 5 (primeira cidade do indivíduo 2). A seguir, exclua todas as referências à cidade 5 do banco de dados.

cidade	fronteiras				cidade	fronteiras		
1	2	6			4	3	2	
2	1	3	6	4	5 - 1 ^a escolha	4	6	1
3	2	4			6	1	2	

Observando, no banco de dados atualizado, que as cidades 4, 6, 1 e 3 fazem fronteira com a *cidade corrente*. Dessa cidades, a que possuir menos fronteiras será a escolhida. Como todas as cidades (4, 6, 1 e 3) têm o mesmo número de fronteiras (2 fronteiras cada), a próxima cidade é escolhida aleatoriamente. Considere a cidade 4 como escolhida e atualize novamente o banco de dados.

Cidade	fronteiras				Cidade	fronteiras		
1	2	6			4 - 2 ^a escolha	3	2	
2	1	3	6		5 - 1 ^a escolha			
3	2				6	1	2	

Das cidades que fazem fronteira com a *cidade corrente*, a cidade 3 é a que possui agora menos conexões, tornando-se, portanto, a nova *cidade corrente*.

cidade	fronteiras	cidade	fronteiras
1	2 6	4 - 2 ^a corrente	
2	1 6	5 - 1 ^a corrente	
3 - 3 ^a corrente	2	6	1 2

O processo continua até que se percorra todas as cidades. O indivíduo filho, proveniente deste operador de recombinação, é o indivíduo formado pelas cidades que foram *cidade corrente* na ordem de ocorrência. Nas situações em que a *cidade corrente* não possui mais fronteiras e ainda restam cidades a serem utilizadas, a escolha da próxima *cidade corrente* é aleatória. Para esse exemplo, um dos resultados desse tipo de cruzamento é o indivíduo:

5 4 3 2 6 1

Nessa seção, foram apresentadas várias maneiras de se realizar o cruzamento. Os operadores de recombinação (que tratam de arranjo e permutação) não foram analisados numericamente no Capítulo 4. Por outro lado, foram analisados numericamente os cruzamentos Uniforme, CPV, CEVI e os cruzamentos com 1 e 2 pontos de corte.

3.8 A Inversão

Em Holland [18], existem três técnicas que diferenciam a nova geração da anterior, são eles: cruzamento, mutação e inversão. Das três, resta descrever a inversão. Esse operador é considerado como secundário para os GAs e foi criado com a finalidade de evitar que bons esquemas se rompam durante o cruzamento. Para executar o mecanismo de inversão, os GAs devem possuir um vetor que conterá a informação sobre a posição de cada bit no indivíduo, como pode ser visto a seguir:

indivíduo 1	- 1 0 1 1 0 1
posição associada	- 1 2 3 4 5 6

O objetivo é reordenar a estrutura cromossômica para fortalecer o enlace entre caracteres fixos de esquemas de grande *comprimento de definição*, antes de aplicar o cruzamento. A inversão opera sobre um simples indivíduo, numa determinada probabilidade p_{inv} , invertendo-se a ordem dos elementos entre dois pontos de corte aleatoriamente escolhidos, como é mostrado a seguir:

1 0 1 1 0 1

E depois da inversão:

1 0 0 1 1 1

Segundo Vasconcelos & Outros [44] o efeito da inversão em um código genético não está bem claro e nem é de fácil entendimento como os operadores mutação e cruzamento. Para Davis [4], o operador inversão traz consigo um custo computacional adicional aos GAs, não sendo considerado útil na prática e ainda é raramente empregado. Assim sendo, a inversão não será avaliada neste trabalho, e ela está aqui apresentada apenas para completar a descrição dos operadores genéticos. Maiores detalhes veja [4], [11], [18] e [44].

3.9 Métodos de Seleção

O processo seleção/reprodução é responsável pela escolha dos indivíduos que serão submetidos às operações genéticas como cruzamento e mutação, e os indivíduos resultantes dessas operações compõem a nova geração. Não é uma boa característica favorecer sempre a seleção do melhor, muito menos uma escolha aleatória. Por um lado há possibilidade de ocorrer *convergência prematura* e, por outro, a pesquisa é aleatória, deixando de explorar as informações contidas no seio da população. A seguir, é feita a descrição de alguns métodos de seleção, maiores detalhes em [11].

Roleta: Como explicado anteriormente, cada indivíduo tem a oportunidade de ser selecionado de acordo com o seu desempenho relativo ao da população, ou seja, $p_{sel} = f_i / \sum f$. Esse método conduz à *convergência prematura* em poucas gerações, já que o crescimento pode ser exponencial.

Torneio: Retorna o melhor indivíduo entre dois obtidos no método da Roleta. Este método busca dificultar, mas não elimina, as possibilidades de um indivíduo com baixo desempenho ser escolhido. O método também favorece a *convergência prematura*, já que é baseado na Roleta.

Deterministic Sampling (DS): O procedimento de seleção possui dois estágios. O primeiro é a criação de uma população temporária, a qual é preenchida com o número inteiro do cálculo da expectativa de cópias de cada membro i da população (f_i/f_{med}). Devido às partes fracionárias provenientes desta expectativa, haverá vagas ociosas na população. No segundo estágio, essas vagas serão preenchidas de acordo com os indivíduos que possuírem a parte fracionária do valor de desempenho mais alta.

Stochastic Remainder Sampling (SRS): Similarmente ao método anterior, deve-se formar uma população temporária com a parte inteira da expectativa de cópias. As vagas restantes são preenchidas, verificando-se a ocorrência de um evento com a probabilidade da parte fracionária. Por exemplo, um indivíduo que tem a expectativa de 2.3 cópias para a próxima geração, terá 2 cópias garantidas, mais 30% de chances de mais uma vaga. No entanto, deve-se estabelecer algum critério de escolha dos indivíduos candidatos às vagas ociosas. Por exemplo, os indivíduos podem ser escolhidos aleatoriamente e, em seguida verifica-se um evento com probabilidade de sua parte fracionária. O critério adotado nesse trabalho, foi escolher os indivíduos na ordem crescente de desempenho. Fazendo assim, o método de seleção concede uma oportunidade aos indivíduos de menores valores de desempenho.

Stochastic Universal Sampling (SUS): Escolha aleatória entre membros da população. Assim, cada indivíduo tem a chance de $1/n_{pop}$ de ser escolhido, sendo n_{pop} o número total de membros da população.

Alguns mecanismos de seleção são bem tendenciosos, como o Torneio, outros são bem imparciais, como o método SUS. Quais são melhores? A comparação numérica e discussão dos resultados obtidos empregando os diversos métodos de seleção é realizada no Capítulo 4.

3.10 Escalonando a População

Entre os métodos de seleção mais usados encontra-se o da Roleta. Nele, a probabilidade de seleção p_{sel} é baseada na razão entre aptidão do indivíduo f_i e a soma total dos desempenhos da população ($p_{sel}=f_i/\sum f$). Esse método, que parece justo, pode às vezes conduzir os GAs a um ótimo local. Isto acontece porque, normalmente, as populações iniciais contêm indivíduos de baixo desempenho e, se por acaso entre estes surgir um superindivíduo, ele obterá várias cópias para próxima geração, direcionando fortemente o algoritmo para a região de seus esquemas. Se esse indivíduo integrar o corpo de um pico local, certamente haverá muitas chances de se ficar preso nessa região. Com isso, os GAs perdem a diversidade antes de atingir o objetivo principal. Como apresentado anteriormente, esse problema é denominado *convergência prematura*. Evitá-la é praticamente impossível, mas um método de suavizá-la é escalar os membros da população, limitando probabilisticamente o número máximo de cópias para a próxima geração.

O primeiro método de escalonamento é o *Linear*, onde existe uma relação linear entre os valores de desempenho real e o escalonado, como é mostrado na equação:

$$f' = a * f + b \quad (3-11)$$

onde f é valor do desempenho, f' seu valor escalonado, a e b são escolhidos de modo que a reta passe pelos pontos $(f_{\text{med}}, f'_{\text{med}})$ e $(f_{\text{max}}, f'_{\text{max}})$. Os parâmetros f_{med} e f_{max} são respectivamente os valores médio e máximo encontrados na população e os pares correspondentes, são seus valores escalonados. O primeiro ponto tem a finalidade de garantir que os indivíduos medianos consigam manter uma cópia para a próxima geração. Para tentar garantir essa cópia, faz-se então $f'_{\text{med}}=f_{\text{med}}$. O segundo ponto tenta restringir o número máximo de cópias para o melhor indivíduo. Ele é calculado em relação a f_{med} , da seguinte maneira:

$$f'_{\text{max}} = C * f_{\text{med}} \quad (3-12)$$

onde a constante C é definida com valores situados na faixa [1.2-2.0]. A Figura 3-5 mostra a ação do escalonamento linear.

Um problema surge quando a população começa a se uniformizar, ou seja, f_{med} se aproximando de f_{max} . Nesse ponto, indivíduos bem abaixo da média podem ter valores de escalonamento negativo, como mostrado na Figura 3-6. Para eliminar esse problema, define-se $f'=0$ quando $f'<0$.

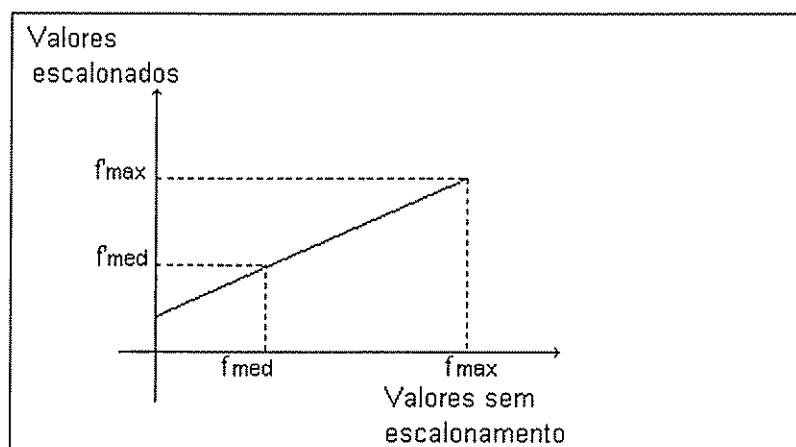
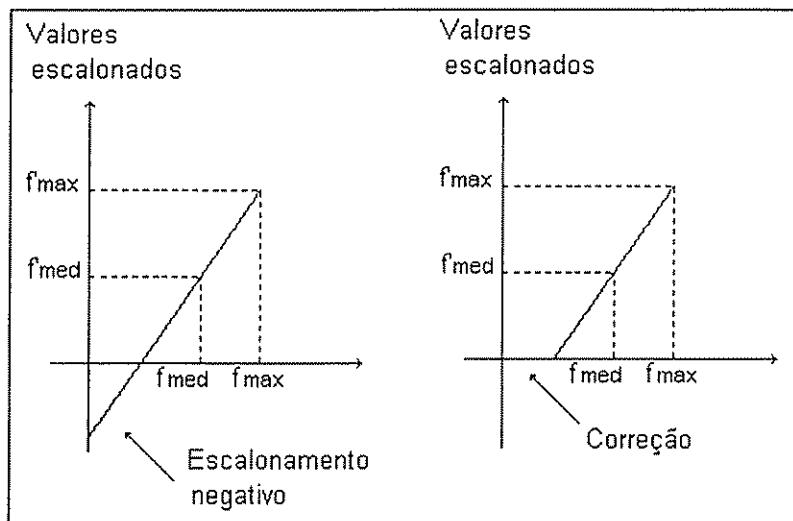


Figura 3-5: Esquema do escalonamento linear.

Figura 3-6: Negatividade de f' à esquerda e, à direita a solução.

O segundo escalonamento é o *Sigma Truncado*, onde o principal objetivo é aproveitar melhor a diversidade da população, com a minimização do problema de negatividade do escalonamento anterior. O método leva em conta o desvio padrão σ e a média f_{med} da população, como é descrito pela equação:

$$\begin{aligned} f' &= f - (f_{\text{med}} - C * \sigma) && \text{se } f > f_{\text{med}} - C * \sigma \\ f' &= 0 && \text{se } f < f_{\text{med}} - C * \sigma \end{aligned} \quad (3-13)$$

neste caso, a constante C tem valores recomendados na literatura [11] entre 1 e 3. O procedimento elimina os indivíduos que tenham desempenho inferior a $(f_{\text{med}} - C * \sigma)$. Esse método tenta explorar estatisticamente a *distribuição normal* dos indivíduos na população, onde o valor de C determina estatisticamente a porcentagem atingida pelo escalonamento. A Figura 3-7 mostra a ação do escalonamento Sigma para alguns valores de C .

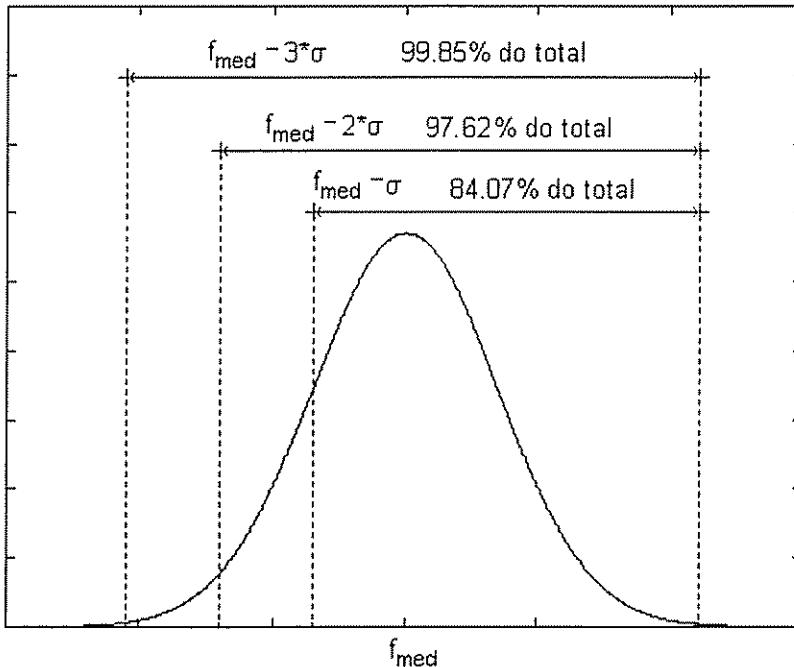


Figura 3-7: Curva da *distribuição normal* da população e a porcentagem dos indivíduos atingidos pelo escalonamento Sigma Truncado.

Após o escalonamento, novamente é necessário o cálculo da média dos valores escalonados, para que se possa usar o método de seleção.

Os métodos de escalonamento realmente suavizam a convergência, tentando aumentar as chances dos GAs encontrarem a solução global. Mas nem sempre conseguem. Adiante, na seção 3.11, serão mostrados outros mecanismos que auxiliam o escalonamento no objetivo de suavizar a convergência.

3.11 Variação Dinâmica das Probabilidades dos Operadores Cruzamento e Mutação

Em problemas multimodais, espera-se que os GAs possam convergir para a solução global. Dessa forma, deve se preocupar em diminuir a possibilidade dele ficar preso numa região de ótimo local. Essa possibilidade aumenta à medida em que a diversidade genética dentro da população se reduz. Se não houver nenhum mecanismo que restaure essa diversidade, a convergência para o ótimo global pode ficar comprometida. Neste trabalho, o desempenho médio da população (f_{med}), dividido pelo melhor resultado (f_{max}), foi tomado como medida da diversidade genética ($m_{dg} = f_{med}/f_{max}$). Se m_{dg} é próximo da unidade, significa que há pouca diversidade, e muita, quando ele se aproxima de zero. A Figura 3-8 mostra que, na medida em que o número de gerações cresce, o valor médio da função desempenho

se aproxima do valor máximo encontrado, o que dá valores de m_{dg} próximos da unidade, indicando escassez de material genético dentro da população.

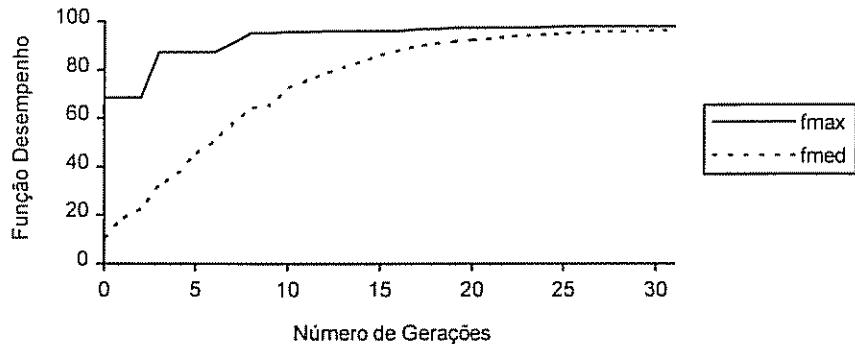


Figura 3-8: Evolução dos valores máximo e médio da função desempenho.

Para se conseguir resultados satisfatórios, deve haver diversidade genética suficiente no âmbito da população, durante as gerações, de tal modo a permitir que outras regiões, além da definida pelo indivíduo de melhor performance, sejam representadas na população. Com isso, os problemas de *convergência prematura* poderiam ser evitados. Os métodos de escalonamento descritos na seção 3.10, suavizam a *convergência prematura*, simplesmente limitando, probabilisticamente, o número de cópias por geração para cada indivíduo. Resultados obtidos na resolução de alguns problemas de otimização utilizando-se de funções teste multimodais, têm mostrado que somente esse mecanismo não é suficiente para se evitar a estagnação em mínimos locais. Assim, com o objetivo de manter um nível suficiente de diversidade genética, é que se tem proposto na literatura a variação dinâmica das probabilidades de cruzamento e mutação [32], [37] e [46], no decorrer das gerações.

Inúmeros trabalhos foram publicados explorando os operadores cruzamento e mutação. Por exemplo, maneiras diferentes de se efetuar a operação cruzamento [33], comportamento dos GAs frente a variados valores das probabilidades p_c e p_m [7] e [14] e, adaptação dinâmica destas probabilidades [32], [37] e [46].

Neste trabalho, a variação dinâmica das probabilidades de cruzamento e mutação são analisadas, utilizando-se de três critérios: a adaptação dinâmica por indivíduo e a adaptação dinâmica segundo o comportamento médio da população, esta última, com duas variações. Uma delas foi proposta no desenvolvimento deste trabalho.

Adaptação dinâmica por indivíduo (PI) [37]: As probabilidades de cruzamento e mutação variam de acordo com o desempenho de cada indivíduo e com os desempenhos médio e máximo encontrados na população. Matematicamente, este critério foi estabelecido como:

$$p_c = k_1 \frac{(f_{\max} - f')}{(f_{\max} - f_{\text{med}})} \quad \text{se } f' \geq f_{\text{med}} \quad (3-14)$$

$$p_c = k_3 \quad \text{se } f' < f_{\text{med}} \quad (3-15)$$

$$p_m = k_2 \frac{(f_{\max} - f)}{(f_{\max} - f_{\text{med}})} \quad \text{se } f \geq f_{\text{med}} \quad (3-16)$$

$$p_m = k_4 \quad \text{se } f < f_{\text{med}} \quad (3-17)$$

Acima, f' representa o maior valor da função desempenho entre os dois membros participantes do cruzamento, f é o valor da função avaliada para o indivíduo sujeito à mutação e f_{\max} e f_{med} são, respectivamente, os valores máximo e médio dessa função no âmbito da população. Os valores $k_1=k_3=1.0$ e $k_2=k_4=0.5$ foram sugeridos em [37].

Nesse critério, deseja-se evitar que as boas soluções se percam com facilidade, e que as más sejam totalmente destruídas. Analisando as equações acima, verifica-se que, se o indivíduo tem bom desempenho (valor da função de desempenho acima do valor médio), então as probabilidades p_c e p_m terão valores pequenos, a ponto de se anularem quando o indivíduo em questão é o de melhor performance. Dessa forma, a melhor solução fica explicitamente guardada para a próxima geração. Quanto aos indivíduos com valores da função desempenho abaixo da média, esses valores serão aumentados, permitindo-lhes máxima possibilidade de cruzamento e altas taxas de mutação.

Adaptação dinâmica de p_c e p_m segundo o comportamento médio da população [32], [45] e [46]: O comportamento da população é analisado observando o valor m_{dg} . Se ele estiver abaixo de um valor V_{\min} , considera-se que há grande diversidade genética. Nesse caso, deve-se diminuir a taxa de introdução de novas características genéticas no seio da população, acelerando a velocidade de convergência do algoritmo. Por outro lado, se m_{dg} for maior que um valor V_{\max} , haveria pouca diversidade genética, o que poderia proporcionar a *convergência prematura*. Para contornar essa situação, aumenta-se a taxa de introdução de novas características, diminuindo a velocidade de convergência, aumentando a possibilidade de se encontrar o ótimo global. Procura-se, portanto, encontrar valores ideais para V_{\min} e V_{\max} tais que, na faixa compreendida entre eles, o nível de diversidade genética seja aceitável. Para manter m_{dg} dentro do intervalo $[V_{\min}, V_{\max}]$, são apresentados dois critérios de adaptação das probabilidades p_m e p_c : adaptação *Fora da Faixa* (FF) e adaptação *Dentro da Faixa* (DF).

A adaptação FF pode facilmente ser implementada. Quando $m_{dg} > V_{max}$, aumenta-se p_m e simultaneamente diminui-se p_c , pois, com p_m maior, há mais inserção de material genético novo na população e com p_c baixo, há pouca troca de material genético. Por outro lado, quando $m_{dg} < V_{min}$, faz-se o contrário. A alteração dos valores das probabilidades é percentual, agindo quando os valores de m_{dg} são respectivamente superiores e inferiores a V_{max} e V_{min} . O algoritmo a seguir mostra o procedimento.

```

se  $m_{dg} > V_{max}$  {
     $p_m = k_m * p_m;$ 
     $p_c = p_c / k_c;$ 
}
se  $m_{dg} < V_{min}$  {
     $p_m = p_m / k_m;$ 
     $p_c = k_c * p_c;$ 
}

```

Se p_m e/ou p_c tiverem valores acima ou abaixo de limites previamente escolhidos, lhes serão atribuídos valores limites.

No tipo de adaptação DF (contribuição desta dissertação), quando $m_{dg} \geq V_{max}$, faz-se p_m e p_c respectivamente iguais a $p_{m\max}$ e $p_{c\min}$ ou $p_{m\min}$ e $p_{c\max}$, caso contrário. Quando os valores são intermediários, usa-se uma interpolação linear em relação a um valor V_{ideal} (por exemplo, $V_{ideal} = (V_{max} + V_{min})/2$). O algoritmo e a Figura 3-9 explicam melhor o mecanismo.

```

se  $V_{min} < m_{dg} < V_{max}$  {
     $p_m = \text{interpolação}(m_{dg}, V_{min}, V_{max}, p_{m\min}, p_{m\max});$ 
     $p_c = \text{interpolação}(m_{dg}, V_{min}, V_{max}, p_{c\min}, p_{c\max});$ 
}

```

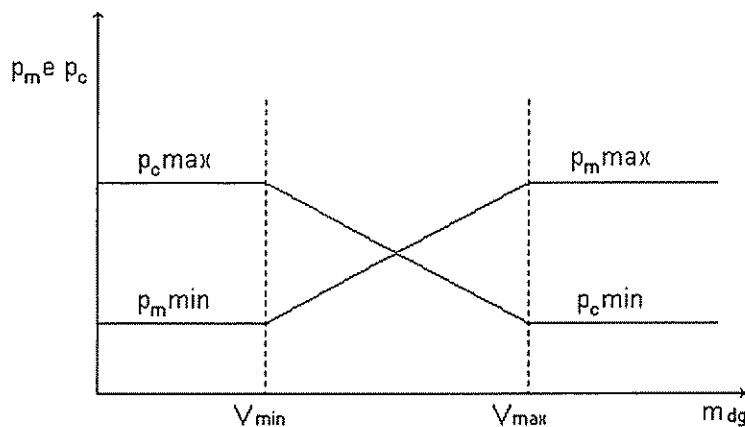


Figura 3-9: Variação dinâmica dos operadores segundo o método DF.

Na verdade, as duas variantes têm o mesmo objetivo que é a manutenção de m_{dg} interior à faixa de controle. A diferença básica é a ação externa e interna das

adaptações. Espera-se que cada método seja mais efetivo de acordo com sua oportunidade de atuação. Assim sendo, a faixa de valores V_{\min} e V_{\max} deverá ser mais estreita para FF e mais longa para DF.

A Figura 3-10 apresenta a variação da diversidade (m_{dg}) de um SSGA (este GA será apresentado na seção 3.14) frente à adaptação DF para um problema simples. Na Figura 3-10 nota-se que a magnitude $p_{m\max}$ influí diretamente no valor de m_{dg} .

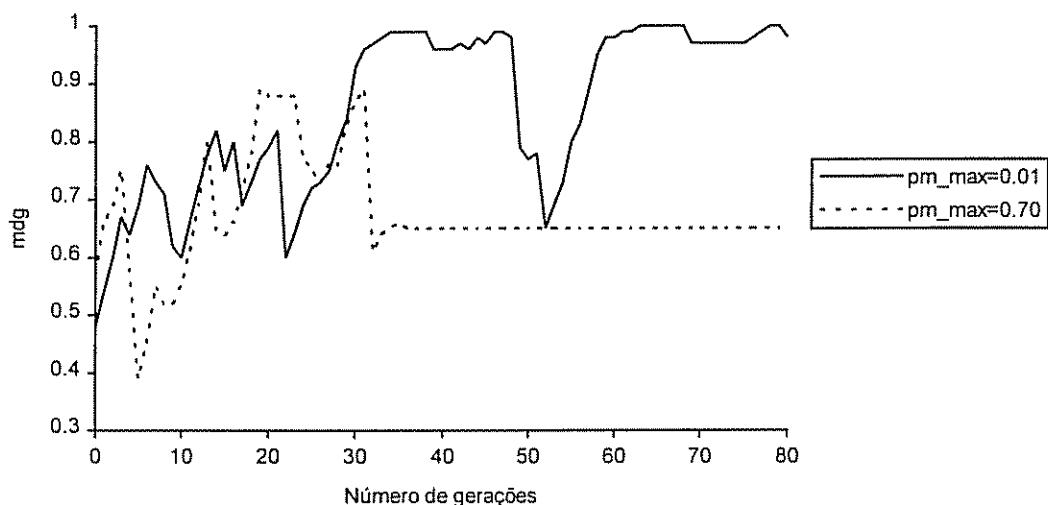


Figura 3-10: Evolução de m_{dg} ao longo das gerações.

A adaptação dinâmica dos operadores genéticos é assunto muito importante aos GAs e ainda pouco explorado na literatura.

3.12 Critérios de Convergência

Os algoritmos estocásticos não possuem nenhum artifício que indique se a solução encontrada é global. Assim sendo, os GAs continuariam operando ao longo das gerações, buscando o melhor indivíduo, mesmo que já o tivesse encontrado. A Figura 3-11 mostra o problema. Nota-se que o algoritmo convergiu a partir da geração 23. Portanto, a definição de bons critérios de convergência pode evitar que se perca tanto tempo de execução.

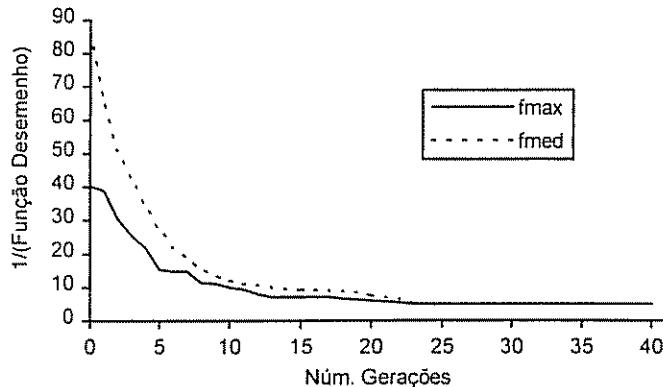


Figura 3-11: Curva característica de convergência de um GA.

Uma análise dos principais critérios de convergência para os GAs é apresentada em [45], onde o objetivo é encontrar critérios para parar os GAs e lançar um algoritmo determinístico para refinamento da solução.

O primeiro critério, apresentado aqui, é a terminação por *número máximo de gerações*. Nesse esquema, quando os GAs alcançam uma determinada geração predefinida, o algoritmo é finalizado, apontando como solução o melhor indivíduo até então. Para um dado problema, obtém-se o valor ideal de gerações, após a análise de convergência para diversas simulações. Esse número ideal de gerações pode ser difícil de ser obtido, pois sendo os GAs uma técnica estocástica, cada execução tem um caminho diferente.

O segundo critério seria finalizar o algoritmo pela *convergência do melhor indivíduo*, que seria feita usando os valores de desempenho dos melhores indivíduos da atual geração $f_{\max(\text{atual})}$ e da geração anterior $f_{\max(\text{anterior})}$, como pode ser visto na equação 3-18.

$$f_{\max(\text{atual})} - f_{\max(\text{anterior})} \leq \varepsilon \quad (3-18)$$

Se esta situação ocorrer n vezes consecutivas, considera-se que o algoritmo convergiu para uma solução. Os resultados não são tão satisfatórios quando o problema possui muitas soluções locais, porque é natural que o algoritmo fique preso nas regiões dessas soluções locais por algumas gerações, ou seja, $f_{\max(\text{atual})} = f_{\max(\text{anterior})}$. Contorna-se o problema aumentando n .

O último critério apresentado é a finalização dos GAs por *convergência da população*, no qual analisa-se a proximidade do valor de desempenho médio da população f_{med} em relação ao melhor valor de desempenho f_{\max} . Quando $f_{\text{med}} \rightarrow f_{\max}$, significa que a população encontra-se homogênea, ou seja, possui todos indivíduos com código bem parecido. Quando a razão f_{med}/f_{\max} atingir certo fator de

homogeneidade, não há mais diversidade genética para ser explorada e qualquer ganho é devido à mutação, assim não há necessidade de prosseguir com a execução do algoritmo. A equação 3-19 mostra o critério.

$$1 - \frac{f_{\text{med}}}{f_{\text{max}}} \leq f_{\text{cp}} \quad (3-19)$$

onde f_{cp} é o fator de convergência da população. Pode-se considerar que a população está homogênea quando $f_{\text{cp}} \leq 0.01$.

Em algumas situações, esse critério não funciona bem, por exemplo, quando o valor da probabilidade de mutação p_m é grande, pois isso implicaria em alta taxa de introdução de características genéticas, impossibilitando, portanto, que a população fique homogênea. Outra situação em que esse critério falha aparece quando o GA em questão é o SGA (a seção 3.14 descreve o SGA e outros dois tipos de GA), pois a cada geração o SGA introduz uma nova população proveniente do processo de seleção e de operações genéticas, como por exemplo, cruzamento e mutação. Opcionalmente, no SGA, mantém-se o melhor indivíduo da geração anterior. Assim, sempre haverá diversidade dentro da população de maneira que a equação 3-19 não se verificará facilmente.

Cada um dos critérios descritos acima tem os seus problemas. Portanto, a saída é utilizar mais de um deles.

3.13 Medindo o Desempenho dos GAs

Uma boa maneira de analisar o desempenho dos GAs ao longo das gerações é através da visualização gráfica da evolução dos valores máximo e médio da função desempenho ao longo do número de gerações. Por exemplo, com as curvas de f_{med} e f_{max} , tem-se um medidor simples de diversidade genética dentro da população, ou seja, o quanto uniforme se encontra a população. Quando f_{med} aproxima-se de f_{max} , significa que há pouca diversidade genética dentro da população, o que pode inibir a procura de novos pontos no espaço de otimização. Verificando-se somente a curva de f_{max} , determina-se a tendência do algoritmo de estagnar em pontos locais do domínio de procura. Entre outras, uma curva interessante envolve o desvio padrão dos desempenhos dos indivíduos da população.

De Jong, em [6] e [7], descreveu dois novos métodos estatísticos para a avaliação do desempenho dos GAs, que são as *on-line* e *off-line performances*, os

quais também são analisados em relação a cada geração e serão explicados a seguir.

On-line performance: Proposta com a intenção de favorecer as aplicações em tempo real, pois ela tenta definir como está o comportamento médio dos integrantes da população em relação às médias anteriores. Matematicamente ela pode ser descrita pela equação 3-20.

$$x = \frac{1}{T} \sum_{t=1}^T f_{\text{med}}(t) \quad (3-20)$$

onde $f_{\text{med}}(t)$ é desempenho médio na geração t e T indica o número da geração na qual o algoritmo se encontra.

Off-line performance: Mede a convergência do algoritmo. Muitas funções de desempenho podem ser simuladas e as melhores alternativas são usadas posteriormente para determinar algum critério de parada. Neste procedimento, mostrado pela equação 3-21, registra-se a média das melhores soluções, geração após geração.

$$\bar{x} = \frac{1}{T} \sum_{t=1}^T f^*(t) \quad (3-21)$$

onde $f^*(t)$ é o melhor desempenho até a geração t e T indica o número da geração na qual o algoritmo se encontra.

Todos os mecanismos descritos nesta seção têm a finalidade de analisar como os GAs se comportam ao longo das gerações para determinado problema. Mas, às vezes, o objetivo é fazer comparações de desempenho entre diferentes implementações computacionais de GAs, ou mesmo, comparar um GA com outro algoritmo de otimização qualquer. Nesses casos, o artifício de verificação é o número de cálculos de função desempenho.

3.14 Tipos Diferentes de GAs

O primeiro GA foi proposto (Holland [18]) sem o uso de escalonamento, com o método da Roleta, usando apenas os operadores cruzamento, mutação e inversão, sendo o cruzamento com apenas um ponto de corte. Ao longo do tempo, esse GA evoluiu em relação aos mecanismos de seleção, escalonamento, variados tipos de cruzamento, entre outros, dando origem ao termo "Algoritmos Genéticos". No entanto, estes métodos alternativos não modificaram a estrutura algorítmica do GA

inicial. Entretanto, De Jong [7] introduziu algumas mudanças significativas na formação das populações, originando os GAs de sobreposição da população, como por exemplo o *Crowding GA* e, Whitley [47] criou o *Steady State GA*. Neste trabalho, além do SGA, também serão tratados os GAs RGA (generalização do estilo *Crowding*) e o SSGA (*Steady State*).

Simple GA (SGA): Trata-se de uma das versões iniciais do GA, descrita por Goldberg [11]. Nessa implementação, a nova geração substitui inteiramente a anterior, com uso apenas da seleção e dos operadores genéticos cruzamento e mutação. A população possuirá sempre número fixo n_{pop} de indivíduos. A Figura 3-12 mostra o procedimento do SGA.

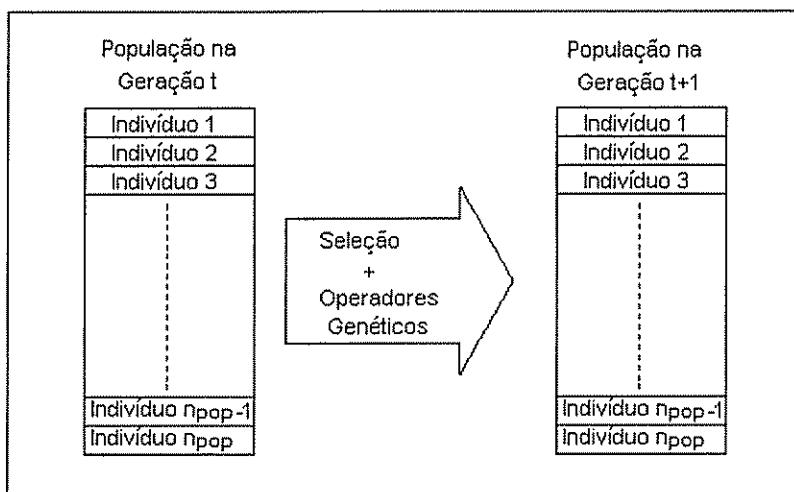


Figura 3-12: Esquema de reposição populacional do SGA.

Em cada nova geração do SGA, pode ser que não apareça nenhum indivíduo com performance superior ou igual a da geração anterior. Se isso ocorrer, a performance do algoritmo cai do patamar que já havia alcançado. Outro problema surge em gerações mais avançadas, onde a população contém várias cópias de bons indivíduos. Se por acaso surgir um melhor que os atuais, este corre o risco de não ser selecionado, justamente pela grande chance proporcionada aos indivíduos com mais cópias. Contornam-se estes dois problemas adicionando ao SGA a estratégia *elitista*, que guarda, para a próxima geração, a melhor solução. O importante para o SGA é manter a característica de reposição total ou quase (com elitismo) da população após cada geração.

Steady State GA (SSGA): Semelhante ao SGA, no SSGA, o tamanho da população é constante, mas há grande diferença quanto ao tipo de reposição dos indivíduos na geração seguinte. Nesse esquema, somente uma porcentagem da população será substituída durante cada geração. Ficarão intactas para a nova

iteração $n_{pop}*(1-G)$, sendo n_{pop} o número de indivíduos da população e G é o intervalo (gap), tal que $G \in [0,1]$. O intervalo G representa uma porcentagem de substituição. A Figura 3-13 mostra a estrutura do SSGA. Para o caso particular $G=1$, tem-se que SSGA=SGA.

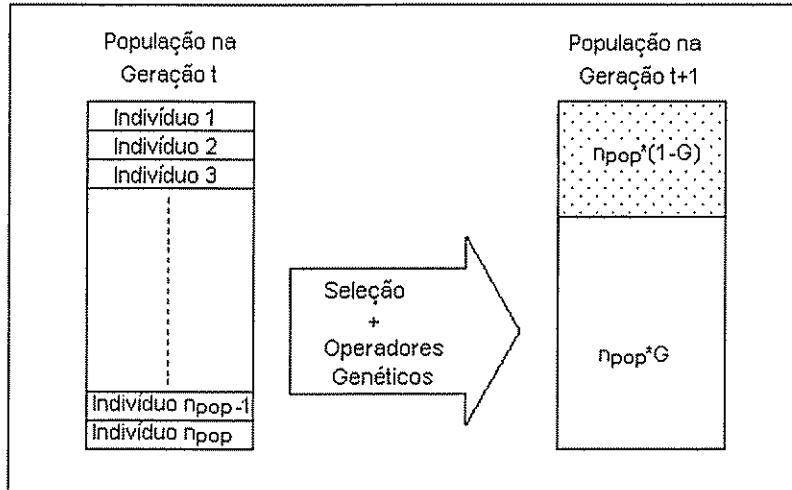


Figura 3-13: Esquema de reposição populacional do SSGA.

Replacement GA (RGA): Quanto ao RGA, a cada nova geração apenas um ou poucos indivíduos são gerados e introduzidos na população através da substituição de outros indivíduos. Inserindo poucos indivíduos a cada geração, pode-se achar que o RGA está deixando de aproveitar a informação existente, mas isto não é verdade. Aproveita-se a riqueza da população, só que em muito mais gerações.

Uma outra questão refere-se a estratégia de substituição. Por exemplo, no caso do RGA, somente um ou dois indivíduos entrarão na próxima geração. Estes novos indivíduos substituirão a quem? Há várias opções: novos membros podem substituir aleatoriamente (com estratégia elitista), os piores da população, os melhores, os pais, os mais parecidos (estilo *Crowding*) ou substituir seguindo algum critério de classificação. Esta questão não foi tratada neste trabalho. A estratégia de substituição usada aqui é a de que novos membros substituem os piores da população corrente. A mesma estratégia de substituição será utilizada para o SSGA, ou seja, os novos indivíduos substituirão os piores n_{pop}^*G da população.

3.15 Técnicas Híbridas

Os GAs têm uma estrutura algorítmica muito versátil e isso possibilita o seu acoplamento a outras técnicas, como por exemplo, a associação aos Métodos Determinísticos, aos Algoritmos Nebulosos e as Redes Neurais. Esta associação de

GAs com outras técnicas denomina-se *técnicas híbridas*. Analisaremos nesta seção o acoplamento com um Método Determinístico.

Num algoritmo de otimização, duas características são importantes para encontrar a solução global. A primeira, é achar a região onde está localizada esta solução e a segunda, como explorar a região para chegar ao cume do pico (convergência final). Entre os dois itens, os GAs são muito mais poderosos no primeiro, onde a teoria dos *building blocks* explica como eles exploram os hiperplanos. Por outro lado, os GAs têm uma deficiência na convergência final. Isso porque, na fase final, a população compartilha praticamente de mesmo código genético, ou seja, seus indivíduos têm a estrutura bem parecida e com muitas cópias. A partir deste estágio, qualquer melhoria deve-se praticamente à mutação, que, além de ocorrer com baixa probabilidade, também age aleatoriamente. A solução para o problema de convergência final pode ser o acoplamento de GAs com Métodos Determinísticos. A razão é que esses métodos têm a forte característica de rápida convergência final na região em que eles se encontram.

Otimização local de uma ou mais variáveis é um assunto bem vasto, boas referências de pesquisa são [1], [9], [24] e [41]. Então, basta escolher um método de preferência e juntá-lo ao código de um GA. Esse seria um procedimento generalizado, mas, para maior desempenho, é indispensável a adequação de técnicas específicas ao problema em questão.

As maneiras de inserir técnicas locais na estrutura de GAs são limitadas pela criatividade. Mas uma boa implementação seria a que, a cada n gerações, o melhor indivíduo fosse o ponto de partida para o lançamento da técnica determinística. Após encontrar a solução local, o indivíduo, com sua estrutura cromossômica aprimorada, é recolocado na população. O algoritmo descrito abaixo mostra o processo.

```

Algoritmo Genético + Técnica Determinística{
    Definindo parâmetros
    Inicializar população aleatória
    Enquanto não alcançar critério de parada faça {
        Avaliar os indivíduos da população
        Executar operadores genéticos
        Se número de gerações é múltiplo de n faça {
            Selecionar melhor indivíduo
            Aplicar técnica determinística
            Inserir solução aprimorada na população
        }
    }
}
```

Os trabalhos que fizeram uso deste tipo de técnica híbrida buscaram algoritmos determinísticos que apresentam bons resultados em termos de precisão, eficiência e robustez. Por exemplo, Vasconcelos & Outros em [43] e [45] utilizaram como técnica determinística o método *Lagrangeano Aumentado*, enquanto que Mendes & Outros [26] utilizaram o método do *Elipsóide*. No Capítulo 4 deste trabalho utilizou-se como técnica local o método *BFGS*. O inconveniente dessas técnicas determinísticas é a necessidade de cálculo de derivadas. Uma possibilidade de fugir do cálculo de derivadas é utilizar o método de *Rosenbrock*. Outra técnica local que pode ser aplicada é a do *gradientlike-bitwise improvement* proposta por Goldberg em [11]. Nela são tomadas como base os dois melhores indivíduos da atual iteração, depois disto varre-se bit a bit cada uma, mudando o valor do bit e avaliando o desempenho do indivíduo. No final, escolhe-se a melhor solução para repô-la na população. O problema desta última técnica local é que ela é dispendiosa computacionalmente.

3.16 Formação de Nichos e de Subpopulações

Em sistemas naturais, define-se nicho como uma porção restrita de um *habitat* onde vigem condições especiais de um ambiente e as espécies que nele vivem disputam seus recursos. Cada espécie possui um grau de desenvolvimento de acordo com a sua adequação ao nicho em que vive. O resultado é que mesmas espécies em *habitats* distintos desenvolvem suas características diferentemente, formando subespécies. É a evolução. Pode-se introduzir o mesmo conceito de nichos e espécies nos Algoritmos Genéticos. Dessa forma, pode-se trabalhar com subpopulações ao invés de uma inteira, propiciando, assim, o aparecimento e desenvolvimento de novas características.

Para entender melhor porque é interessante encorajar a formação de nichos em GAs, considere a ação de um GA sem divisão da população sobre a função da Figura 3-14. Como os indivíduos da primeira população são gerados aleatoriamente, espera-se uma razoável distribuição ao longo do domínio. Com o passar das gerações, a população se concentra na região de apenas um pico, excetuando um ou outro indivíduo que sofreu mutação e se encontra em qualquer lugar do domínio. É interessante notar que há vários máximos de igual magnitude e o algoritmo encontra apenas uma solução. Esta convergência para apenas uma região é denominada por *tendência genética*.

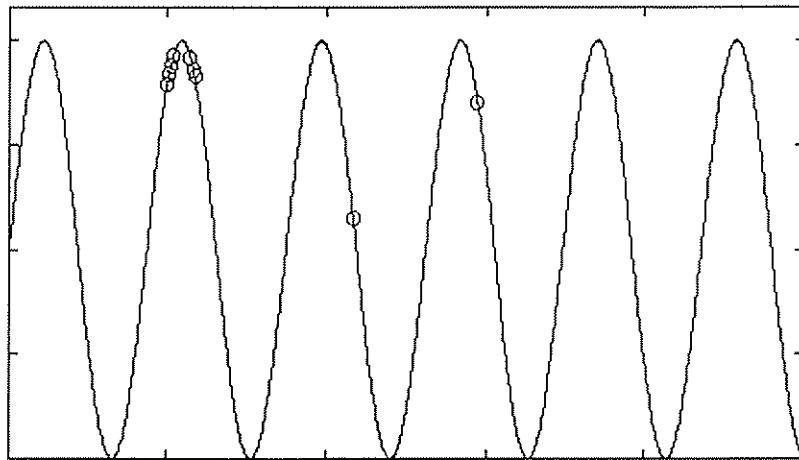


Figura 3-14: Distribuição da população após algumas gerações, numa função com picos de alturas iguais.

Agora, se aquele GA teve o comportamento tendencioso em relação à função da Figura 3-14, imagine a sua atuação sobre uma função que tenha a forma da Figura 3-15. Certamente, todos os indivíduos se posicionariam no pico mais alto rapidamente.

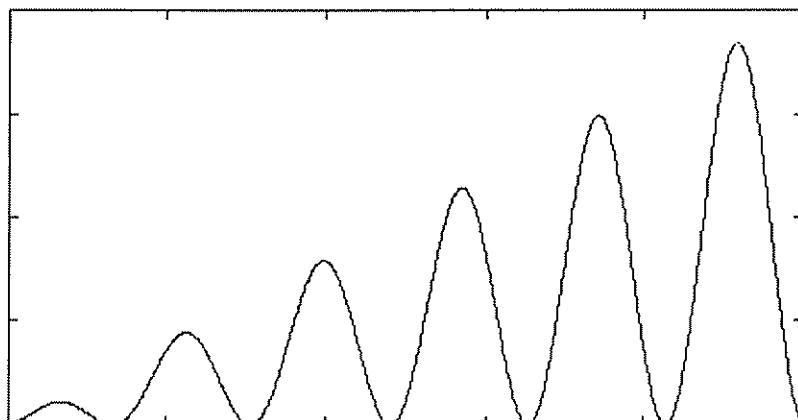


Figura 3-15: Função com picos de alturas diferentes.

Muitas vezes, a melhor solução pode significar um projeto mais arrojado e com grande uso no futuro, mas para um tempo imediato, talvez ela seja inviável. Nestes casos é interessante ter em mãos outras soluções possíveis, e assim tomar uma decisão a respeito da mais conveniente. As técnicas de nicho tentam condicionar a sobrevivência aos indivíduos que estão fora do grupo de elite.

Neste trabalho de dissertação são descritas duas técnicas de formação de nichos, mas somente a primeira será analisada numericamente.

No primeiro esquema, descrito em [10] e [11], analisando do ponto de vista de sistemas naturais, a idéia é bem simples. Picos no espaço são tratados como recursos e os indivíduos que neles se encontram têm que repartir os recursos. O

superpovoamento numa região implica em menor quantidade de recurso para cada indivíduo. A escassez de recursos conduz os indivíduos a migrarem para outras regiões. Voltando ao mecanismo artificial, devido à tendência genética, a população vai se concentrando ao longo das gerações para apenas uma região. Com o intuito de diminuir a densidade populacional em determinado local, o primeiro passo foi a elaboração de uma *função de partilha (sharing function)* para medir o *grau de vizinhança*, ou seja, quantificar a proximidade de um indivíduo em relação aos outros da população (vizinhança). Depois disso, o método de seleção, analisaria o indivíduo por uma aptidão aparente $f_{s,i}$ (relativo ao indivíduo i) que dependesse do *grau de vizinhança*.

A aptidão aparente é obtida da divisão do desempenho próprio f_i , pelo *grau de vizinhança* total do indivíduo i , o qual é calculado em relação a todos os integrantes da população (que possui n_{pop} membros), como pode ser notado na equação 3-22:

$$f_{s,i} = \frac{f_i}{\sum_{j=1}^{n_{pop}} s(d_{ij})} \quad (3-22)$$

onde s é a *função de partilha* e d_{ij} representa uma função de distância entre os indivíduos i e j . O termo d_{ij} fica melhor definido como $d_{ij}=|x_i-x_j|$, sendo que os termos x podem ser simplesmente retorno da função desempenho ou a descrição do fenótipo (análise bit a bit) dos indivíduos vistoriados. O número total de cálculos da *função de partilha* por geração é n_{pop}^2 .

Para inserir esta técnica de nicho em GAs, basta mudar o método de seleção, que, ao invés de escolher os mais aptos, escolherá os indivíduos de acordo com sua aptidão aparente proveniente do somatório da *função de partilha*. Para cada par de indivíduos analisados, quanto maior a proximidade, mais perto de 1 é o retorno da *função de partilha*. A equação 3-23 representa um conjunto de *funções de partilha (power law functions)* descritas em [10] e [11].

$$\begin{aligned} s(d_{ij}) &= 1 - \left(\frac{d_{ij}}{\sigma_s} \right)^\alpha && \text{se } d_{ij} < \sigma_s \\ s(d_{ij}) &= 0 && \text{se } d_{ij} \geq \sigma_s \end{aligned} \quad (3-23)$$

onde a constante σ_s indica o menor *grau de vizinhança* entre dois indivíduos e a constante $\alpha (> 0)$ tem a função de penalizar os indivíduos mais próximos. Para o caso especial de $\alpha=1.0$, a equação 3-23 descreve a *função de partilha triangular*.

Essa função favorece a proporcionalidade da quantidade de indivíduos em relação à importância da região. A Figura 3-16 mostra o comportamento de três possíveis *funções de partilha* provenientes da equação 3-23.

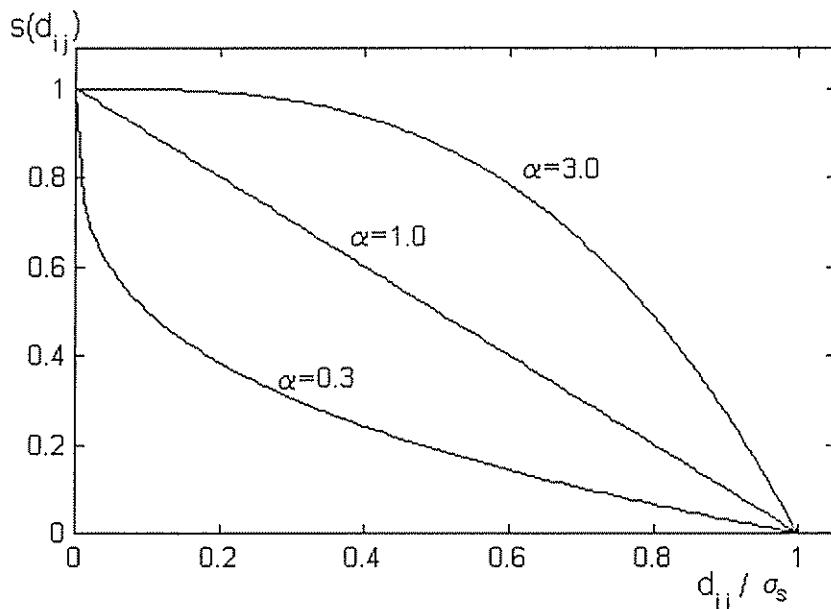


Figura 3-16: Representação gráfica das *funções de partilha* da equação 3-23.

Aplicando a *função de partilha* triangular no problema da Figura 3-15, um possível resultado da distribuição da população (com 24 indivíduos) de um GA encontra-se na Figura 3-17 que vem a seguir.

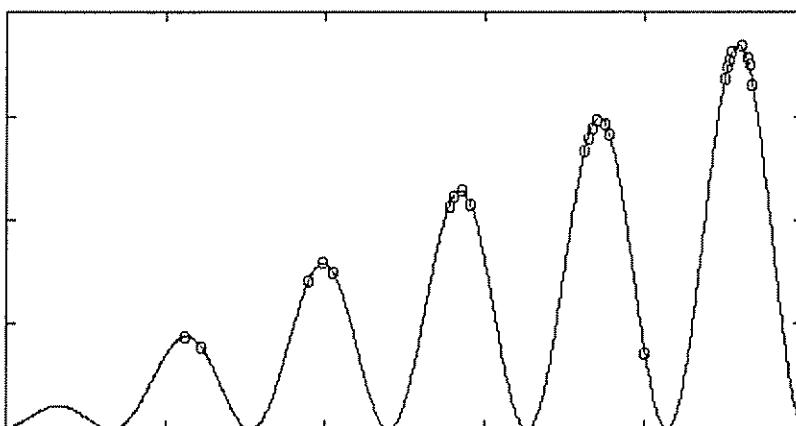


Figura 3-17: Distribuição da população após algumas gerações, numa função com picos de alturas diferentes.

O cálculo da aptidão aparente $f_{s,i}$ traz um custo adicional de processamento. Preocupado com este problema, Goldberg propôs que cada indivíduo realizasse o cálculo da *função de partilha* com apenas k indivíduos ($k \ll n_{pop}$), com os quais seria calculado um valor médio de *função de partilha*. O somatório da equação 3-22 seria estimado por $(n_{pop}-1)*u_i + 1$, onde u_i corresponde ao valor médio da *função de partilha* relativa ao indivíduo i .

Outro mecanismo de nicho foi proposto por Spears em [36], denominado de SSS (*Simple Subpopulation Schemes*). Esse mecanismo cria subpopulações via uso de etiquetas de identificação (*labels*), as quais são anexadas aos indivíduos. Portanto, os indivíduos que tiverem mesma etiqueta, pertencem a mesma subpopulação. As etiquetas são formadas por uma cadeia binária de comprimento ℓ' . Dessa forma, é possível a formação de $2^{\ell'}$ agrupamentos distintos de indivíduos. A ilustração a seguir mostra quatro indivíduos de comprimento $\ell=7$ bits e suas respectivas etiquetas de comprimento $\ell'=3$ bits. Pelas etiquetas, observa-se que os indivíduos 1 e 4 pertencem a mesma subpopulação.

	ℓ				ℓ'			
1	-	1	1	1	1	0	0	0
2	-	0	0	1	0	1	0	1
3	-	1	1	0	0	0	1	0
4	-	1	1	1	1	0	1	1

O processo inicia-se no momento da criação da população. Associado a cada indivíduo, as etiquetas são criadas aleatoriamente. Como no mecanismo de nicho de Goldberg, os indivíduos serão selecionados de acordo com uma aptidão aparente. A performance aparente de cada indivíduo é medida a partir da aptidão individual dividida pelo número de integrantes do seu nicho.

Em seu trabalho, Spears analisou dois mecanismos diferentes: SSS1 e SSS2. O SSS1 usa as etiquetas para restringir cruzamento. Só podem cruzar indivíduos de mesma subpopulação. O SSS2 se diferencia do SSS1 por permitir cruzamento entre indivíduos de subpopulações vizinhas, sendo que cada subpopulação tem duas subpopulações vizinhas, uma à direita e outra à esquerda. Outra diferença surge no cálculo da aptidão aparente. Enquanto que no esquema SSS1 a aptidão aparente é obtida do valor do desempenho individual pelo número de indivíduos de mesma etiqueta, no esquema SSS2 a aptidão aparente é calculada relativamente ao número de componentes das três subpopulações vizinhas.

Comparando os métodos de Goldberg e Spears, o segundo apresenta-se muito mais viável para grandes populações, visto que a classificação por etiquetas tem um custo muito inferior que a outra técnica, a qual executa n_{pop}^2 avaliações de *função de partilha* em cada geração. Por outro lado, o número de subpopulações é variável, enquanto que, no método de Spears, o número máximo de subpopulações é fixo e determinado por $2^{\ell'}$. Finalmente, o método de Goldberg é mais preciso, porém de custo mais elevado.

A formação de nicho conduz os GAs a bons resultados. Agora, produz resultados ainda melhores se as subpopulações forem estáveis, Deb & Goldberg em [8] definem que, se uma subpopulação com 100 membros sobrevive por mais de 200 gerações, ela é estável. Uma subpopulação estável pode ser obtida fazendo-se restrição do cruzamento ou, em outras palavras, somente permiti-lo entre membros do mesmo nicho. Para a técnica de Spears, simplesmente verificam-se as etiquetas no ato do cruzamento, enquanto que para o método de Goldberg associam-se etiquetas binárias (*tag bits*) que indicam uma seqüência que deve conter o outro indivíduo para haver cruzamento (maiores detalhes, veja [11]). Spears também analisou a mutação agindo sobre a etiqueta, permitindo a mudança de nicho. Restrição do cruzamento é um item que não será verificado nesta dissertação.

3.17 Redução do Espaço de Procura

Para resolução um problema de otimização por intermédio de GAs, deve-se limitar um domínio de procura para cada variável e, durante toda a pesquisa para encontrar a solução global, os GAs continuam verificando todo o espaço delimitado na fase inicial. Se o melhor indivíduo da população, quando um GA converge, não corresponde à melhor solução, pelo menos esse indivíduo define uma subregião em torno dele, na qual há grandes chances de se encontrar a solução global. Então, se os GAs têm facilidade de encontrar boas subregiões, por que não explorá-las? Com o propósito de responder a essa questão, desenvolveu-se, nesta dissertação, uma técnica de redução de intervalo de procura. Escolheu-se reduzir o espaço exponencialmente de acordo com a equação 3-24, que vem a seguir.

$$I_n = I_0(1+j)^n \quad (3-24)$$

onde I_0 corresponde ao intervalo inicial de procura, I_n é o intervalo na n -ésima redução e j a taxa de redução. Depois de calculado I_n , o novo espaço de procura é determinado tendo como base cada uma das variáveis do melhor indivíduo. A Figura 3-18 mostra o procedimento.

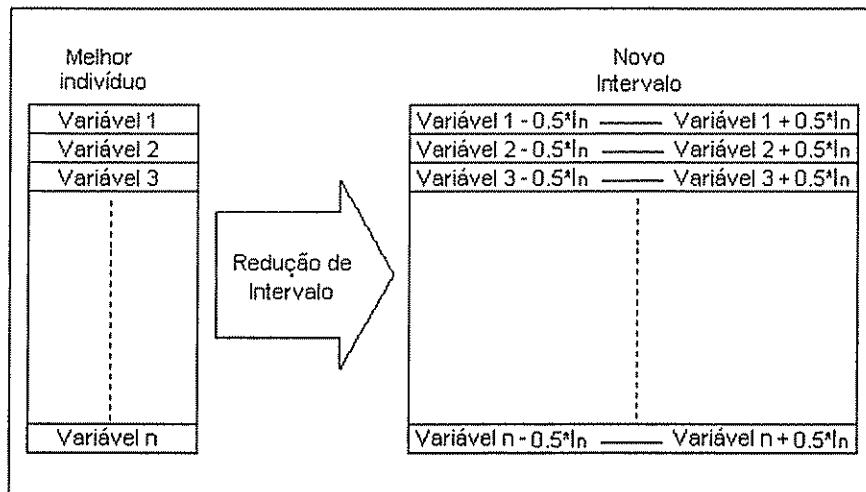


Figura 3-18: Mecanismo de redução de intervalo.

Se por acaso durante a redução, algum limite novo de algum intervalo ficar de fora do intervalo inicial, ele será redefinido como o limite inicial. Depois de definido o novo intervalo, guarda-se o melhor indivíduo e a seguir gera-se uma nova população aleatória, dando continuidade ao processo evolutivo. Quando essa nova população estiver um pouco homogênea, introduz-se o melhor indivíduo guardado. Essa espera tem como finalidade gerar bons concorrentes ao melhor indivíduo do intervalo anterior. O número de bits de um intervalo é fixo e, com isso, a cada diminuição do intervalo haverá melhor resolução por variável, proporcionando aumento de precisão.

A equação 3-24 mostra matematicamente como esta redução pode ser feita, no entanto, a redução de intervalo é um mecanismo que precisa ser analisado com muito cuidado, pois, se uma variável é remapeada num subdomínio e o ponto solução do problema se encontra fora dele, o algoritmo pode ter dificuldades em encontrá-lo. Se a redução de intervalo for feita continuamente, certamente o espaço de procura final seria um micro domínio para cada variável, o que conduziria à situação anterior. Um alto valor de redução, poderia conduzir o algoritmo a sair definitivamente do espaço da solução global de uma única vez. Portanto, é necessário tomar algumas precauções de forma a reduzir o intervalo com mais segurança.

Tentando esquivar destes problemas, a implementação da técnica de redução no código de GAs foi um pouco complicada. Para acionar o mecanismo de redução de intervalo, três condições devem estar satisfeitas. Essas condições, quando satisfeitas, são assinaladas por *indicadores (flags)* ativos. O primeiro *indicador* ativo aponta que o melhor indivíduo é o mesmo das últimas n gerações, ou seja, o

algoritmo está estagnado num ponto por n gerações consecutivas. O *indicador 2*, quando ativo, mostra que há pouca diversidade genética dentro da população, existindo, portanto, menores chances do algoritmo sair da região definida pelo melhor indivíduo. O medidor de diversidade m_{dg} é o mesmo usado na adaptação dinâmica dos operadores genéticos. O *indicador 3* encontra-se ativo enquanto o intervalo de procura não chegar ao valor mínimo estipulado. Com os três *indicadores* ativos, antes de executar a redução, guarda-se o melhor indivíduo. Usando a equação 3-24 obtém-se o valor de I_n e utiliza-se cada variável x_i do melhor indivíduo para definir o novo intervalo, que será $x_i \pm I_n/2$. Depois disso, reinicia-se o processo evolutivo com nova população aleatória. Quando m_{dg} indicar certo valor de uniformização, insere-se o melhor indivíduo do intervalo anterior, no novo intervalo. O processo de redução continua até o *indicador 3* ser desativado. A Figura 3-19 mostra o comportamento de um SSGA com o mecanismo de redução de intervalo, sobre uma função de minimização genérica, ao longo das gerações.

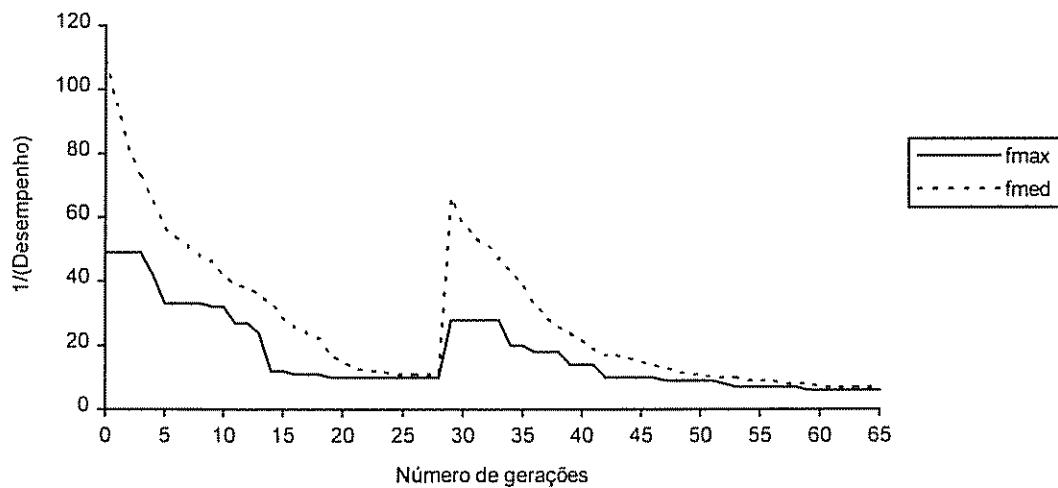


Figura 3-19: Comportamento de um SSGA com o mecanismo de redução de intervalo.

Analizando a Figura 3-19, observa-se que, a partir da vigésima geração $f_{med} \rightarrow f_{max}$ ($m_{dg} \geq 1$), e portanto, a população está ficando bem homogênea. Nota-se também que, entre as gerações 18 e 28, o SSGA não obteve melhorias quanto ao valor de f_{max} . Com $m_{dg} \geq 1$ e f_{max} estagnado, dificilmente o algoritmo sairá desse patamar de solução. Então, aplica-se o mecanismo de redução de intervalo.

No Capítulo 4 estão apresentados o comportamento típico do RGA, SGA e SSGA (veja Figura 4-5, Figura 4-6 e Figura 4-7) segundo a evolução dos valores de desempenho médio e máximo encontrados na população. Nas figuras observa-se que o SGA é inadequado à aplicação deste mecanismo de redução de intervalo,

pois há muita flutuação dos valores de f_{med} , deixando portanto, o medidor de homogeneidade da população m_{dg} impreciso.

Um problema que inviabiliza o uso do mecanismo da redução de intervalo aparece quando o número de variáveis do problema é grande. Nesses casos, é muito difícil definir as subregiões com segurança. No Capítulo 4, a redução de intervalo é um dos assuntos tratados nos testes numéricos.

3.18 Conclusão

O SGA, descrito por Goldberg em [11], com o método de seleção e os operadores cruzamento e mutação ganhou, no decorrer dos anos, muitas inovações, entre elas, vários tipos de cruzamento, seleção, técnicas de suavização da convergência (escalonamentos, critérios de adaptação dinâmica dos operadores e métodos de formação de nicho), e também, a inclusão de outras técnicas ao código dos GAs (técnicas determinísticas). Tão importante quanto estas inovações, é a codificação das variáveis, introdução de restrições e modelamento da função objetivo de forma a facilitar o processo evolutivo dos GAs. Outro ponto importante é a evolução do próprio Algoritmo Genético, que a partir do GA de Holland [18], surgiram também vários outros como o SGA, RGA e o SSGA, cada um com características próprias.

4. Análise Numérica de Algumas Técnicas Genéticas

4.1 Introdução

Este capítulo trata da análise numérica dos métodos descritos no capítulo anterior, aplicados aos três tipos de GAs, os quais se diferenciam pela quantidade de indivíduos gerados de uma geração para outra. No caso do SGA, a nova população substitui completamente a anterior, sendo uma opção manter o melhor indivíduo até então. Entretanto, tratando-se do SSGA, especifica-se uma porcentagem de integrantes da população a ser substituída. Já o RGA, introduz na nova geração apenas um ou dois membros novos. O estudo dos métodos foi dividido em três partes distintas. Na primeira, serão analisadas, algumas técnicas como escalonamento, cruzamento e métodos de seleção. Nesse ponto, também será introduzido o mecanismo de adaptação dinâmica das probabilidades de cruzamento e mutação. O que se quer aqui é propor uma seqüência de testes que possibilite encontrar um grupo de parâmetros que torne os GAs mais adaptados para a resolução de um problema específico. Inicialmente, foi analisado o desempenho conjunto dos três GAs, com isso, espera-se conseguir um conjunto de parâmetros mais robusto. Em seguida, cada GA será desenvolvido separadamente, buscando assim, os métodos que se identificam mais com seu próprio tipo. Na segunda parte, as técnicas formação de nicho, redução do intervalo de procura ao longo das gerações e o acoplamento com um Método Determinístico ao corpo de um GA serão executadas como demonstração, pois seria praticamente impossível avaliar todas as combinações intrínsecas de cada método conjuntamente com outros parâmetros e métodos dos GAs. Para exemplificar, um ponto a ser tratado é o acoplamento de uma técnica determinística à estrutura de um GA. Nesse caso, um problema seria verificar mais uma técnica local e, para cada técnica, variar seus parâmetros, e assim por diante. Finalmente, na terceira parte, serão mostrados alguns resultados de pesquisadores sobre métodos descritos aqui, porém não implementados.

Para a primeira parte, os GAs foram avaliados através de três funções, com as respectivas soluções previamente conhecidas. Foram feitos uma série de testes, com o objetivo de aprimorar o desempenho dos GAs, perante essas três funções. Infelizmente, a grande diversidade de problemas de otimização impossibilita afirmar que este ou aquele mecanismo ou técnica é o melhor. Para a segunda parte, os mecanismos considerados não genéticos como redução do intervalo de procura ao longo das gerações e adição de técnica determinística, juntamente com a formação de subpopulações (nínhos), interagem com a adaptação dinâmica formando GAs mais robustos. O único inconveniente é relativo ao tipo de método de formação de nínhos, que é muito caro computacionalmente. Essas técnicas serão analisadas para apenas uma função, a mais complicada dentre as da primeira fase de teste, mas com o número de variáveis bem maior. Na terceira parte, a qual trata de trabalho de pesquisadores, geralmente a análise é feita para o SGA com elitismo e as funções usadas são as mais diversas. Mesmo assim, considerou-se válida a inclusão de seus resultados nesta dissertação.

Antes de se descrever os procedimentos de teste, serão apresentadas as funções a que os GAs serão submetidos.

Para realização de todos os experimentos deste capítulo e também do capítulo de aplicações (Capítulo 5), foi utilizado um *software* desenvolvido no MIT (*Massachusetts Institute of Technology*) chamado *GAlib*, versão 2.3.2.

4.2 Melhorando o Desempenho dos GAs

4.2.1 Funções Teste

Não há algoritmo de otimização que resolva satisfatoriamente todo tipo de problema, devido à grande variedade dos mesmos. Então, foram escolhidas três funções teste de forma a tentar agrupar algumas características comuns a uma série de outras funções. As características são de continuidade, descontinuidade e de ser multimodal com apenas uma solução global.

A primeira função, conhecida como Degrau ou F3 de De Jong [7], mostrada na Figura 4-1, tem como característica ser descontínua. A Figura 4-1 representa o caso 2D, mas a função é extensível a n variáveis (dimensões). A função Degrau é obtida a partir da Equação 4-1.

$$ff(x) = \sum_{i=1}^n \text{int eiro_arredondado}(x_i) \quad (4-1)$$

Deseja-se encontrar o mínimo para essa função, que obviamente, ocorre quando cada $x_i < 0.5$. Para a realização dos testes para análise numérica, foi tomado $n=2$ e o intervalo de procura foi definido com $x_i \in [-20, 20]$. Como os GAs trabalham em termos de maximização e nesse caso quer-se o mínimo, a função objetivo $ff(x)$ foi transformada na forma de uma função desempenho $f(x)$ como é mostrado na equação 4-2.

$$\max f(x) = \frac{1}{|ff(x)| + \epsilon} \quad (4-2)$$

onde a constante $\epsilon=10^{-6}$.

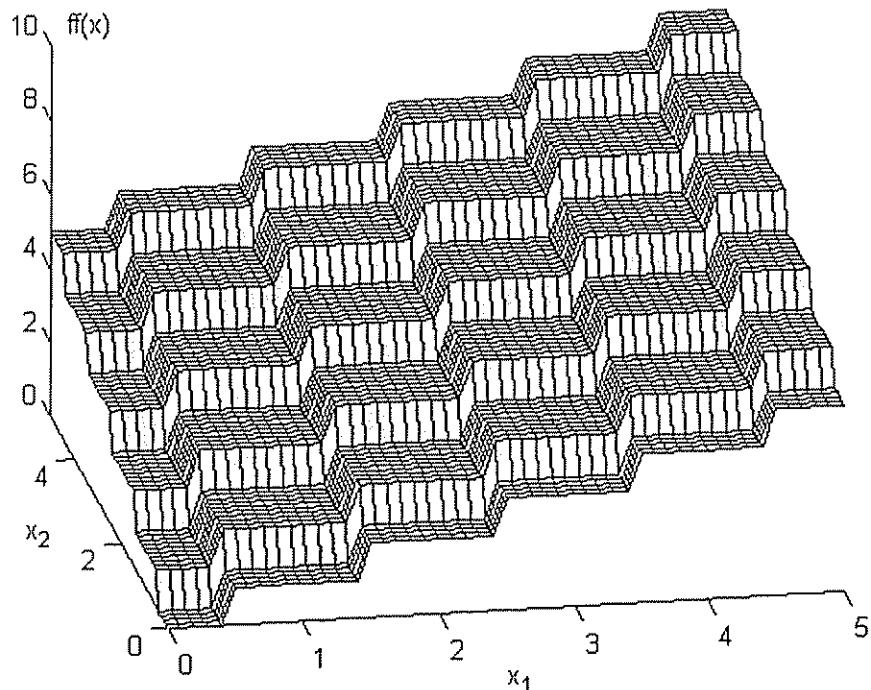


Figura 4-1: Função Degrau 2D.

A segunda função, conhecida como função de Rastrigin, é ilustrada na Figura 4-2, também para o caso de duas variáveis.

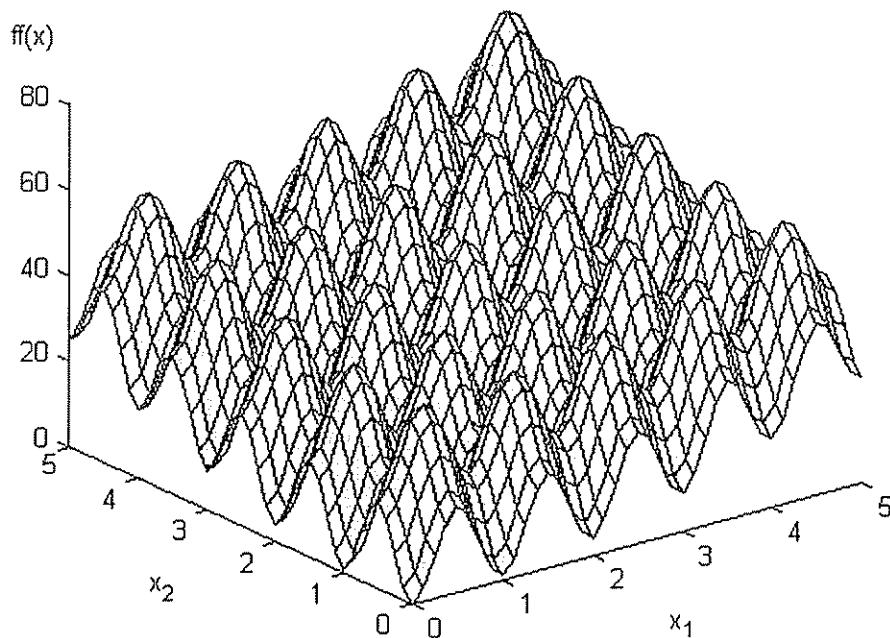


Figura 4-2: Função de Rastrigin.

O caso 2D foi o caso tratado na fase de testes. Da mesma forma que a função Degrau, essa função também pode ser extensível a n variáveis. Como pode ser visto na equação 4-3, é uma função contínua e altamente multimodal. No intervalo escolhido $x_i \in [-5.12, 5.12]$, ela possui aproximadamente 10^n mínimos.

$$ff(x) = A * n + \sum_{i=1}^n [x_i^2 - A * \cos(2 * \pi * x_i)] \quad (4-3)$$

O parâmetro A pode ser qualquer número real. O mínimo global é dado por $x_i=0.0$ para todo $i \in [1, n]$. Foi admitida como solução ótima, toda solução em que a *norma euclidiana* do vetor solução for inferior a 0.3, pois, nesse ponto, a região do mínimo já está caracterizada. Mais uma vez, o objetivo do problema é minimização. Para adequar o problema para maximização, introduziu-se a equação 4-3 na equação 4-2.

A última função (veja Figura 4-3), proveniente da adaptação da função "peaks" do Matlab, será denominada por Picos. É uma função contínua e com apenas 5 máximos, cuja forma analítica é mostrada pela Equação 4-4. Esta é uma função puramente 2D.

$$ff(x) = 3 * (1 - x_1)^2 * e^{-x_1^2 - (x_2 + 1)^2} + \left| 10 * \left(\frac{x_1}{5} - x_1^3 - x_2^5 \right) * e^{-x_1^2 - x_2^2} + \frac{1}{3} * e^{-(x_1 + 1)^2 - x_2^2} \right| \quad (4-4)$$

O domínio de procura para x_1 e x_2 está situado no intervalo $[-3;3]$. Nesse caso, o objetivo é maximizar $ff(x)$, logo $f(x)=ff(x)$. Para precisão de 4 casas decimais a

solução global acontece quando $x_1 = -0.0094$ e $x_2 = 1.5814$. Mas, será considerado que o algoritmo chegou ao máximo quando a *norma euclidiana* em relação ao ponto solução for inferior a 0.3.

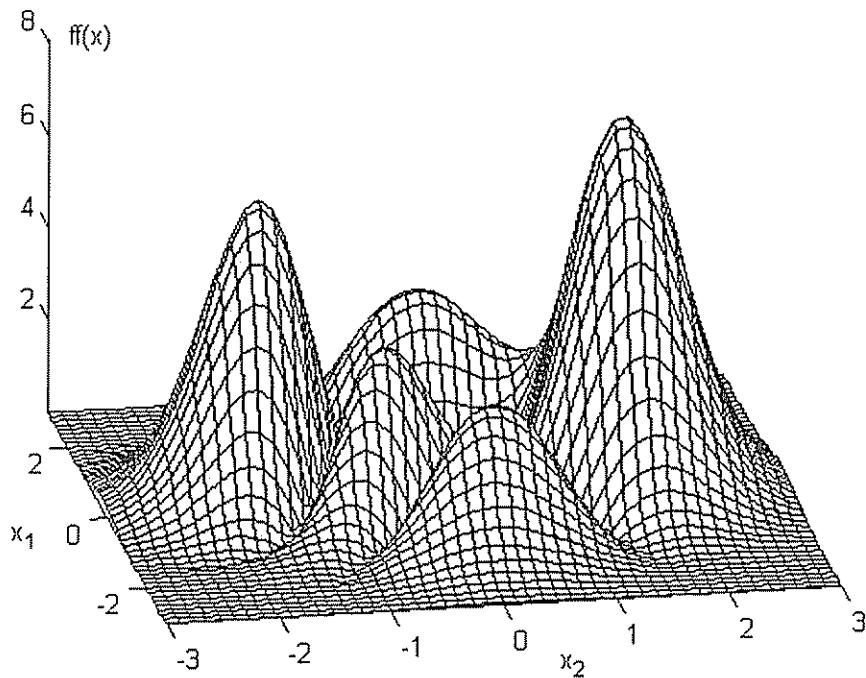


Figura 4-3: Função Picos.

A função de Rastrigin é, entre as três, a mais difícil para o algoritmo encontrar a solução global. Isso acontece por causa da grande quantidade de mínimos locais ao longo de todo o domínio. Já a função Picos é a mais simples. Espera-se com ela excelente performance dos algoritmos em todos os testes. A função Degrau possui como dificuldade a descontinuidade e também os degraus de performance, nos quais, a função objetivo retorna o mesmo valor, dificultando os GAs decidirem quais indivíduos podem ser mais, ou menos, úteis. A Tabela 4-1, que vem a seguir, tenta sintetizar algumas informações sobre as três funções teste.

Tabela 4-1
Informações gerais sobre as funções teste.

Função teste	Limite das variáveis	Solução global quando	Comprimento ℓ de cada x_i	Número máximo de cálculo de função
Degrau	$-20.0 \leq x_i \leq 20.0$	$x_i < 0.5$	18 bits ..	2050
Rastrigin	$-5.12 \leq x_i \leq 5.12$	norma euclidiana < 0.3	16 bits	2050
Picos	$-3.0 \leq x_i \leq 3.0$	norma euclidiana < 0.3	15 bits	650

A penúltima coluna indica o comprimento ℓ utilizado para cada x_i , de forma que a resolução do espaço de procura seja inferior a 0.0001. A última coluna mostra um dos dois critérios de parada dos GAs, que é o número máximo de cálculo de função permitido para a resolução de cada problema. O segundo critério, o de convergência da população, finalizará os GAs caso $1-m_{dg} < 0.01$.

4.2.2 Testando Métodos Básicos

Os Algoritmos Genéticos trabalham com um conjunto de parâmetros simultaneamente. Os valores numéricos desses parâmetros são decisivos no desempenho do algoritmo. Parâmetros como probabilidade de cruzamento e mutação já foram temas de muitos trabalhos como [7] e [14]. Grefenstette em [14], após uma extensa série de testes, concluiu que a escolha ideal dos parâmetros é um problema não linear. Além dessa não linearidade, um outro problema adicional é que todos os mecanismos dos GAs são guiados probabilisticamente, ou seja, a cada execução, o caminho utilizado para procurar a solução dificilmente seria o mesmo. Com isso, não se pode afirmar que um método é superior a outro com uma simples execução. Para aumentar a confiabilidade, cada teste foi executado 30 vezes. Talvez esta quantidade ainda não seja ideal, mas, de acordo com o número de execuções realizados nos trabalhos [5], [33] e [37] (20, 10 e 30 respectivamente), a quantidade parece ser suficiente.

Os parâmetros probabilidade de mutação e cruzamento, assim como quantidade de indivíduos na população, não foram reavaliados nesta dissertação. Foram usados os valores pertinentes ao trabalho de De Jong [7], no qual é sugerido que a população seja composta por 50 indivíduos, a mutação e o cruzamento com probabilidades 0.001 e 0.6, respectivamente. Também para realização desta fase, foram fixos a Roleta como método de seleção, o cruzamento com 2 pontos de corte e o escalonamento Linear com multiplicador 1.2. A Tabela 4-2 agrupa esse conjunto inicial de parâmetros. Esse grupo de parâmetros (com poucas variações) tem sido usado como padrão em trabalhos de muitos pesquisadores que apenas utilizam os GAs como ferramentas de otimização. Observando esse conjunto de parâmetros e métodos “consagrados”, será possível melhorá-lo? Para tentar responder essa pergunta, criou-se essa seção. Quanto ao número de gerações este foi ajustado de forma que o número máximo de cálculo de função descrito na Tabela 4-1 possa ser atingido.

Tabela 4-2	
Conjunto de parâmetros utilizados inicialmente nos GAs.	
Parâmetro	Conteúdo
tamanho da população	50 indivíduos
probabilidade p_m	0.001
probabilidade p_c	0.6
método de seleção	Roleta
método de cruzamento	2 pontos
método de escalonamento	Linear(1.2)

Inicialmente, os GAs foram avaliados perante suas características próprias. No SGA, foi observado se melhores resultados aparecem usando estratégia elitista ou

sem tal estratégia. No SSGA, procurou-se pelo valor do melhor intervalo G e, finalmente no RGA, buscou-se determinar a melhor quantidade de indivíduos (um ou dois) a serem inseridos na próxima geração. Tanto no SSGA quanto no RGA a estratégia de substituição utilizada estabelece que os novos membros substituem os piores da população. Vale lembrar que foram realizadas 30 execuções em cada teste e o resultado (número de execuções com sucesso) encontra-se na Tabela 4-3 e na Tabela 4-4.

Tabela 4-3
Testando o elitismo para o SGA.

Função	Elitismo	Sem Elitismo
Degrau	23	15
Rastrigin	11	10
Picos	27	25

Tabela 4-4
Testando a reposição para o RGA.

Função	1 indivíduo	2 indivíduos
Degrau	16	11
Rastrigin	9	3
Picos	26	25

Com esses testes, escolheu-se trabalhar, de agora em diante, com o SGA que assegura a sobrevivência do melhor a cada geração e com o RGA que recoloca em cada geração apenas 1 indivíduo. Para encontrar o melhor intervalo G do SSGA o procedimento foi um pouco mais complicado. Ao invés de valores numéricos, um gráfico foi feito para tentar encontrar uma região na qual se pudesse esperar um bom desempenho médio relativo à execução das três funções teste. Variou-se o intervalo G de 0.05 até 0.95 com incrementos de 0.05 a cada teste. O resultado está mostrado na Figura 4-4.

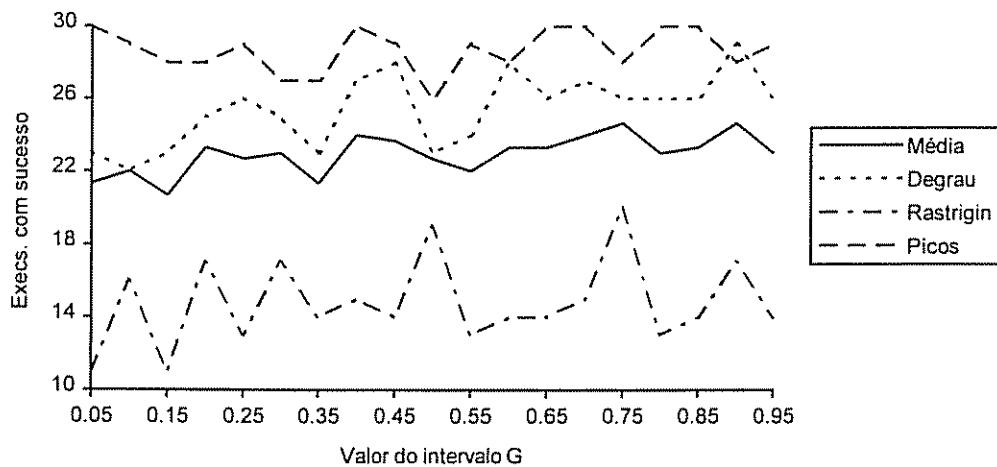


Figura 4-4: Testando o intervalo G de população para o SSGA.

A região foi escolhida pela melhor média de 5 execuções consecutivas. O melhor intervalo aconteceu para $G \in [0.7, 0.9]$. Então escolheu-se $G=0.8$ por que, além de

estar contido no intervalo, proporciona um número inteiro de indivíduos a cada geração ($50 \times 0.8 = 40$).

Quanto a estes testes, pode-se questionar: não poderia a estratégia elitista do SGA ser analisada, assegurando a sobrevivência para a próxima geração dos dois (ou três) melhores ao invés de apenas o melhor? Por que se testou para o RGA, apenas a inserção de um ou dois indivíduos a cada geração e não mais que isso? Por que o intervalo G não foi implementado para valores menores que 0.05 ou maiores que 0.95? A resposta para estas perguntas é simples. Se o elitismo garantir a sobrevivência de outros indivíduos, o SGA torna-se SSGA. O contrário acontece quando o intervalo G do SSGA é bem baixo. De forma análoga, o RGA se comportará como o SSGA (de G baixo) ou com SGA padrão quando o número de indivíduos novos a cada geração forem altos.

Observando a Figura 4-4, nota-se uma variância bem acentuada na função de Rastrigin. Isso indica que, mesmo usando 30 execuções, às vezes não se consegue definir uma predominância de um ou outro mecanismo. O motivo é que os GAs são dirigidos por técnicas estocásticas e, assim sendo, não se pode afirmar que ele convergirá para a solução global em 100% dos casos, mesmo que ele tenha tido sucesso em todas as execuções feitas. A Tabela 4-5 agrupa os resultados desta fase inicial.

Tabela 4-5
Resultados da fase inicial de testes.

Função	RGA	SGA	SSGA	Média
Degrau	23	16	26	21.7
Rastrigin	11	9	13	11.0
Picos	27	26	30	27.7
Média	20.3	17	23	-

Antes de iniciar outros testes, é importante apresentar como os GAs se comportam ao longo de uma execução. Para mostrar o comportamento, registraram-se os valores máximo e médio da função desempenho em cada geração. O resultado pode ser visto na Figura 4-5, Figura 4-6 e Figura 4-7. Dessa forma, fica mais fácil visualizar e compreender a convergência de cada algoritmo.

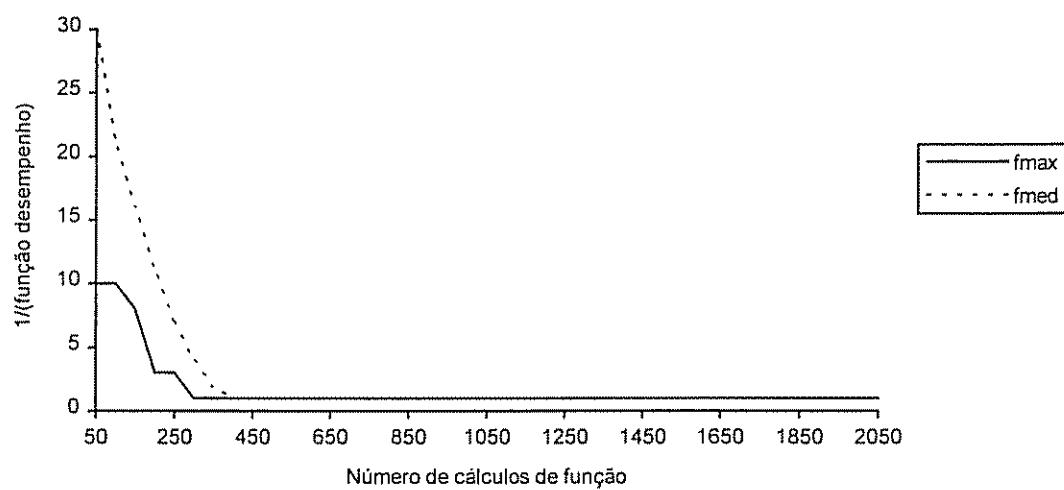


Figura 4-5: Evolução dos valores máximo e médio da função desempenho ao longo de uma dada execução do RGA (para a função de Rastrigin com n=2).

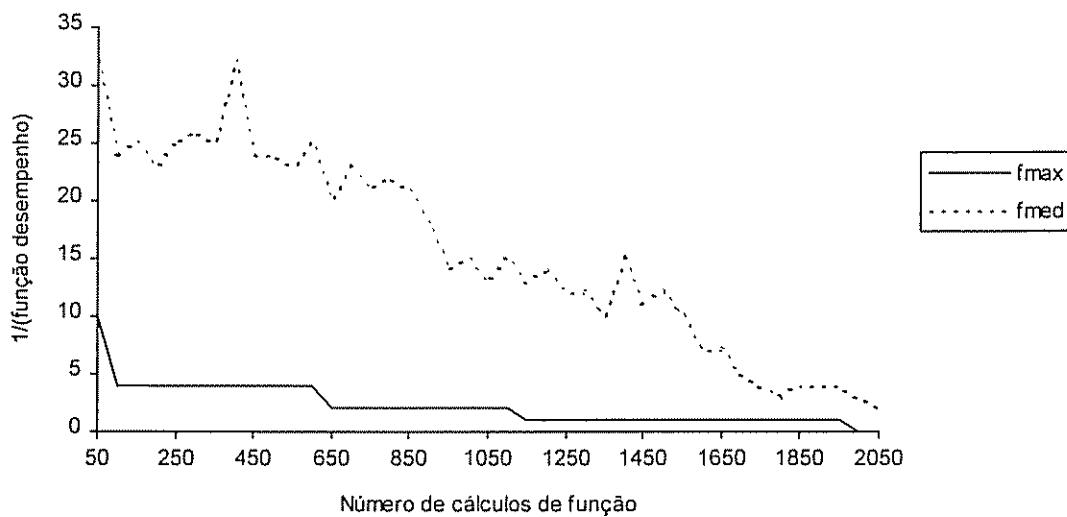


Figura 4-6: Evolução dos valores máximo e médio da função desempenho ao longo de uma dada execução do SGA (para a função de Rastrigin com n=2).

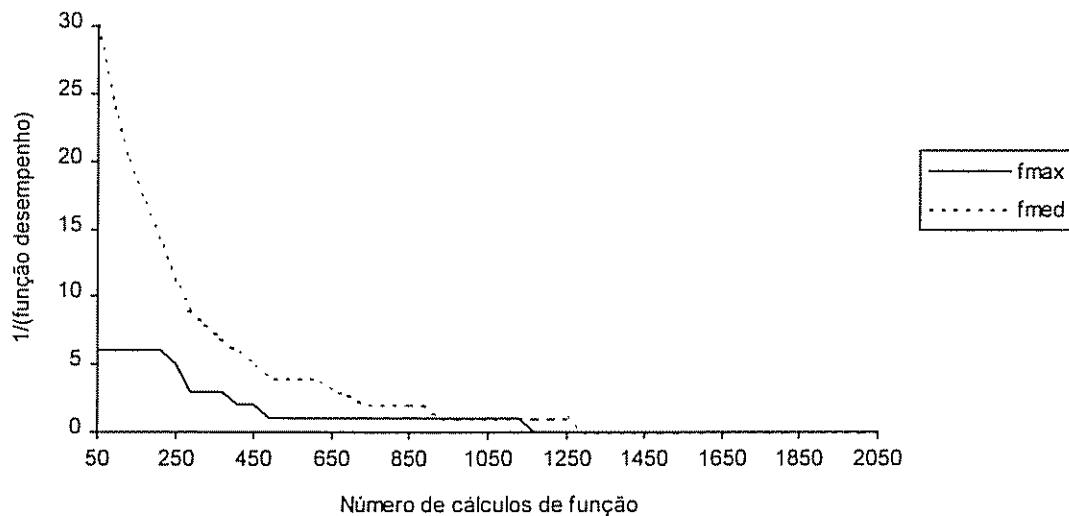


Figura 4-7: Evolução dos valores máximo e médio da função desempenho ao longo de uma dada execução do SSGA (para a função de Rastrigin com $n=2$).

Nas próximas seções de testes, foram analisadas outras opções para os mecanismos de seleção, escalonamento e cruzamento, que estavam fixos até então. Inicialmente, foram tratados outros métodos de seleção. Os resultados encontram-se nas tabelas a seguir. O termo MCF (Média de Cálculos de Função) denota a média de cálculo de função sobre as 30 execuções. Esse é um fator importante para a avaliação da manutenção de diversidade dentro da população num GA.

Tabela 4-6
Função Degrau e os métodos de seleção.

GAs	Roleta		Torneio		SUS		SRS		DS	
	ES	MCF	ES	MCF	ES	MCF	ES	MCF	ES	MCF
SGA	22	2050	27	1613.3	18	2050	15	2050	18	2050
SSGA	25	810	23	572.7	24	826	26	803.3	25	843.3
RGA	16	347.4	7	220.6	14	376.7	13	391.5	18	457.3
Média	21.0	1069.1	19.0	802.2	18.7	1084.2	18.0	1081.6	20.3	1116.9

Tabela 4-7
Função Rastrigin e os métodos de seleção.

GAs	Roleta		Torneio		SUS		SRS		DS	
	ES	MCF	ES	MCF	ES	MCF	ES	MCF	ES	MCF
SGA	11	2050	16	1851.7	9	2050	10	2050	10	2050
SSGA	17	1466	15	992.7	15	1590	21	1696.7	14	1538
RGA	9	440.2	6	266.7	9	535.3	10	486.3	11	520
Média	12.3	1318.7	12.3	1037.0	11.0	1391.8	13.7	1411.0	11.7	1369.3

Tabela 4-8
Função Picos e os métodos de seleção.

GAs	Roleta		Torneio		SUS		SRS		DS	
	ES	MCF	ES	MCF	ES	MCF	ES	MCF	ES	MCF
SGA	29	650	28	650	23	650	26	650	23	650
SSGA	27	611.3	25	495.3	28	650	29	648.7	29	650
RGA	26	499	24	274.4	23	563.3	27	529	23	481.8
Média	27.3	586.8	25.7	473.2	24.7	621.1	27.3	609.2	25.0	593.9

Nestas tabelas é interessante notar as colunas relativas à média de cálculo de função. Alguns algoritmos são finalizados pelo segundo critério de parada ($1-m_{dg}<0.01$), ou seja, antes que o número máximo de cálculo de função seja atingido.

Essa rápida uniformização pode facilitar os GAs a pararem numa solução local. As tabelas mostram que, considerando conjuntamente os três GAs, o método da Roleta encontra-se bem cotado para as três funções e que o método do Torneio funciona muito bem quando se trata do SGA, mas em contrapartida não é eficiente para o RGA. Isso porque o Torneio é o mais tendencioso dentre os métodos de seleção apresentados e o RGA entre os algoritmos. O resultado é a *convergência prematura*. No caso do SGA, que possui convergência mais suave, o método do Torneio contribui como agente compensador, sendo, portanto, benéfico à sua interação. Fazendo um balanço de desempenho dos GAs, o SSGA mostrou-se superior na maioria dos testes.

Pelos resultados da Tabela 4-6, que se referem à função Degrau, os métodos da Roleta e DS foram os que apresentaram melhor performance. Para escolher o método de seleção que seguirá nos próximos testes, as duas técnicas foram executadas mais 50 vezes, prevalecendo o método da Roleta. Para a função de Rastrigin, também houve um impasse entre os métodos Roleta, Torneio e SRS. Seguindo o procedimento aplicado à função Degrau, executou-se mais 50 vezes cada um destes. O SRS sobressaiu-se levemente, e portanto, ele foi o escolhido. O mesmo problema ocorreu para a função Picos e, mesmo executando 100 vezes, ainda não se pôde afirmar qual o melhor método. O problema de encontrar melhores técnicas, para a função Picos é mais acentuado porque os GAs não encontram tantas dificuldades quanto às outras funções para a sua resolução. Com isso, vários métodos proporcionam boa performance. O SRS foi o escolhido por ter ficado bem classificado em todas as execuções.

Após verificar a potencialidade dos métodos de seleção, foram analisados dois métodos de escalonamento e também com a ausência de qualquer tipo de escala. A performance dos dois métodos de escalonamento foi observada para vários valores dos multiplicadores Sigma e Linear. Em [11], Goldberg diz que as faixas mais usuais destes escalonamentos são [1.2-2.0] para o tipo Linear e, [1.0-3.0] para o tipo Sigma. Na tentativa de melhorar o desempenho dos GAs relativamente aos métodos de escalonamento, foram testados neste trabalho, valores fora destas faixas. Para o escalonamento Linear, pesquisou-se dentro do intervalo compreendido entre 1.0 e 4.0, com incrementos de 0.05, totalizando 61 testes para cada GA. Usando o mesmo incremento, a faixa para o escalonamento Sigma variou de 0.5 até 3.0, o que resultou em 51 testes. Para encontrar o melhor escalonamento para cada função teste, inicialmente calculou-se a média de execuções com sucesso para os três GAs

em cada valor dos multiplicadores. A seguir, utilizou-se um procedimento semelhante ao usado para descobrir o melhor intervalo G do SSGA. A diferença é que, ao invés de buscar a região com uma largura de faixa δ fixa, procurou-se também regiões para outras larguras de faixa. O propósito seria certificar se a região de melhor escalonamento está ou não bem definida. As figuras a seguir demonstram melhor a situação.

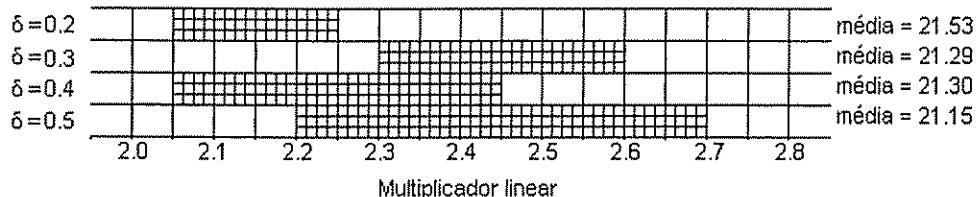


Figura 4-8: Faixas de melhor desempenho para a função Degrau com escalonamento Linear.

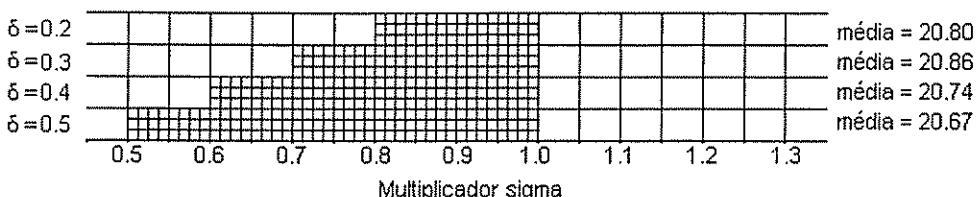


Figura 4-9: Faixas de melhor desempenho para a função Degrau com escalonamento Sigma.

No escalonamento Linear, nota-se que as faixas não definem uma única região como no escalonamento Sigma. Mas assim mesmo vale a pena usar o tipo Linear porque as médias das faixas são ligeiramente superiores às do tipo Sigma. O valor do multiplicador Linear foi escolhido como o valor médio da região de interseção das faixas, ou seja, 2.4.

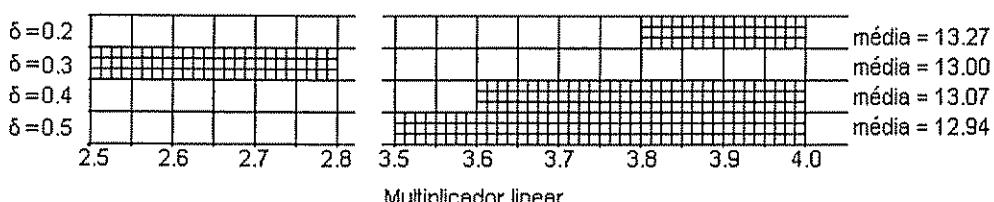


Figura 4-10: Faixas de melhor desempenho para a função Rastrigin com escalonamento Linear.

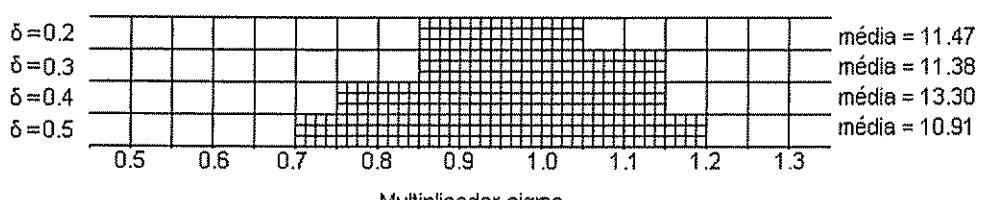


Figura 4-11: Faixas de melhor desempenho para a função Rastrigin com escalonamento Sigma.

Observando as figuras para a função de Rastrigin, depara-se com o caso anteriormente descrito, ou seja, mesmo que as faixas do escalonamento Sigma determinem uma região e as do escalonamento Linear não a definam, a superioridade da média do tipo Linear favorece a escolha do método. Então, o valor escolhido para o multiplicador Linear foi de 3.9.

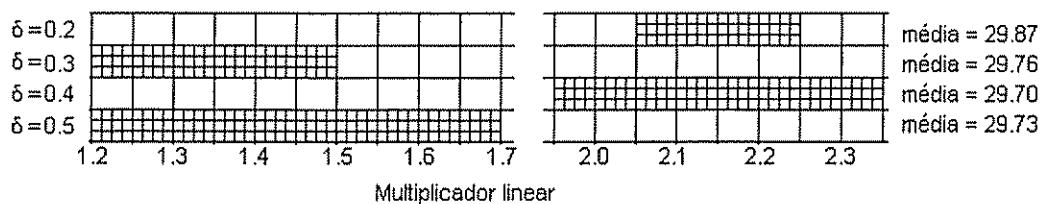


Figura 4-12: Faixas de melhor desempenho para a função Picos com escalonamento Linear.

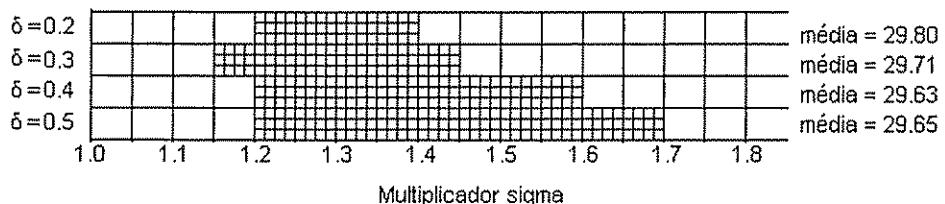


Figura 4-13: Faixas de melhor desempenho para a função Picos com escalonamento Sigma.

Para a função Picos, o escalonamento escolhido foi o Sigma, por ter uma região bem definida entre as faixas e também porque quase não há diferenças entre as médias. O valor para o multiplicador Sigma foi 1.3.

Para os valores de escalonamento escolhidos, aprimoramos os GAs como pode ser visto na Tabela 4-9. Na mesma tabela, também foram introduzidos os resultados dos GAs, quando executados sem escalonamento. Observa-se, claramente, as vantagens de se trabalhar com uma população escalonada.

Tabela 4-9
Resultados para o escalonamento escolhido.

Função	Escalonamento	RGA		SGA		SSGA		Média	
		escala	sem	escala	sem	escala	sem	escala	sem
Degrau	Linear(2.40)	15	9	20	28	29	19	21.33	18.67
Rastrigin	Linear(3.90)	9	6	15	8	14	12	12.67	8.67
Picos	Sigma(1.30)	29	22	30	26	30	25	29.67	24.33

Terminado os testes de escalonamento e adotando os valores de multiplicador escolhidos, o próximo passo foi verificar o melhor método de cruzamento. Obviamente, os operadores de recombinação PMX, OX, CX e ERO, não foram tratadas nesses testes porque as funções não são de combinação e permutação. Os resultados relativos aos métodos de cruzamento estão nas tabelas a seguir.

Tabela 4-10
Função Degrau e os tipos de Cruzamento.

GAs	1ponto		2Pontos		CPV		CEVI		Uniforme	
	ES	MCF	ES	MCF	ES	MCF	ES	MCF	ES	MCF
SGA	16	2050	19	2050	21	2050	10	2050	21	2050
SSGA	22	771.3	29	782	14	1890.3	12	700.7	27	964.7
RGA	12	385.7	19	389.9	13	1796	6	355.3	21	456
Média	16.7	1069.0	22.3	1074.0	16.0	1912.1	9.3	1035.3	23.0	1156.9

Tabela 4-11
Função Rastrigin e os tipos de Cruzamento.

GAs	1ponto		2Pontos		CPV		CEVI		Uniforme	
	ES	MCF	ES	MCF	ES	MCF	ES	MCF	ES	MCF
SGA	10	2050	11	2050	9	2050	5	2050	11	2050
SSGA	10	1351.3	20	1624.7	9	2036	11	1267.3	18	1975.3
RGA	8	617.2	13	651.7	10	765.2	5	450.6	11	776.6
Média	9.3	1339.5	14.7	1442.1	9.3	1617.1	7.0	1256.0	13.3	1600.6

Tabela 4-12
Função Picos e os tipos de Cruzamento.

GAs	1ponto		2Pontos		CPV		CEV		Uniforme	
	ES	MCF	ES	MCF	ES	MCF	ES	MCF	ES	MCF
SGA	29	650	30	650	30	650	24	650	30	650
SSGA	30	626	30	624.7	30	650	30	616.7	30	647.3
RGA	30	397.2	30	397.7	30	526.6	23	379.8	30	431.5
Média	29.7	557.7	30.0	557.5	30.0	608.9	25.7	548.8	30.0	576.3

Os resultados mostram que não há diferença acentuada entre os cruzamentos Uniforme e com 2 pontos de corte. Procedendo como nos testes de métodos de seleção, os dois cruzamentos foram executados mais 50 vezes. Para a função Degrau, o método Uniforme foi bem superior ao de 2 pontos. No entanto, para a função de Rastrigin, a técnica com 2 pontos foi ligeiramente superior, sendo, portanto, a escolhida. Finalmente, para a função Picos, somente o cruzamento CEVI não propiciou bons resultados. A Tabela 4-13 mostra a evolução dos GAs para as funções teste.

Tabela 4-13
Técnicas que aprimoraram o desempenho conjunto dos GAs em relação às três funções teste.

Função	Seleção	Escalonamento	Cruzamento	Resultado
Degrau	Roleta	Linear(2.4)	Uniforme	24.00
Rastrigin	SRS	Linear(3.9)	2Pontos	14.67
Picos	SRS	Sigma(1.3)	2Pontos	30.00

Todos os resultados mostraram que as técnicas que favorecem a *convergência prematura* produzem piores resultados. Nesse ponto, introduziremos a adaptação dinâmica dos operadores cruzamento e mutação (seção 3.11), que vem com a finalidade de manter diversidade dentro da população por mais tempo e, por conseguinte, aumentar as chances do algoritmo encontrar a solução global. Para este trabalho foram testados os critérios de adaptação DF [32], com ajuste percentual FF [45] e [46] e, PI [37]. Em [32] foi pesquisada também a ação conjunta dos três critérios de adaptação.

Os testes compreendem apenas procurar pela melhor faixa de ação dos métodos DF e FF. Quanto ao método da adaptação individual, usaram-se os valores numéricos sugeridos pelos autores em [37].

O procedimento de encontrar a faixa mais adequada não foi simples. Inicialmente avaliaram-se todas as larguras de faixa δ de 0.1 até 1.0 com incrementos de 0.05. Também para cada largura varreu-se toda região compreendida no intervalo [0-1]. O resultado mostrou muita eficiência dos métodos de adaptação, mas, infelizmente, não foi possível distinguir, com clareza, qual era a melhor largura e em que posição ela se encontrava.

Relativo ao critério FF, o que se pôde observar é que ele produz melhores resultados quando a faixa é mais estreita (com $\delta \leq 0.4$) e situada mais no início do intervalo. No entanto, para o critério DF, o qual atua em todo o intervalo possível de m_{dg} (o critério age dentro da faixa com as probabilidades dos operadores interpolados e fora da faixa com valores limites destas probabilidades, veja Figura 3-9), não se conseguiu definir um δ ideal, mas observou-se melhores resultados quando a faixa encontra-se situada mais no início do intervalo. Não se conseguiu definir um critério para a escolha da largura e da posição da faixa tanto da adaptação FF quanto da DF e, portanto, os valores dos termos foram escolhidos simplesmente pegando as faixas que propiciaram os resultados mais significativos nessas execuções. O resultado está fixado na Tabela 4-15.

Tabela 4-14
Valores limites de p_m e p_c e das constantes k .

Parâmetros	FF	DF	PI
p_m	0.001-0.05	0.001-0.05	0.0-0.5
p_c	0.50-1.0	0.50-1.0	0.0-1.0
k_m	1.15	-	-
k_c	1.20	-	-

Tabela 4-15
Faixas que propiciaram em melhores resultados para as funções teste.

Adaptação	Parâmetros	Degrau	Rastrigin	Picos
Dentro da Faixa	Largura δ	0.85	0.85	0.40
Fora da Faixa	Faixa m_{dg}	0.0 - 0.85	0.05 - 0.90	0.05 - 0.45
Dentro da Faixa	Largura δ	0.10	0.15	0.20
Fora da Faixa	Faixa m_{dg}	0.20 - 0.30	0.15 - 0.30	0.20 - 0.40

A Tabela 4-16 mostra a média de todas as execuções feitas na análise dos critérios de adaptação.

Tabela 4-16
Resultados dos critérios da ação dos critérios de adaptação.

Função	Adaptação DF		Adaptação FF		Adaptação PI	
	MES	MCF	MES	MCF	MES	MCF
Degrau	27.7	1075.1	28.0	1302.3	19.3	1536.7
Rastrigin	21.0	1702.4	22.7	2025.6	11.3	1566.7
Picos	30.0	598.2	30.0	585.3	29.3	500
Média	26.2	1125.2	26.9	1304.4	20.0	1201.1

Comparando os dados relativos às médias de execuções com sucesso fixados nas Tabela 4-6, Tabela 4-7 e Tabela 4-8 com os valores mostrados na Tabela 4-16, observou-se uma melhora significativa do desempenho médio dos GAs. Comparando também os resultados pertinentes à média de cálculo de função, nota-se que a média subiu, mostrando que o algoritmo sustentou a diversidade por mais tempo.

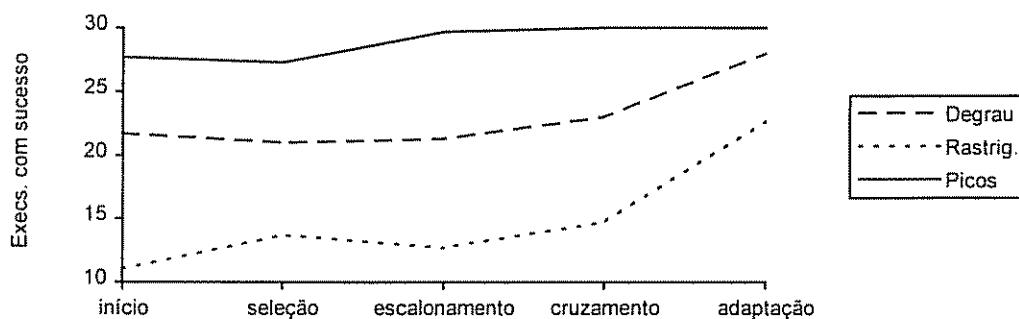


Figura 4-14: Balanço final dos testes de técnicas sobre as funções.

Observando na Figura 4-14 o desempenho médio dos GAs da fase inicial até os testes de cruzamento, não se obteve melhora tão significativa. Às vezes, de um teste a outro, houve até mesmo desempenho ligeiramente inferior à fase anterior. Isso acontece porque os GAs são guiados por regras de probabilidade, e flutuações existentes entre as execuções são completamente possíveis. É interessante notar que o escalonamento linear sugerido estava contido no intervalo [1.2-2.0] e, para as funções Degrau e Rastrigin, isto não se verificou. O método da Roleta esteve entre os melhores em toda execução, e os cruzamentos Uniforme e com dois pontos de corte também se efetivaram como boas opções de troca de material genético. Finalmente, os critérios de adaptação DF e FF, contribuíram decisivamente para melhoria da performance dos GAs.

Com esses procedimentos, encontraram-se os mecanismos mais robustos para cada função se analisados conjuntamente os três GAs. Depois desses testes, o próximo passo foi o desenvolvimento de cada GA separadamente, observando desta vez o desempenho médio relativo às três funções teste. O processo de desenvolvimento foi o mesmo anteriormente descrito, não tendo, portanto, motivos

de demonstrá-los detalhadamente como foi feito até agora. Os resultados provenientes da interação com os primeiros métodos estão fixados na Tabela 4-17 e os vindos da adaptação dinâmica foram inseridos na Tabela 4-18. Essa separação de resultados se justifica porque a adaptação dos operadores trouxe grande melhoria na performance dos GAs e isso poderia ocultar a ação dos métodos de seleção, escalonamento e cruzamento.

Tabela 4-17
Métodos mais adequados para cada GA.

GAs	Seleção	Escalonamento	Cruzamento	Resultado
RGA	Roleta	Linear(1,2)	Uniforme	21.0
SGA	Torneio	Linear(2,4)	2pontos	23.3
SSGA	SRS	Linear(3,1)	2pontos	25.6

Tabela 4-18
Ação da adaptação dinâmica sobre os GAs da Tabela 4-17.

GAs	Adaptação	Degrau	Rastrigin	Picos	MES
RGA	DF(0.00-0.20)	30	27	30	29
SGA	FF(0.05-0.25)	30	27	30	29
SSGA	FF(0.10-0.25)	30	30	30	30

Na Tabela 4-18, observa-se que as melhores faixas para a ação da adaptação dinâmica são estreitas e se situam no início do intervalo possível [0 - 1] de m_{dg} . Esperava-se que a faixa para o método DF fosse mais larga, adaptando p_c e p_m conforme o valor de m_{dg} . Na Tabela 4-18, nota-se justamente o contrário. Os valores p_c e p_m mantiveram-se constantes por longo espectro de faixa. O que se pode concluir quanto à adaptação DF é que os parâmetros da Tabela 4-14, provenientes de Soares & Vasconcelos em [32], não estão bem dimensionados, sendo, portanto, uma grande motivação de continuação da pesquisa neste tema. Já o critério de adaptação FF é bem mais flexível que o DF, pois ele se aplica fora da faixa quando não estiverem ultrapassados os valores limites de p_c e p_m . Entretanto, esses parâmetros também deverão ser investigados.

Terminada mais esta fase de testes, vale a pena fazer um balanço da ação dos métodos sobre a performance dos GAs, relativamente às três funções teste.

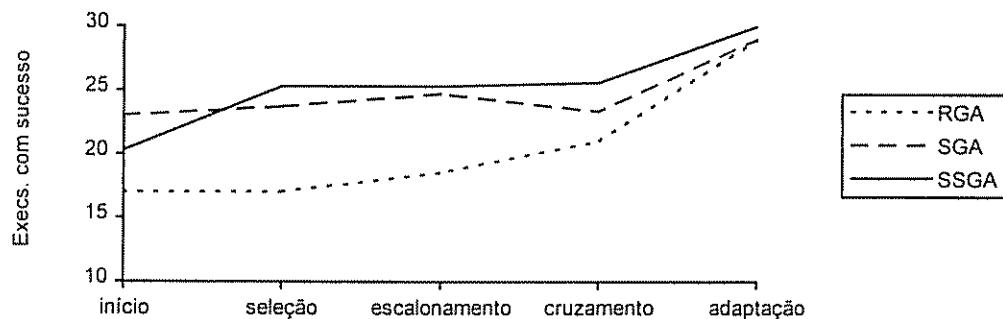


Figura 4-15: Balanço final dos testes sobre os GAs.

Não houve, entre os métodos de seleção, um que apresentasse melhores resultados quando empregado nos três Algoritmos Genéticos (RGA, SGA e SSGA). Observou-se que, para um dado GA, um critério de seleção funcionou melhor que os outros. Provavelmente, se as funções fossem mais complexas, métodos de seleção mais imparciais, tipo o SUS, poderiam ser os mais adequados. A ação do escalonamento não parece eficaz, visto que o desempenho foi praticamente o mesmo da fase anterior de teste. Porém, a ausência de escalonamento facilita a *convergência prematura*, conduzindo os GAs à diminuição de desempenho, como pode ser visto na Tabela 4-9. Os cruzamentos Uniforme e com 2 pontos de corte sobressaíram-se, mais uma vez, em relação ao demais. Quanto aos cruzamentos CPV e CEVI descritos aqui, ainda não se mostraram eficientes. O tipo CPV conduz os GAs a uma convergência mais lenta, enquanto que o CEVI conduz a uma convergência bem rápida. Analisar mais atentamente as características destes cruzamentos será um grande motivo de continuidade de pesquisa nessa direção. Antes de se comentar as técnicas de adaptação, observa-se que os GAs com formação padrão (Roleta, Linear(1.2) e o cruzamento 2pontos) conduzem a um bom desempenho, pois a melhoria dos GAs não foi tão significativa quando buscou-se técnicas alternativas de seleção, escalonamento e cruzamento. Os gráficos da Figura 4-14 e Figura 4-15 mostram uma importante melhoria trazida pelos métodos de adaptação dinâmica das probabilidades de cruzamento e mutação, que foram uma contribuição deste trabalho na pesquisa sobre Algoritmos Genéticos. Extrapolando a idéia de adaptação dinâmica para atingir também os métodos de seleção, escalonamento, cruzamento, tamanho da população, intervalo G, entre outros parâmetros, obtém-se um GA totalmente adaptativo. Este é, sem dúvida, um interessante tema de pesquisa sobre GAs.

4.2.3 Outras Técnicas

Esta seção apresenta técnicas que aprimoram ainda mais a performance dos GAs. O objetivo não é comparar uma técnica com outra, pois cada técnica tem o seu melhor ponto de funcionamento em dada situação. Para a realização de cada um dos testes, apenas um dos três GAs foi escolhido e com alguns parâmetros definidos. São eles:

- $p_c: 0.6$; $p_m: 0.001$;
- seleção: Roleta;
- escalonamento: Linear (1.2);
- cruzamento: 2pontos;
- critério de adaptação dinâmica: DF ($V_{\min}=0.10$; $V_{\max}=0.80$) (valores dos parâmetros DF são obtidos da Tabela 4-14).

O motivo de se fixar esses parâmetros é a grande possibilidade de combinações de valores de parâmetros entre estas técnicas e os GAs. Para analisar essas técnicas, foi tomada como teste somente a função de Rastrigin (sob o formato da equação 4-2 com $\epsilon=10^{-5}$), porém com o número de variáveis aumentado (2, 4, 6, ..., 20 variáveis). A análise dos resultados foi feita através de gráficos que relacionam a convergência final dos GAs (número de execuções com sucesso) ou do número de cálculos de funções com o número de variáveis da função teste. O critério de verificação da convergência para a solução global foi definido como *norma euclidiana* inferior a 0.3. No caso de duas variáveis, esse critério implica em uma região bem definida na qual se encontra a solução global. Entretanto, aumentando-se o número de variáveis, a adoção desse mesmo valor de norma conduz a um pequeno erro na definição da região global, se comparado ao caso de duas variáveis. Assim, quanto maior o número de variáveis mais precisa deve ser a solução, de modo a satisfazer o critério *norma euclidiana* < 0.3. Para mais de duas variáveis, um critério mais justo seria o critério do *valor máximo*, ou seja, verificar se o maior valor do vetor solução fosse menor que uma dada constante. Infelizmente, só depois de realizados todos os testes é que se observou essa outra possibilidade. O critério de parada utilizado foi o de convergência por número de gerações. Para cada número de variáveis foram feitas 30 execuções.

Inicialmente, foi estudado um GA acoplado a um Método Determinístico (seção 3.15). O objetivo foi melhorar a convergência final para o ótimo global. O Método

Determinístico utilizado foi o BFGS ([1] e [41]), que, incorporado ao código do GA escolhido, opera de n em n gerações. A função de Rastrigin tem os mínimos locais muito próximos, fazendo com que o BFGS gastasse apenas de 5 a 6 iterações para convergir, com precisão de 10^{-3} para cada variável, não onerando tanto o tempo computacional final.

O GA empregado foi o SSGA ($G=0.10$) e, além dos parâmetros citados no início desta seção, incluíram-se o tamanho da população e o número de gerações, os quais foram variados de forma a dar condições ao GA a convergir para a solução global, sem exagerar excessivamente o número de cálculo de funções. Os valores numéricos destes parâmetros, relativamente ao número de variáveis analisado, podem ser obtidos na Figura 4-17 (como os parâmetros são de natureza diferente, não tem sentido dar um título ao eixo das ordenadas). A Figura 4-16 mostra os resultados das execuções que obtiveram com sucesso.

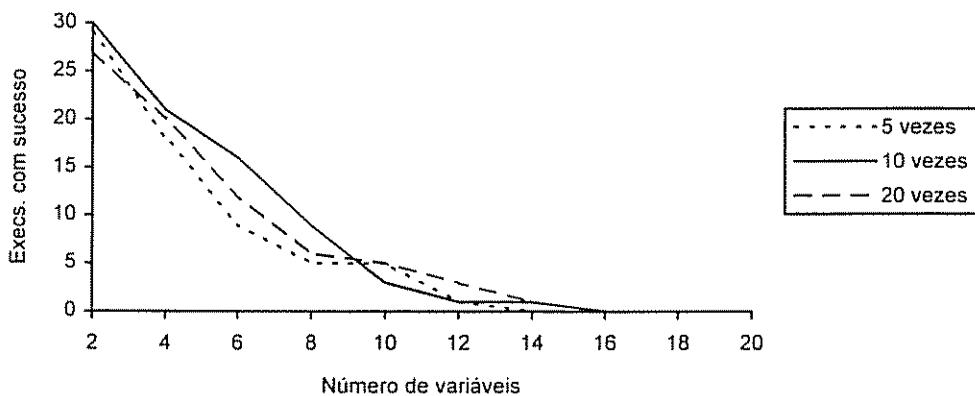


Figura 4-16: Analisando o desempenho do SSGA associado ao BFGS.

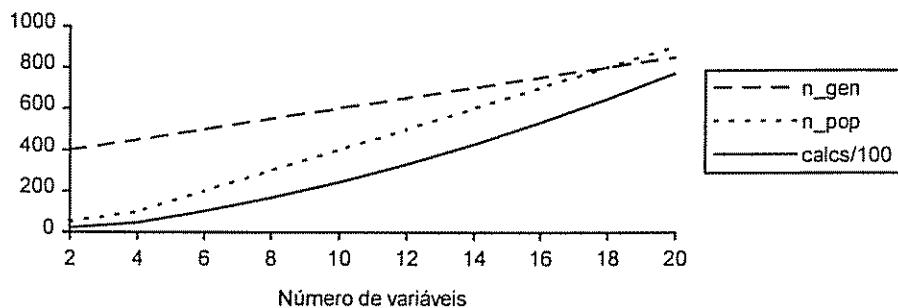


Figura 4-17: Parâmetros variáveis nas execuções da técnica híbrida: Tamanho da população, número de gerações e, por conseguinte, número de cálculo de função médio.

Realmente, esta técnica híbrida consegue refinar a resposta. Porém, quanto à melhoria do número de execuções com sucesso, os resultados (veja Figura 4-16)

não possibilitam nenhuma conclusão. Esse GA híbrido não teve problemas em satisfazer o critério de convergência para a solução global (*norma euclidiana* < 0.3) porque com o BFGS, obtém-se a solução local com a precisão desejada (neste caso, cada x_i do vetor solução foi calculado de forma que o erro seja inferior a 0.001).

O próximo item que foi analisado foi a formação de nichos (seção 3.16) com o método descrito por Goldberg & Richardson [10] e Goldberg [11]. Nessa técnica, o tamanho n_{pop} da população influí muito no tempo computacional do algoritmo, pois, para medir o grau de proximidade de cada indivíduo com os demais, são feitos aproximadamente $n_{pop}^2/2$ (usando simetria) cálculos da *função de partilha* em cada geração. O grupo de métodos e parâmetros usado foi o mesmo apresentado no início desta seção. No entanto, o valor do $\epsilon=10^{-6}$ na equação 4-2 foi aumentado para $\epsilon=2$, de forma que não favorecesse tanto à região da solução global (isso fica melhor explicado observando a Figura 3-3). Caso o ϵ fosse mantido com seu valor anterior, praticamente quaisquer quantidades de indivíduos poderiam pertencer ao pico global, mesmo que a *função de partilha* penalize esses indivíduos próximos. O GA empregado agora é o SGA. O motivo é que se quer mostrar que a técnica de Goldberg mantém diversidade através da formação de nichos e, caso fosse utilizado SSGA, esta manutenção de diversidade não corresponderia fielmente, pois o intervalo G mantém alguns indivíduos de uma geração a outra. Os resultados obtidos encontram-se na Figura 4-18.

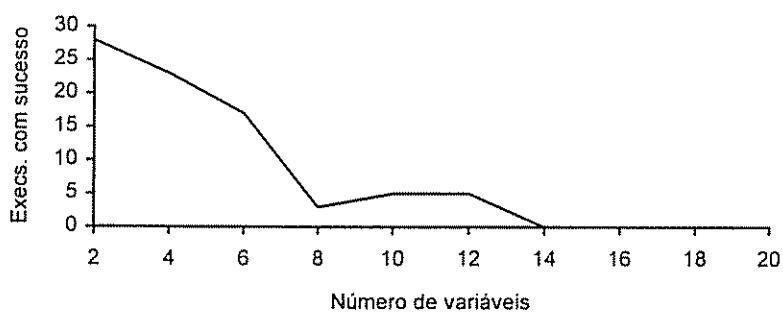


Figura 4-18: Desempenho do SGA com uso da formação de subpopulações.

Antes de fazer a análise da Figura 4-18, achou-se conveniente apresentar os valores quantitativos dos parâmetros população e geração usados (Figura 4-19). Outro dado importante, que está mostrado na Figura 4-20, é a evolução do número de integrantes dos quatro nichos principais para a função de Rastrigin com quatro variáveis.

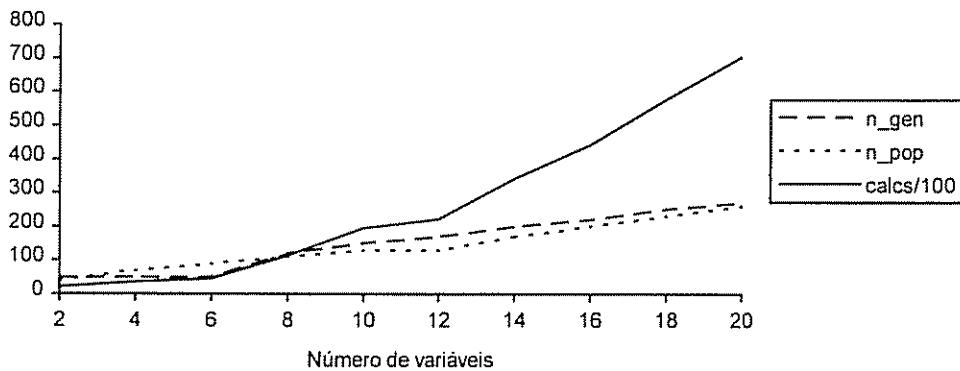


Figura 4-19: Parâmetros variáveis nas execuções da formação de nichos: Tamanho da população, número de gerações e, por conseguinte, número médio de cálculos de funções.

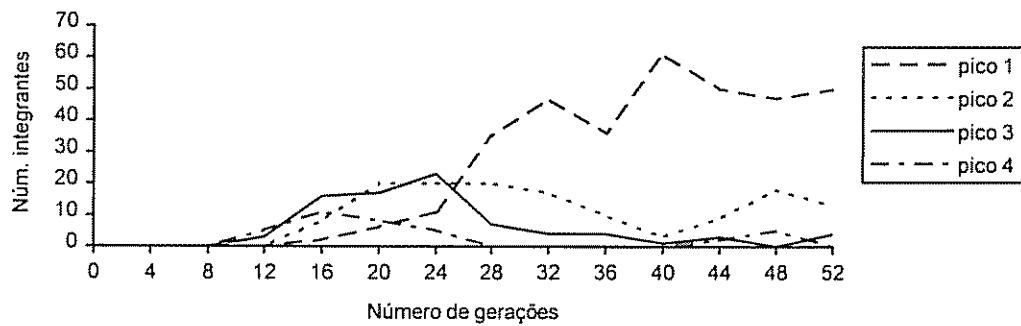


Figura 4-20: Número de integrantes das 4 subpopulações principais no decorrer das gerações.

Observando a Figura 4-18, nota-se que o desempenho em termos de execuções com sucesso cai bruscamente quando o SGA tenta resolver a função de Rastrigin com número de variáveis superior a 6. Uma possível explicação para esse comportamento seria que o número de integrantes da população tenha sido insuficiente. Infelizmente com essa técnica de nicho não se deve trabalhar com grandes populações, porque o custo computacional cresce muito (veja Apêndice B). Em momento algum Goldberg ([10] e [11]) comentou sobre a performance desta técnica de nicho em relação à convergência para o ponto ótimo global. O que foi transmitido em seus trabalhos é que a técnica buscaria criar nichos através da penalização de indivíduos que estivessem próximos, assim sendo, ao terminar uma execução, ter-se-iam como resultado várias soluções. Para mostrar a distribuição dessas soluções ao longo das gerações, observe a Figura 4-20. Ela mostra o resultado para Rastrigin com 4 variáveis, mas todos os casos (de 2 a 20 variáveis) foram acompanhados e concluiu-se que o SGA com a técnica de nicho pode ser muito útil quando se quer avaliar várias soluções.

A última técnica a ser verificada numericamente é a redução do intervalo de procura. Como visto na seção 3.17, há algumas questões a serem respondidas quanto ao procedimento: Quando aplicar a redução? Qual a taxa de redução j do intervalo? Quantas vezes reduzir? Essa última questão não será respondida, pois ainda está em estudo. Com isso, para todos os testes aqui realizados sobre redução de intervalo, o número de reduções foi 1. Nessa fase de testes, utilizou-se o SSGA ($G=0.8$) juntamente com os parâmetros citados no começo desta seção e a taxa de redução $j=-0.85$. O número de gerações e o tamanho da população estão apresentados na Figura 4-21. Obviamente, uma mudança de parâmetros pode conduzir a resultados totalmente diferentes. Por exemplo, se o número de gerações não for suficiente, o SSGA pode finalizar a busca sem utilizar o mecanismo de redução.

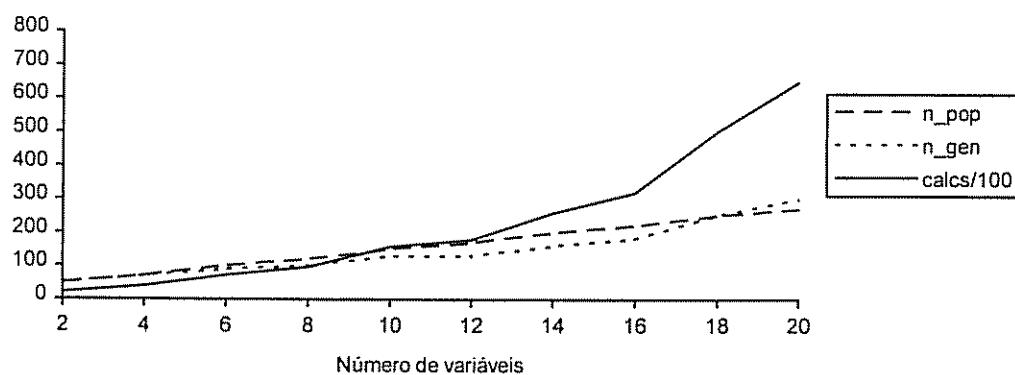


Figura 4-21: Parâmetros variáveis nas execuções de redução de intervalo: Tamanho da população, número de gerações e, por conseguinte, número médio de cálculos de funções.

Para que haja redução de intervalo, é conveniente definir um m_{dg} mínimo, a partir do qual, se fará a redução. Foram observados todos os valores de m_{dg} , tal que, $m_{dg} \in [0.05, 0.95]$, com passo 0.05. Os resultados que melhor representam o comportamento do SSGA para diversos m_{dg} estão impressos na Figura 4-22.

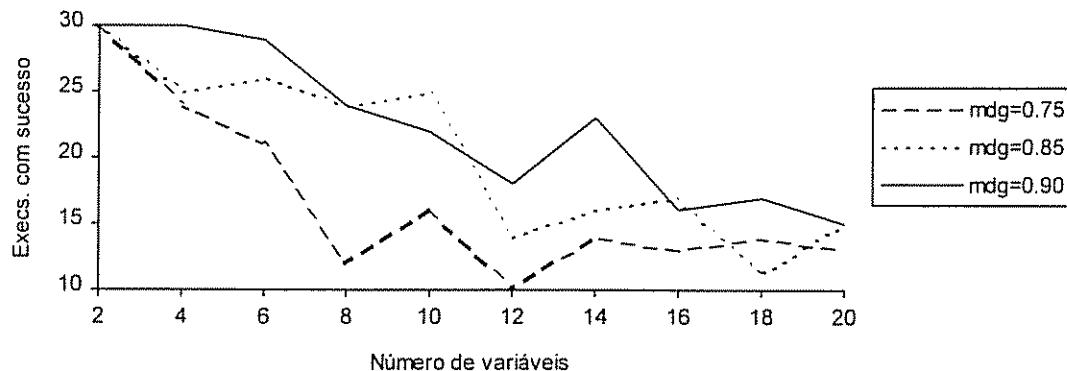


Figura 4-22: Analisando o desempenho do SSGA aplicando redução de intervalo, quando varia-se o valor de m_{dg} .

A Figura 4-22 mostra o quanto é importante saber o momento de executar a redução de intervalo. Se for aplicada a redução com m_{dg} baixo, provavelmente o GA não conseguirá definir a região onde se encontra o mínimo. Por outro lado, se o valor de m_{dg} estiver bem próximo da unidade, a região da solução estará mais bem definida, porém se gastará muito mais cálculos de função para atingir o desejado m_{dg} , penalizando-se, assim, a performance final (por causa do número total de gerações). Deve-se observar ainda os parâmetros dos critérios de adaptação. Por exemplo, dependendo do valor de $p_{m\max}$, a população pode nunca convergir.

Realizando o mesmo procedimento utilizado para encontrar os melhores valores de m_{dg} , a taxa de redução também foi analisada dentro do intervalo [0.05, 0.95], com passo 0.05. A Figura 4-23 mostra os valores da taxa de redução que melhor representam o comportamento do GA. Nesses testes, utilizou-se o melhor m_{dg} da fase anterior ($m_{dg}=0.90$).

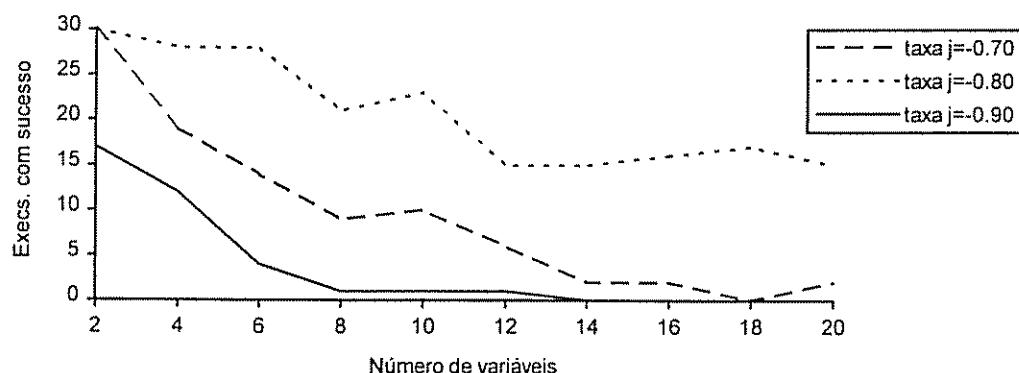


Figura 4-23: Analisando o desempenho do SSGA aplicando redução de intervalo, quando varia-se o valor das taxas de redução.

Como se pode notar na Figura 4-23, conhecer a melhor a taxa de redução é um fator crucial para que se consiga melhores resultados na convergência final desse GA. Se a taxa de redução for muito pequena, o domínio de procura é praticamente o mesmo anterior, não aproveitando, portanto, a região definida pelo melhor indivíduo. Foi experimentado que, prosseguindo-se a redução continuamente, o desempenho torna-se superior ao melhor visto na Figura 4-23. O grande problema é o elevado número de cálculos de função exagerado. Por outro lado, se a taxa de redução for muito grande, o novo domínio de procura é insuficiente para proporcionar a saída do GA da redondeza do indivíduo tomado como base.

Um outro aspecto importante é a precisão do vetor solução com redução de intervalo. Quando reduz-se o intervalo, mantém-se o número de bits utilizados na resolução do intervalo inicial, e com isto, aumenta-se a precisão de cada variável. Essa precisão pode ser observada na Figura 4-23, onde foram obtidos 15 sucessos para Rastrigin com 20 variáveis. Vale lembrar que o critério utilizado foi *norma euclidiana <0.3*.

Como dito no início da seção, essas técnicas não foram apresentadas com o objetivo de comparar uma com outra, pois cada técnica é utilizada para diferentes finalidades. O BFGS para melhorar a precisão da solução, a técnica de nicho para encontrar mais de uma solução em apenas uma execução e o mecanismo de redução de intervalo para melhorar o desempenho quanto ao número de execuções com sucesso e também melhorar a precisão da solução. Cada uma das técnicas tem suas particularidades e, o seu melhor ponto de funcionamento em dada situação. Mas, em todo caso, como alguns parâmetros utilizados para todas as técnicas foram comuns, resolveu-se mostrar algum resultado comparativo. A Figura 4-24 apresenta o número de cálculos de função utilizado e a Figura 4-25 mostra número de execuções com sucesso.

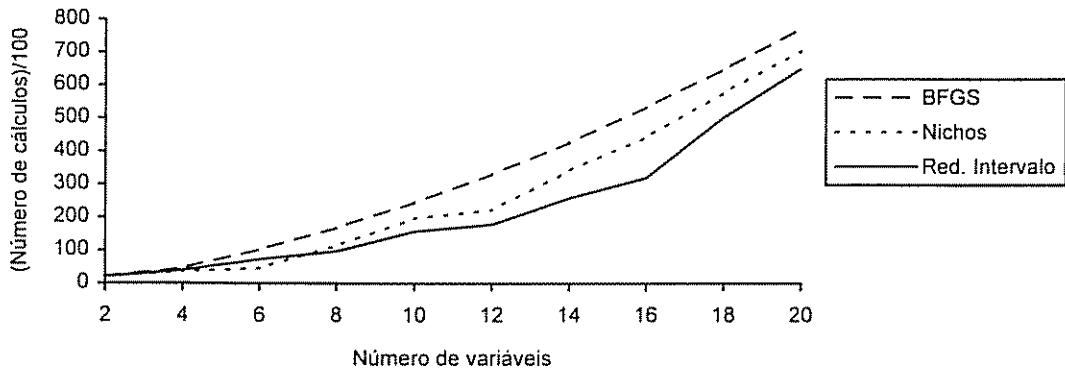


Figura 4-24: Comparando o número de cálculos de funções utilizados na associação dos GAs com a técnica BFGS, formação de nichos e redução de intervalo.

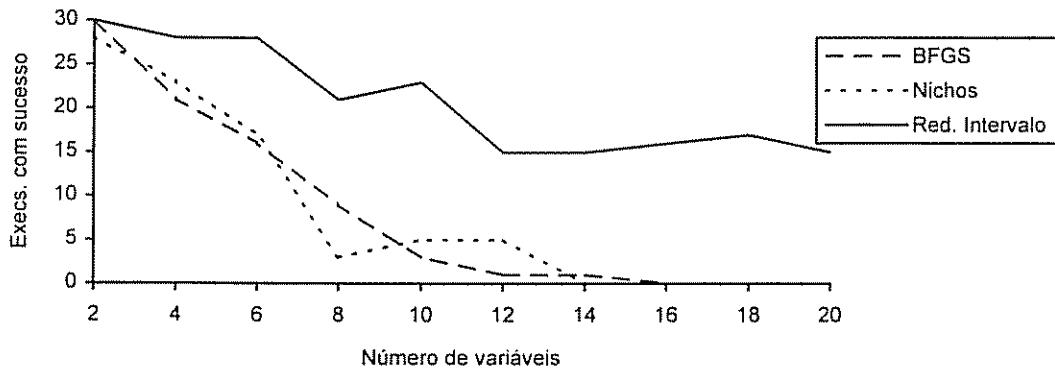


Figura 4-25: Comparando o número de execuções com sucesso obtido na associação dos GAs com técnica BFGS, formação de nichos e redução de intervalo.

Observando a Figura 4-24, nota-se que o mecanismo de redução de intervalo utilizou um número de cálculos de função inferior ao utilizado nas outras técnicas e ainda obteve um de execuções com sucesso mais expressivo número (veja Figura 4-25). A técnica híbrida teve comportamento semelhante à técnica de nichos, se observado o número de execuções com sucesso relativamente ao número de variáveis tratado. Vale lembrar que cada técnica tem uma finalidade específica e a associação delas teoricamente traz bons resultados. Uma associação imediata, sem causar prejuízos, seria lançar o BFGS no final da execução de cada uma das outras técnicas.

4.3 Resultados de Alguns Pesquisadores

Alguns métodos explicados no Capítulo 3 e não verificados numericamente neste capítulo foram objeto de trabalho de muitos pesquisadores. Esta seção é formada

por alguns resultados desses trabalhos, tendo a finalidade de preencher lacunas deixadas pela ausência de testes.

4.3.1 Resultados de Spears quanto à Formação de Nicho

O método de formação de nichos de Spears [36], descrito na seção 3.16, consiste na criação de subpopulações, simplesmente anexando a cada indivíduo da população uma etiqueta binária.

Em seu trabalho, Spears analisou numericamente o comportamento de seus esquemas de formação de subpopulações (SSS1 e SSS2) com o de Goldberg e, teoricamente, com esquemas que formam subpopulações agrupando indivíduos em processadores trabalhando em paralelo.

Para verificar o desempenho dos mecanismos, Spears usou 4 funções, as quais estão reimpressas na Figura 4-26.

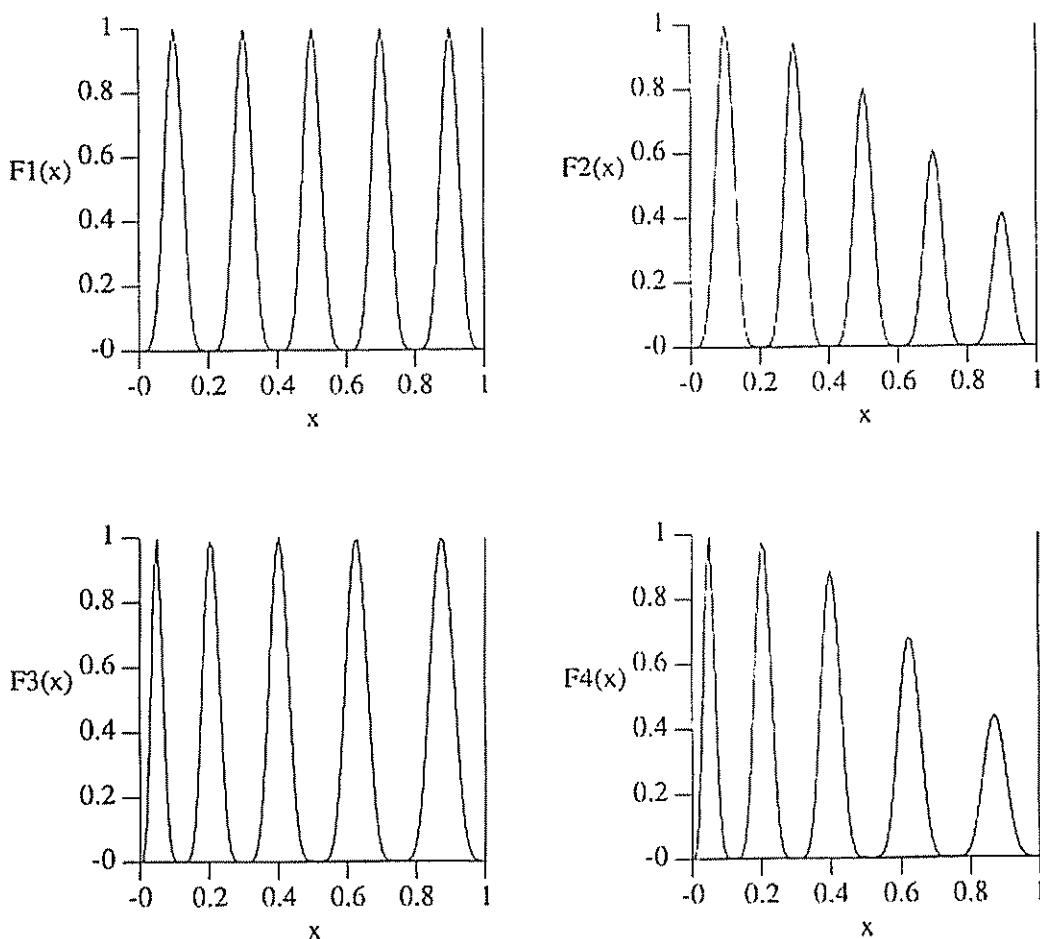


Figura 4-26: Funções teste de Spears.

Observando com cuidado, percebe-se que as 4 funções se diferenciam uma das outras pela altura dos picos e pela posição no eixo x de seus cumes.

O mecanismo SSS1 aplicado às funções F1 e F3 se mostrou muito eficiente, conseguindo manter subpopulações estáveis em todos os picos e, consequentemente, obtendo resultados similares ao esquema de Goldberg, com um custo operacional muito mais barato. No entanto, quando SSS1 foi usada nas funções F2 e F4, ele teve problemas em sustentar subpopulações nos picos mais baixos. Para essas funções, o esquema de Goldberg funcionou perfeitamente. O motivo da instabilidade nos picos inferiores é que cada subpopulação é desenvolvida separadamente e a competição interna leva os indivíduos a migrarem de pico. Uma possibilidade de correção seria que as subpopulações tivessem poucos membros, mas esta não seria uma boa solução porque um pequeno grupo conduz a uma procura pobre e ainda corre o risco de desaparecer devido a erros estocásticos. Amenizar o problema de SSS1 foi a motivação para a criação de SSS2. Finalizando, SSS1 é um mecanismo adequado para problemas cujo objetivo é a distribuição das subpopulações sobre as soluções mais importantes.

O esquema SSS2 aplicado às funções F1 e F3 se comportou como SSS1, entretanto, quanto às demais funções, os resultados foram bem melhores. SSS2 conseguiu manter subpopulações estáveis mesmo tratando-se de funções com picos de magnitude diferentes, mas assim mesmo o método de Goldberg é superior na distribuição da população total sobre os picos, sendo, infelizmente, um método muito caro computacionalmente. Se o método de Spears funcionasse bem, seria esperado que o tamanho das subpopulações fosse proporcional à importância do pico como em Goldberg. Para mostrar o comportamento do mecanismo de Spears perante a função F4, o SSS2 foi executado com a população total de 200 indivíduos e 16 subpopulações. A Figura 4-27 acompanha o desenvolvimento de 4 subpopulações durante 250 gerações.

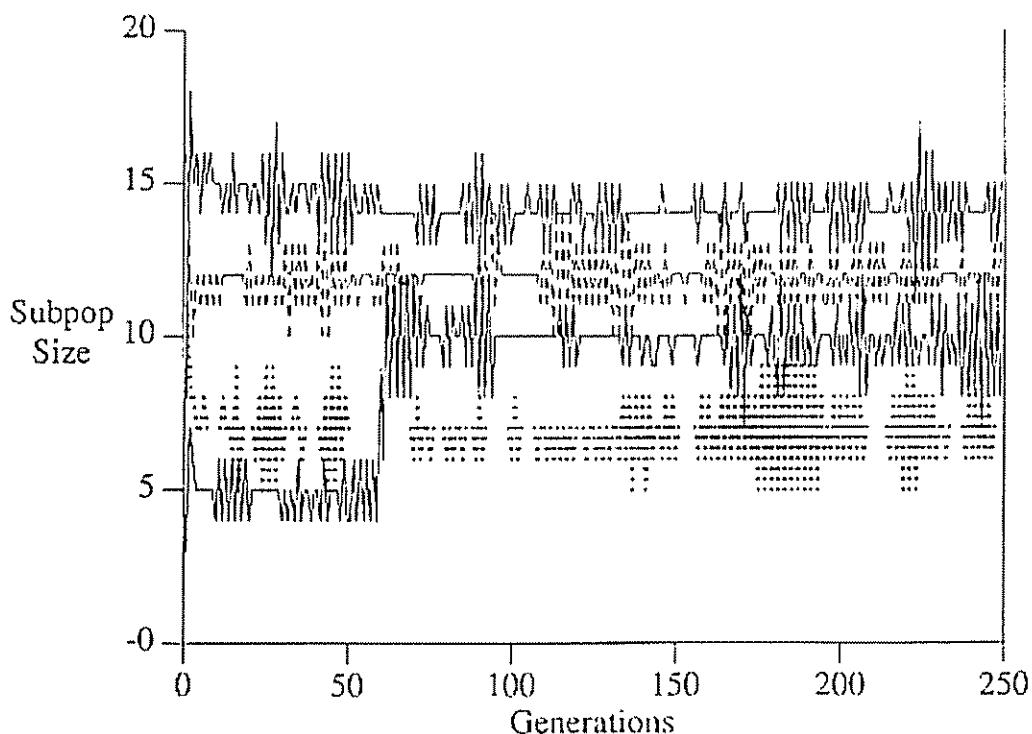


Figura 4-27: Evolução de quatro subpopulações ao longo das gerações.

Interessante observar na Figura 4-27 que se a função fosse F1 esperar-se-ia que o tamanho das subpopulações ficasse em torno de 12.5 ($=200/16$). Como se trata de F4, o ideal seria que a quantidade de membros de cada subpopulação fosse proporcional à importância da região ocupada por eles, ou seja, valores menores e maiores que 12.5. Outro ponto importante, visto no gráfico, é a mudança de pico da subpopulação que tem menor quantidade de indivíduos no início da execução. Isso pode ser notado observando o crescimento do número de integrantes da sua subpopulação. A última observação sobre a Figura 4-27 é a respeito da flutuação do tamanho de cada subpopulação, que surge devido à sensibilidade do mecanismo de partilhamento.

Em relação ao método de Goldberg, que associa precisão numérica (entre importância da região e tamanho da população) a um custo elevado, os procedimentos de Spears também se mostraram eficientes na formação de subpopulações estáveis e não trazem sobrecarga computacional significativa. Comparando o esquema seqüencial de Spears com GAs que usam arquitetura baseada em processadores paralelos (uma subpopulação para cada processador), há algumas diferenças. Nos algoritmos de Spears, a aptidão individual é dinâmica, baseada na quantidade de membros da subpopulação ao qual o indivíduo está contido. Como consequência, pode-se concentrar esforço nas regiões mais promissoras e ainda manter indivíduos sobre outras áreas do espaço. Num tipo de

construção de *GA Paralelo*, cada processador cuida de uma subpopulação, a qual tem tamanho constante. Dependendo da implementação, permite-se a migração de indivíduos entre subpopulações. Nessa implementação de *GA Paralelo*, a aptidão individual é calculada da forma tradicional (retorno da função desempenho).

4.3.2 Resultados de Oliver & Outros quanto aos Cruzamentos de Permutação

No trabalho [28], Oliver & Outros compararam desempenho dos cruzamentos PMX, CX e OX modificado, na resolução de um TSP para 30 cidades. O cruzamento OX analisado aqui se diferencia do método original. Aqui os autores forçam o primeiro ponto de corte sempre na primeira posição do indivíduo.

Foram executados muitos testes com vários grupos de parâmetros. As populações analisadas continham 50, 200 e 500 membros. A probabilidade de cruzamento foi fixa em 0.8. Para cada cruzamento executado, uma mutação adaptada (*swap*), que, quando acionada, troca duas cidades de posição, foi empregada com probabilidade p_{swap} variando de 0.3 a 1.0 em intervalos de 0.1. O número de gerações variou de forma que o GA utilizado parasse em 50000 cálculos de função. O método de seleção foi a Roleta, usada com algumas modificações (não foi citada quais modificações) para reduzir erros estocásticos. Nenhuma estratégia elitista foi usada. O problema é fictício, portanto não há unidade de medida para as distâncias entre as cidades. As coordenadas das cidades estão fixadas em [28] e o melhor resultado para todo o percurso é 424 (conseguido por outro autor). Na Tabela 4-19, encontram-se os melhores resultados dos testes.

Tabela 4-19
Comparação entre OX, PMX e CX.

Cruzamento	População	p_{swap}	Resultado
OX	500	0.30	449
PMX	50	0.60	498
CX	50	0.50	517

Para este problema OX foi 11% superior ao PMX e 15% ao CX. Pela análise teórica (em termos de probabilidade de sobrevivência de esquemas) realizada em [28], espera-se que OX vença o tipo PMX nos problemas em que posições adjacentes são importantes no indivíduo, como é o caso do TSP. Por outro lado, os autores esperam um melhor desempenho do cruzamento PMX.

Achando o resultado não tão satisfatório, Oliver & Outros tentaram buscar a melhor solução (424) e testou algumas variações de forma que o número total de cálculos de função não ultrapassassem 200000. Consegiu a menor distância igual

a 425, com populações de 500 e 200 membros e p_{swap} igual a 15% e 25% respectivamente.

Um trabalho realizado por Whitley & Outros (Capítulo 22 de [4]) também analisou este mesmo caso de TSP. Nesse trabalho, os autores introduziram o operador de recombinação ERO e obtiveram excelentes resultados. Enquanto que Oliver & Outros não encontraram a melhor solução nem com 200000 cálculos, Whitley & Outros encontraram a solução em 28 vezes das 30 execuções realizadas, utilizando 70000 cálculos. Quando usaram a formação de subpopulações a convergência para a solução global foi 30/30 com 80000 cálculos.

4.3.3 Análise de De Jong & Spears sobre a Interação entre Tipo de Cruzamento e Tamanho da População

De Jong & Spears [5] tratam da interação entre tamanho da população e cruzamento. A idéia surgiu de um trabalho anterior [34], no qual os pesquisadores observaram o cruzamento apenas sob o ponto de vista de ruptura de esquemas, ou seja, aquele cruzamento que fornecesse menor probabilidade de ruptura ao esquema seria melhor (por causa da hipótese dos *building blocks*- vide equação 2-5). Em [34], eles analisaram, teoricamente e numericamente, os cruzamentos com n pontos de corte (vários valores de n) e o cruzamento Uniforme. Pelas equações desenvolvidas, os cruzamentos menos destruidores eram o com 2 pontos de corte e com 1 ponto, nessa ordem e, por outro lado, o que provocava mais ruptura era o Uniforme. Na prática, nem sempre se verificavam estas equações e o cruzamento Uniforme às vezes produzia resultados melhores que todos. Então em [34] não foi possível estabelecer o tipo de cruzamento que realmente poderia ser melhor.

De Jong & Spears citaram outros trabalhos em que o cruzamento com 16 pontos de corte e o Uniforme derrotavam os demais. Com esse resultado, eles puderam supor que as aproximações feitas nas equações de ruptura poderiam conduzir a algum erro significativo, ou então, somente a minimização da ruptura não é o melhor caminho para selecionar os operadores de cruzamento apropriadamente.

Se a segunda hipótese for correta, então deve existir situações em que a ruptura ajuda no processo de procura. Os autores analisaram vários trabalhos nos quais diferentes tipos de cruzamento produzem melhor performance. Eles chegaram à conclusão de que existe pelo menos duas importantes situações em que a ruptura traz vantagens. A primeira ocorre quando a população está bastante uniformizada e

a segunda quando a população é muito pequena para um problema complexo. As duas situações serão tratadas a seguir.

No princípio do processo de evolução, a população possui muita diversidade. Nesse ponto, o cruzamento deve favorecer a formação de novas soluções a partir da sobreposição das melhores existentes (*building blocks*). Com o passar das gerações, a população vai ficando mais e mais homogênea, até chegar ao ponto de todos os membros possuírem praticamente o mesmo código genético. Quando isso acontece, os cruzamentos com menores probabilidades de ruptura têm dificuldade de gerar indivíduos diferentes dos pais. É fácil perceber isto verificando o exemplo de cruzamento a seguir.

1	-	1	1	1	1	1	1	1	1	1	1	1
2	-	1	1	1	1	1	1	1	1	0	0	0

Se o último ponto de corte do cruzamento ocorrer antes do *locus* 9, os indivíduos formados serão apenas clones. Em casos como esse, a ruptura é benéfica à formação de novos indivíduos.

A outra situação favorecida pela alta taxa de ruptura do cruzamento aparece quando os problemas são mais complexos e a população é pequena. Problemas complexos exigem boa amostragem para a convergência global, e uma população pequena pode não oferecer a quantidade necessária. Nesse caso, alta taxa de ruptura é um fator importante, pois retarda a homogeneização da população, explora mais os *locus* e, com isso, suaviza o problema da pequena amostragem.

Depois dessas observações, partiu-se para a parte de análise numérica. Para simular a complexidade dos problemas, foram escolhidas 6 funções teste com o número de picos variando de 1 a 6. Todas as funções possuem apenas um pico (solução) global e $n-1$ (n número de picos total) soluções locais. As funções estão apresentadas em [5] e [34] e não serão reproduzidas neste trabalho. Importante, porém, é salientar que o retorno das funções foi mapeado para o intervalo compreendido entre 0 e 1. Somente solução global retorna 1 e todas as soluções locais retornam o mesmo valor menor que 1. Foram realizados 20 experimentos para cada função, combinando 4 tamanhos de população (20, 50, 100, e 1000) e 5 tipos de cruzamento (Uniforme, 2, 4, 8 e 16 pontos). Cada experimento foi executado 10 vezes. Não sendo possível a apresentação de todos os resultados, De Jong & Spears apresentaram os que acharam mais significativos.

A Tabela 4-20, reproduzida de [5], compara a performance dos cruzamentos Uniforme e com 2 pontos de corte, indicando aquele que se saiu melhor nas dez execuções. Os dois tipos de cruzamento representam os dois extremos de ruptura.

Tabela 4-20
Comparação dos cruzamentos 2pontos e Uniforme.

Problem	2-point vs. Uniform			
	Population Size			
	20	50	100	1000
6-peak	Uniform	Uniform	2-point	2-point
5-peak	2-point	2-point	2-point	2-point
4-peak	?	2-point	?	2-point
3-peak	Uniform	?	2-point	2-point
2-peak	Uniform	2-point	?	2-point
1-peak	Uniform	Uniform	?	?

O símbolo "?" indica que houve dúvidas quanto ao melhor método. Da Tabela 4-20, nota-se a predominância do cruzamento 2pontos quando se move para direita e para cima. Por outro lado, o cruzamento Uniforme é melhor no canto inferior esquerdo. Resultados similares são obtidos comparando 2-pontos com 16-pontos.

Alguns resultados gráficos relativos aos dois tipos de cruzamento da Tabela 4-20 estão fixados na Figura 4-28. Nota-se que, embora as grandes populações cheguem a melhores soluções, elas também precisam de muito mais gerações para a convergência.

Estes resultados sugerem um caminho para entender melhor o papel do cruzamento n-pontos. Para pequenas populações, cruzamentos que favorecem a ruptura tais como o Uniforme e o n-pontos ($n > 2$) são mais prováveis que produzem melhores resultados porque eles ajudam a aproveitar melhor a diversidade de pequenas populações e, com isso, suavizam a uniformização. Entretanto, as grandes populações já estão providas de quantidade suficiente de amostragem, e cruzamentos menos destrutivos aos esquemas, provavelmente, possam trabalhar melhor.

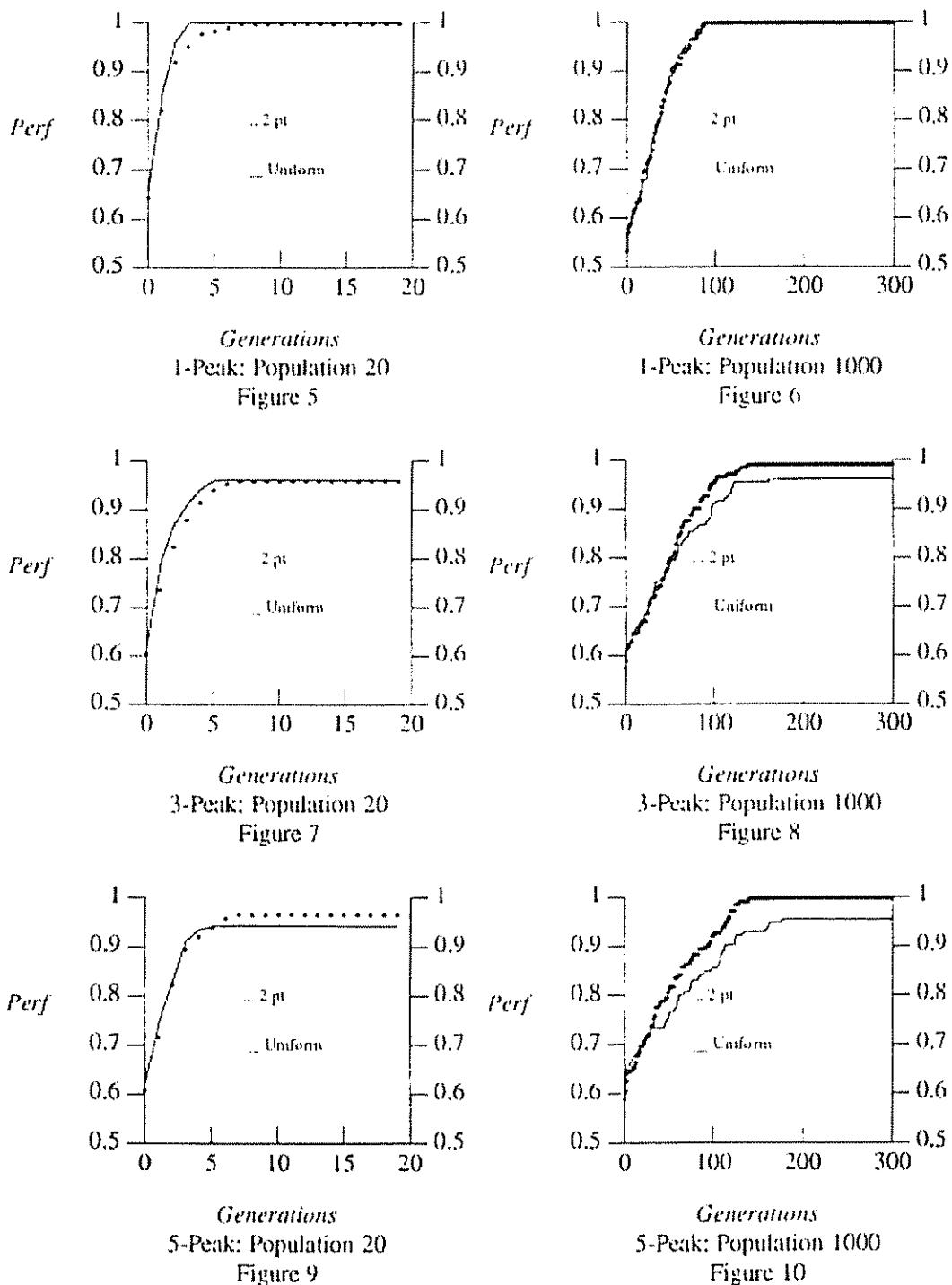


Figura 4-28: Interação entre cruzamento e tamanho da população.

4.4 Conclusão

Um dos objetivos do Capítulo 4 foi encontrar quais técnicas, dentre as conhecidas na bibliografia, seriam as mais robustas, se analisadas conjuntamente com os três GAs apresentados e as três funções teste propostas. Inicialmente, foram analisadas as técnicas do ponto de vista das funções e depois do ponto de vista dos GAs.

Mesmo que tenha havido melhorias da fase inicial de testes até a fase dos cruzamentos, estas não foram tão significativas, visto que os GAs são algoritmos estocásticos e, portanto, pode haver flutuações nas execuções. O que pode ser concluído é que os cruzamentos Uniforme e 2pontos se mostram como as melhores opções de cruzamento, o método da Roleta esteve sempre nas primeiras posições nos testes de métodos de seleção. Concluiu-se, também, que os melhores multiplicadores de escalonamento não necessariamente devam estar dentro de faixas predeterminadas (para o caso do escalonamento Linear, a faixa sugerida em [11] por Goldberg é [1.2-2.0]). Nessa primeira fase de testes, o fator que realmente contribuiu para o melhor desempenho dos GAs foi a aplicação das técnicas de adaptação dinâmica dos operadores cruzamento e mutação. Com a adaptação tenta-se controlar a diversidade genética dentro da população, sendo este um fator importantíssimo para condicionar os GAs a encontraram a solução principal.

Em relação à segunda parte de testes, pôde-se notar que, ao se adicionar a técnica híbrida (GA + BFGS), ganha-se precisão na solução final, mas não necessariamente há melhoria de desempenho. Situação semelhante verifica-se na formação de nichos. Consegue-se a manutenção da diversidade por meio de uma função de partilha e, com isso, obtém-se uma distribuição da população de acordo com a importância da região. Mesmo trabalhando com diversidade (população distribuída), o GA com formação de nichos, não obteve resultados positivos quando a função de Rastrigin foi avaliada com número de variáveis superior a 14. Finalizando, foi analisada a técnica de redução de intervalo, que é uma contribuição desta dissertação. Esse foi o mecanismo que propiciou melhores resultados de todos os testes até então. A idéia da redução de intervalo se baseia no fato de que os GAs têm facilidades de encontrar a região onde se encontra o ponto ótimo, mas, em contrapartida, tem dificuldades de convergência final para a solução. Então, a partir do momento em que a população possui um dado m_{dg} , supõe-se que o melhor indivíduo até então define uma região onde se pode encontrar a solução global. Reduzindo-se o intervalo em torno do melhor indivíduo, aumentam-se as chances de aprimorar a convergência final.

Finalizou-se o capítulo analisando trabalhos de outros pesquisadores. Esse item vem com a finalidade de cobrir algumas das muitas lacunas deixadas nesta dissertação sobre a pesquisa de GAs. Certamente, é impossível fazer um apanhado de tudo o que já foi publicado sobre o tema.

5. Aplicações

5.1 Introdução

Depois de apresentar um GA com suas características básicas (Capítulo 1), demonstrou-se como ele age e porque é um poderoso método de otimização (Capítulo 2). A seguir foram descritos (Capítulo 3) e analisados numericamente (Capítulo 4), via funções teste, técnicas que podem auxiliar os GAs na busca do ponto ótimo. E agora, neste capítulo, com o intuito de validar os resultados obtidos até então, propõe-se a resolução de alguns problemas reais. Foram escolhidos três problemas de correntes induzidas simples.

5.2 Aplicações a Problemas de Correntes Induzidas

As correntes induzidas são correntes que surgem a partir da interação de um condutor com um campo magnético que varia no tempo. As correntes induzidas produzem perdas ôhmicas no condutor, provocam um campo magnético de reação e geram força, a qual é proveniente da interação dos campos induzido e de indução. Portanto, fica fácil perceber a importância do estudo das correntes induzidas, pois elas podem aparecer em todas as máquinas e aparelhos eletromagnéticos que estão sujeitos a campos variáveis no tempo. O fenômeno das correntes induzidas, em algumas aplicações é desejável, como no caso dos motores de indução, veículos com levitação eletromagnética e ensaios não destrutivos de materiais. Em outros casos, é indesejável como as correntes induzidas em tanques de transformadores, blindagens de cabos, sistemas de comunicação, entre outros.

A formulação matemática de cada problema tratado, que foi desenvolvida em [22] e [40], é extensa, o que provocou um alto tempo computacional gasto no cálculo de cada função de otimização (veja Apêndice B). Por esse motivo foram feitas apenas 20 execuções em cada caso. O GA utilizado é o SSGA com $G=0.8$, sendo que o método de seleção é o da Roleta, o escalonamento é o Linear e o cruzamento 2pontos. Juntamente com estes parâmetros utilizou-se o mecanismo de adaptação

dinâmica DF (valores de constantes provenientes da Tabela 4-14). A técnica de redução de intervalo ($m_{dg}=0.90$ e taxa=0.80) foi utilizada somente à Aplicação 2. O tamanho da população e o número de gerações variou de aplicação para aplicação.

5.2.1 Correntes Induzidas numa Superfície Plana Devido a uma Corrente numa Espira Paralela à Superfície

O primeiro caso trata do acoplamento magnético entre um anel com corrente $I=I_0e^{j\omega t}$ e uma placa plana condutora com características μ e σ . O plano do anel encontra-se paralelo ao da superfície. A geometria do problema é mostrada na Figura 5-1.

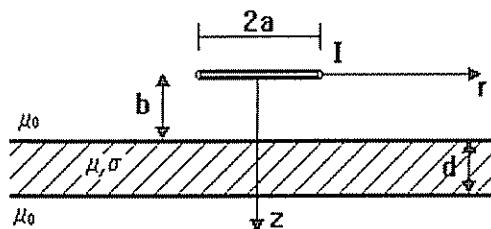


Figura 5-1: Anel com corrente circular paralelo à placa condutora.

O campo magnético, proveniente da corrente do anel, induz o aparecimento de uma densidade de corrente de indução J sobre a placa. A expressão analítica de J , dada em [22], é descrita pela equação 5-1.

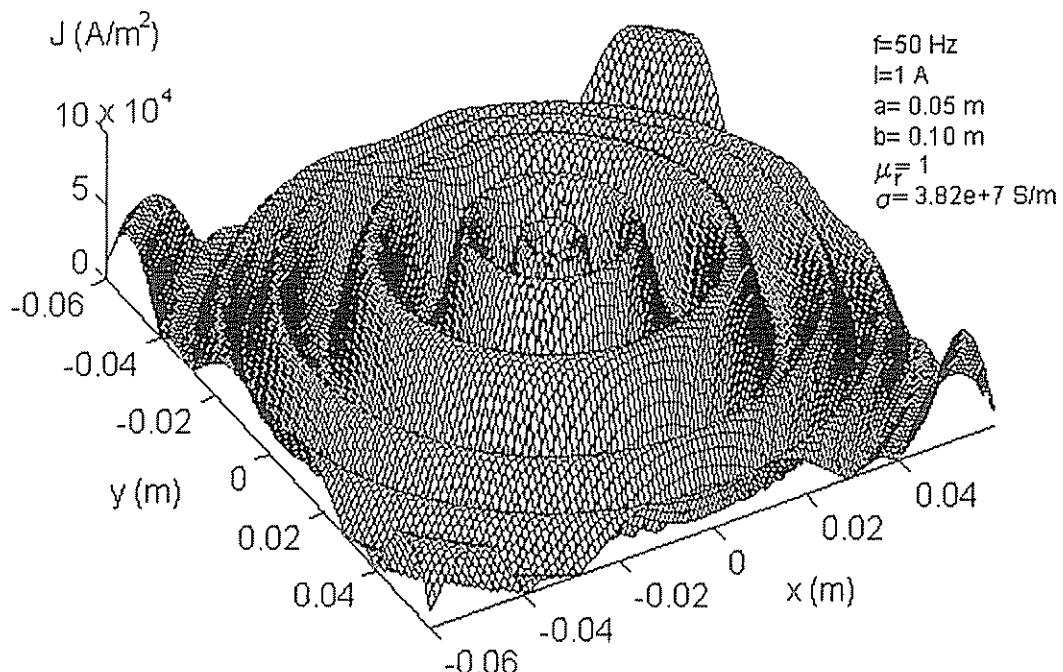
$$J_\phi(r, z) = -j \frac{Ip^2}{a^2} \int_0^\infty e^{-kb} J_1(k) J_1\left(\frac{kr}{a}\right) \left[\frac{(\mu_r k + q)e^{\frac{q(d-z)}{a}} - (\mu_r k - q)e^{-\frac{q(d-z)}{a}}}{(\mu_r k + q)e^{\frac{qd}{a}} - (\mu_r k - q)e^{-\frac{qd}{a}}} \right] k dk \quad (5-1)$$

onde $q=(k^2+j^*p^2)^{1/2}$, $p^2=\mu\sigma\omega a^2$, $z^*=z-b$ e, J_1 é a função de Bessel do primeiro tipo analisada no ponto de integração k .

A partir da equação 5-1, que calcula a densidade de corrente J em qualquer ponto da placa, pode-se obter as perdas totais como:

$$\text{perdas} = \frac{1}{\sigma} \int_r J(r)^2 dr \quad \text{no volume da placa.}$$

Na equação 5-1, fixando $z=b$ obtém-se a distribuição de J na superfície da placa (veja Figura 5-2) e, consequentemente, as perdas por unidade de comprimento em qualquer área desejada. Utilizando a configuração de parâmetros apresentada na Figura 5-2, e ainda, considerando a superfície de perdas como um quadrado de lado 0.14 m e concêntrico ao anel, as perdas obtidas na área foram 0.654436 W/m.

Figura 5-2: Distribuição de J na superfície da placa ($z=0.1\text{m}$).

Quanto maior o J , maiores serão as perdas. Dessa forma, quando se deseja como resultado o aquecimento da superfície, a intenção é procurar o maior J . Por outro lado, pode-se querer minimizar perdas como, por exemplo, nas blindagens de cabos. O problema de otimização, definido aqui, consiste em reduzir as perdas na superfície da placa em 40% em relação à configuração inicial. O resultado otimizado foi obtido através da modificação do material (novos μ_r , σ) e dos parâmetros geométricos a e b . A função objetivo $ff(x)$ foi escolhida como:

$$\min ff(x) = |\text{perdas} - 0.6 * \text{perdas}_{\text{iniciais}}| \quad (5-2)$$

Partindo dos valores da configuração inicial, limitou-se a região de busca para cada variável. Os limites estão fixados na Tabela 5-1.

Tabela 5-1 Limites das variáveis de otimização da Aplicação 2.		
Limite inferior	Variável	Limite superior
0.010 m	a	0.050 m
0.050 m	b	0.300 m
1	μ_r	100
$0.5 \times 10^7 \text{ S/m}$	σ	$7.0 \times 10^7 \text{ S/m}$

Nesta aplicação, o SSGA trabalhou com população com 20 membros durante 60 gerações, sendo gastos, portanto, 980 cálculos de função. Das 20 execuções, 18 foram bem sucedidas. Os conjuntos de parâmetros a , b , σ e μ_r que satisfizeram a função objetivo encontram-se fixados na Tabela 5-2.

Tabela 5-2
Soluções factíveis para Aplicação 1.

	a (m)	b (m)	σ (S/m)	μ_r
1	0.010142827	0.276256600	3.75981930*1E7	5.7253637
2	0.010462661	0.244631790	2.39364290*1E7	5.2389293
3	0.010623798	0.245623650	3.79691460*1E7	4.2660604
4	0.010764184	0.216516610	5.62231830*1E7	2.1722770
5	0.011097445	0.254702910	1.62991730*1E7	7.1907101
6	0.011462448	0.086431465	0.85151219*1E7	6.7254248
7	0.016406446	0.230539880	1.09709470*1E7	5.1996522
8	0.016830043	0.088209171	3.6035035*1E7	4.6286201
9	0.018119144	0.248736530	0.77950376*1E7	7.4807582
10	0.020414138	0.261684020	1.75171660*1E7	5.2026734
11	0.021550646	0.089841919	0.86301768*1E7	6.5985289
12	0.025636463	0.266467780	0.66762292*1E7	7.4777369
13	0.027990050	0.295750290	1.82551040*1E7	4.2358470
14	0.031558275	0.250727860	2.47477650*1E7	3.0273139
15	0.037626576	0.261104160	2.22165580*1E7	3.1088901
16	0.041232643	0.244914090	3.91494490*1E7	2.0937223
17	0.044650717	0.250323510	6.31403540*1E7	1.7281421
18	0.045917845	0.296253850	0.59918517*1E7	6.7918944

5.2.2 Correntes Induzidas numa Casca Cilíndrica Devido a uma Corrente num Filamento Paralelo Interno ao Cilindro

A segunda aplicação consiste no tratamento do acoplamento entre uma casca cilíndrica condutora e um filamento paralelo conduzindo uma corrente de excitação $I=I_0 e^{j\omega t}$. Este filamento encontra-se posicionado internamente à casca condutora, como pode ser visto na Figura 5-3.

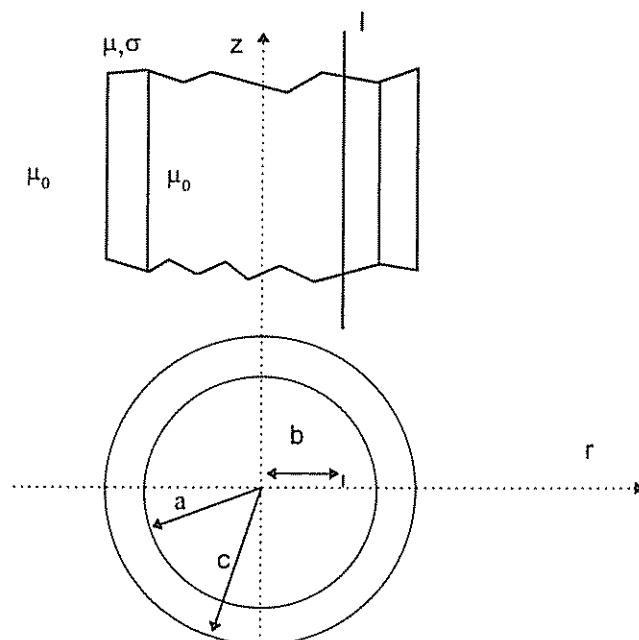


Figura 5-3: Cortes Longitudinal e transversal da casca cilíndrica infinita e o filamento de corrente interno à casca.

Como a casca cilíndrica e filamento têm comprimentos infinitos, o problema se reduz ao caso 2D. A distribuição de J no corte transversal é mostrada na Figura 5-4

e a seguir, na equação 5-3, é apresentada a formulação para esse problema de correntes induzidas.

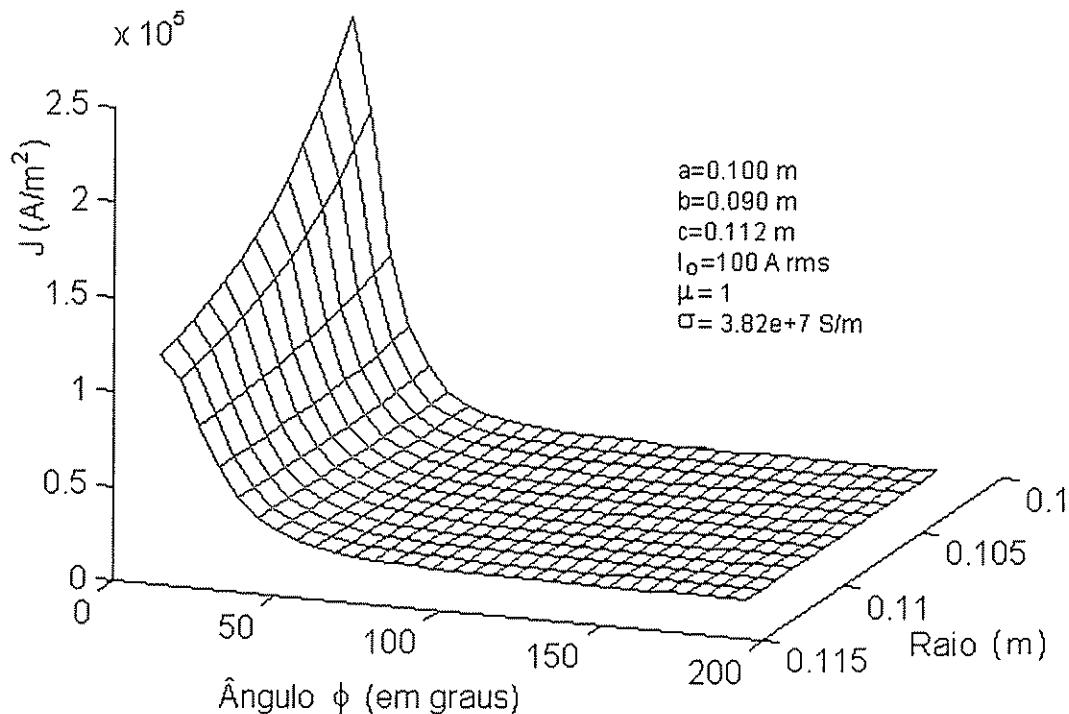


Figura 5-4: Distribuição de J ao longo da casca cilíndrica.

$$J_z(r, \phi) = \sum_0^n [F_n I_n(j^{1/2}pr) + C_n K_n(j^{1/2}pr)] \cos(n\phi) \quad (5-3)$$

Na equação 5-3, $p^2=\mu\sigma\omega$ e, I_n e K_n são as funções de Bessel Modificada de ordem n . Os termos F_n e C_n estão descritos nas equações que vêm a seguir.

$$F_n = \frac{j^{1/2}pl}{2\pi D_n} \left(\frac{b}{a}\right)^n \left(\frac{1}{a} + q_n\right) \left[\frac{\mu_r n K_n(j^{1/2}pc)}{j^{1/2}pc} - K_{n-1}(j^{1/2}pc) - n \frac{K_n(j^{1/2}pc)}{j^{1/2}pc} \right] \quad (5-4)$$

$$C_n = -\frac{j^{1/2}pl}{2\pi D_n} \left(\frac{b}{a}\right)^n \left(\frac{1}{a} + q_n\right) \left[\frac{\mu_r n I_n(j^{1/2}pc)}{j^{1/2}pc} + I_{n-1}(j^{1/2}pc) - n \frac{I_n(j^{1/2}pc)}{j^{1/2}pc} \right] \quad (5-5)$$

Nas equações 5-4 e 5-5, $q_n=0$ quando $n=0$ e, para $n>0$, q_n é igual a $1/a$. E finalmente, o termo D_n está descrito pela equação 5-6.

$$D_n = I_{n+1}(j^{1/2}pa) * K_{n-1}(j^{1/2}pc) - I_{n-1}(j^{1/2}pc) * K_{n+1}(j^{1/2}pa) \quad (5-6)$$

As perdas na seção transversal por unidade de comprimento são dadas pela equação (5-7).

$$\text{perdas} = \frac{1}{\sigma} \int_0^{2\pi} \int_a^c J^2 r dr d\phi \quad (5-7)$$

As perdas iniciais calculadas a partir da configuração descrita na Figura 5-4, resultaram em 0.055914482 W/m (calculando a equação 5-3 para n=20 termos). O problema de otimização definido para essa aplicação é encontrar configurações na qual as perdas se reduzam a 0.039140137 W/m (correspondente a 70% do valor inicial de perdas). Para alcançar este objetivo, foram procurados novos parâmetros geométricos (a e c) e um novo material (μ_r , σ). O posicionamento do condutor foi mantido proporcional à posição do raio interno ($b=0.9*a$). A partir dessas considerações, definiu-se a função objetivo $ff(x)$ como:

$$\min ff(x) = |\text{perdas} - 0.7 * \text{perdas}_{\text{iniciais}}| \quad (5-8)$$

Os valores numéricos da configuração inicial foram tomados como referência para limitar o domínio de procura para o novo grupo de parâmetros. Os limites de cada variável estão fixados na Tabela 5-3.

Tabela 5-3 Limites das variáveis de otimização da Aplicação 2.		
Limite inferior	Variável	Limite superior
0.050 m	a	0.100 m
0.101 m	c	0.200 m
1	μ_r	100
0.5×10^7 S/m	σ	7.0×10^7 S/m

Para esse problema de otimização, utilizou-se o SSGA acoplado ao mecanismo de redução de intervalo. Um trabalho adicional foi escolher o número de gerações e tamanho da população. A relação entre número de gerações e tamanho da população deve ser tal que proporcione a convergência da população em duas instâncias: a primeira para realização da redução de intervalo e a segunda para convergência do SSGA para o ponto solução final. Outro fator importante a ser considerado é o número de cálculo de funções, pois o tempo de processamento de cada função desempenho é muito alto (veja Apêndice B). Por esses motivos descritos, através de simulações realizadas previamente, escolheu-se uma população de 16 indivíduos e número de gerações igual a 76. Realizando 20 execuções com o SSGA, obteve-se sucesso em 12, sendo admitido como solução factível quando $ff(x) < 10^{-3}$. A Tabela 5-4 apresenta os resultados das 12 execuções que tiveram sucesso, enquanto que a Figura 5-5 mostra a distribuição de J no raio médio da casca cilíndrica para a configuração antes da otimização e para algumas configurações otimizadas.

Tabela 5-4
Soluções factíveis para Aplicação 2.

	a (m)	c (m)	σ (S/m)	μ_r
1	0.085830264	0.10955339	4.7862389e7	1.0170064
2	0.087931454	0.10254994	2.4468062e7	2.4600134
3	0.088950776	0.10919688	3.1532630e7	1.7523490
4	0.089605391	0.10215415	2.8839872e7	2.2061094
5	0.091419727	0.10781612	3.0476699e7	1.7130344
6	0.091518909	0.11057762	2.1680453e7	2.3752349
7	0.093595691	0.13020721	0.76561785e7	4.4171267
8	0.094085507	0.11530903	2.2168951e7	2.2170476
9	0.097238079	0.11867177	2.6265343e7	1.7380231
10	0.099874876	0.11383159	3.7102873e7	1.2084738
11	0.099908449	0.11937273	0.98184144e7	4.2509537
12	0.099943541	0.12173843	1.0467086e7	3.6799219

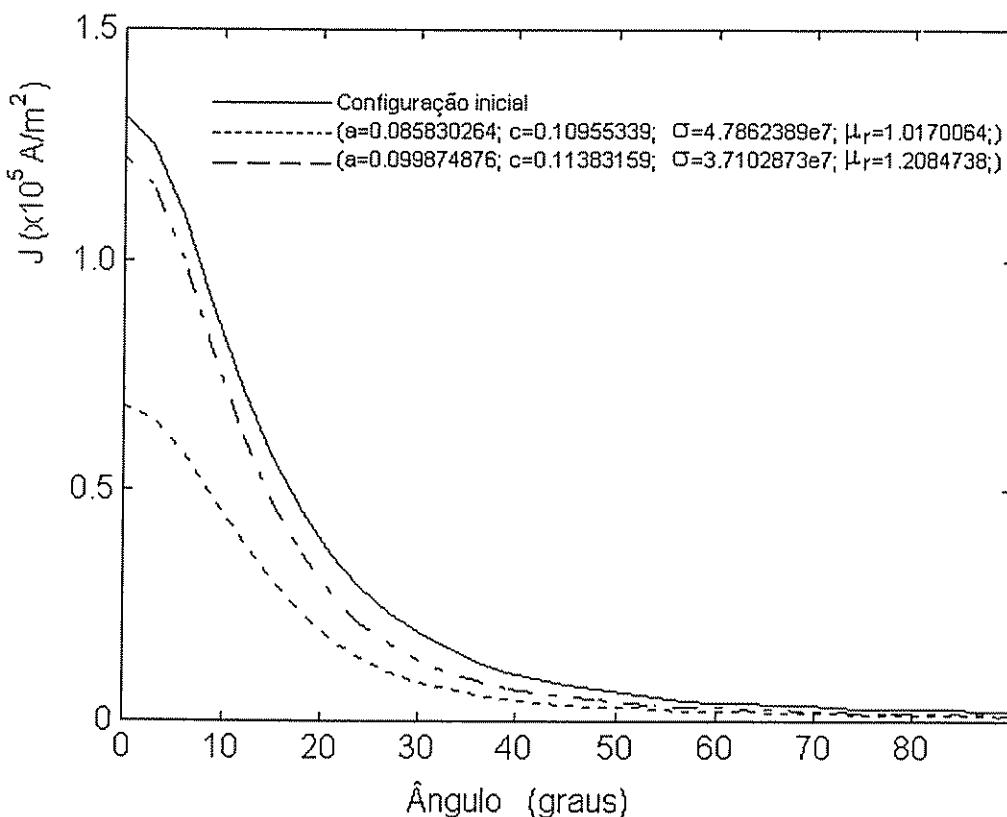


Figura 5-5: Distribuição de J no raio médio da configuração inicial e de duas configurações otimizadas para o caso do filamento interior à casca cilíndrica.

5.2.3 Correntes Induzidas numa Casca Cilíndrica Devido a uma Corrente num Filamento Paralelo Externo ao Cilindro

A terceira aplicação é bem parecida com a segunda, sendo a diferença o posicionamento do condutor, que agora se encontra externamente à casca condutora. A Figura 5-6 mostra a nova situação. Quanto à formulação, a diferença entre a Aplicação 2 e esta, consiste nas constantes F_n e C_n , as quais foram reescritas nas equações 5-9 e 5-10.

$$F_n = \frac{j^{1/2}pl}{2\pi D_n} \left(\frac{c}{b} \right)^n \left(\frac{1}{c} + q_n \right) \left[\frac{\mu_r n K_n(j^{1/2}pa)}{j^{1/2}pa} + K_{n-1}(j^{1/2}pa) + n \frac{K_n(j^{1/2}pa)}{j^{1/2}pa} \right] \quad (5-9)$$

$$C_n = -\frac{j^{1/2}pl}{2\pi D_n} \left(\frac{c}{b} \right)^n \left(\frac{1}{c} + q_n \right) \left[\frac{\mu_r n I_n(j^{1/2}pa)}{j^{1/2}pa} - I_{n-1}(j^{1/2}pa) + n \frac{I_n(j^{1/2}pa)}{j^{1/2}pa} \right] \quad (5-10)$$

Nas equações 5-9 e 5-10, $q_n=0$ quando $n=0$ e, para $n>0$, q_n é igual a $1/c$.

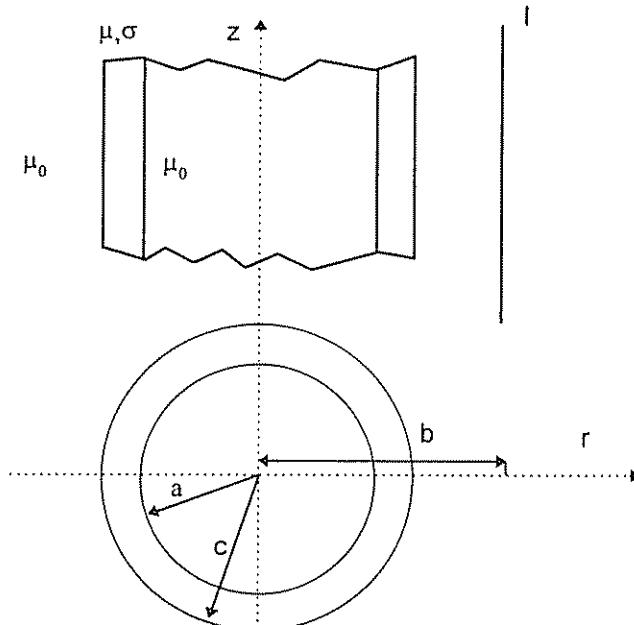


Figura 5-6: Cortes Longitudinal e transversal da casca cilíndrica infinita e o filamento de corrente externo à casca.

Os valores numéricos dos parâmetros da configuração inicial são: $f=50\text{Hz}$, $a=0.100\text{ m}$, $c=0.112\text{ m}$, $b=c/0.9$ ($\approx 0.124\text{ m}$), $I=100\text{ A rms}$ e o material é o alumínio. O problema de otimização para esta última aplicação é encontrar uma alternativa forma geométrica (novos a , b e c) de tal maneira que as perdas na seção transversal fossem mantidas. Então, definiu-se a função objetivo $ff(x)$ como:

$$\min ff(x) = |\text{perdas} - \text{perdas}_{\text{iniciais}}| \quad (5-11)$$

onde as perdas são calculadas a partir da equação (5-7). Aplicando o conjunto de parâmetros na equação (5-7), obteve-se 0.0544515 W/m (calculando a equação 5-3 para $n=20$ termos). A Tabela 5-5 mostra o domínio de procura para cada variável.

Tabela 5-5 Limites das variáveis de otimização da Aplicação 2.		
Limite inferior	Variável	Limite superior
0.010 m	a	0.110 m
0.221 m	b	0.600 m
0.111 m	c	0.220 m

Dentre as três aplicações, esta é a menos complicada. Vários grupos de parâmetros satisfazem a função objetivo. Por isso, o número de cálculo de funções necessário foi bem inferior aos utilizados até aqui. Os parâmetros do SSGA utilizados foram a população com 10 indivíduos e o número de gerações igual a 15. Das 20 execuções, 14 tiveram sucesso. A Tabela 5-6 mostra os resultados das 14 execuções que tiveram sucesso. A Figura 5-7 mostra a distribuição de J no raio médio da casca cilíndrica para algumas configurações de geométricas.

Tabela 5-6
Soluções factíveis para Aplicação 3.

	a (m)	c (m)	b (m)
1	0.0565926	0.1317475	0.2244121
2	0.0567086	0.1271769	0.2313636
3	0.0624789	0.1185379	0.1471292
4	0.0689238	0.1400256	0.2099402
5	0.0774947	0.1240965	0.2373435
6	0.0776107	0.1115289	0.2412877
7	0.0777633	0.1242129	0.2367536
8	0.0890429	0.1157636	0.2350533
9	0.0957143	0.1323562	0.2362909
10	0.0983969	0.1127265	0.2367073
11	0.1003088	0.1308909	0.2323537
12	0.1030906	0.1423691	0.2296402
13	0.1060997	0.1378683	0.2317221
14	0.1069573	0.1232482	0.2230704

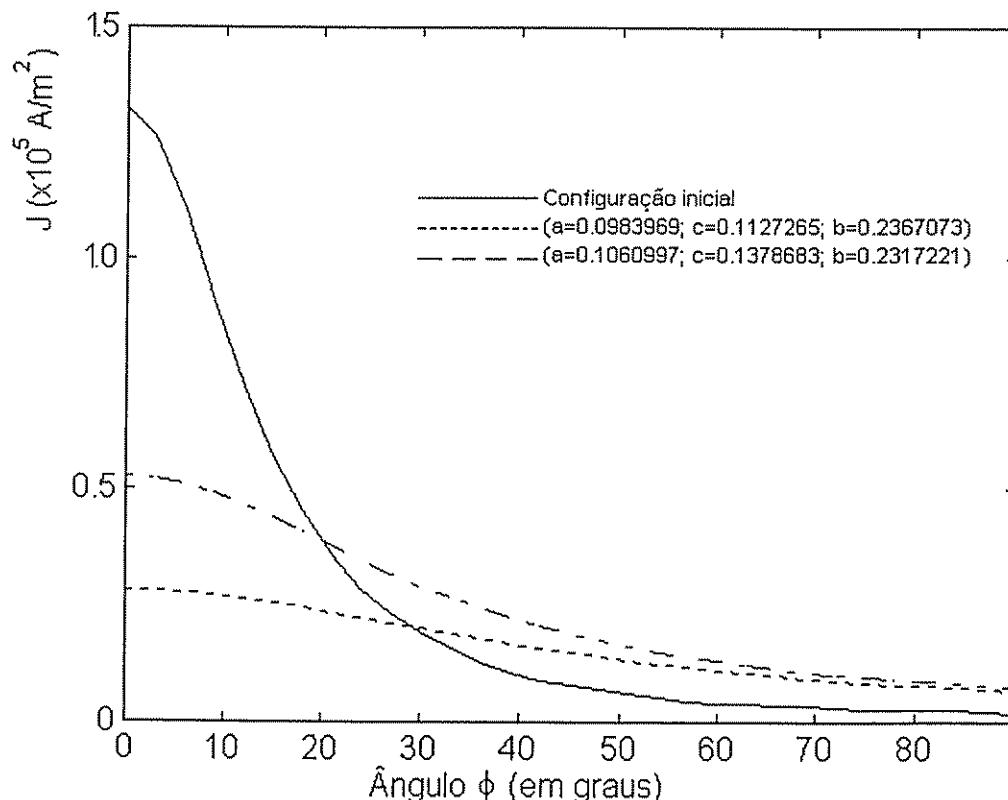


Figura 5-7: Distribuição de J no raio médio da configuração inicial e de duas configurações otimizadas para o caso do filamento exterior à casca cilíndrica.

5.3 Validação dos Resultados Obtidos

Os Algoritmos Genéticos precisam apenas da avaliação da função desempenho para realizar a busca pelo ponto ótimo. O resultado otimizado estará de acordo com a formulação feita na função desempenho, mesmo que ela não corresponda fielmente ao problema tratado. No caso das funções teste, não foi necessário validar as formulações, o importante era conhecer as soluções globais. Entretanto, no caso das aplicações deste capítulo, antes de se procurar pelas soluções buscou-se ainda validar as formulações.

Tratando inicialmente a Aplicação 1, validou-se a formulação desenvolvida em C++ com uma outra realizada no Matlab por um aluno de iniciação científica (Adriano Vilela Barbosa), pois nas referências [22] e [40] não são mostrados resultados experimentais sobre essa aplicação.

No entanto, para as Aplicações 2 e 3, os autores em [40] apresentaram resultados numéricos. A Figura 5-8 mostra o resultado encontrado em [40].

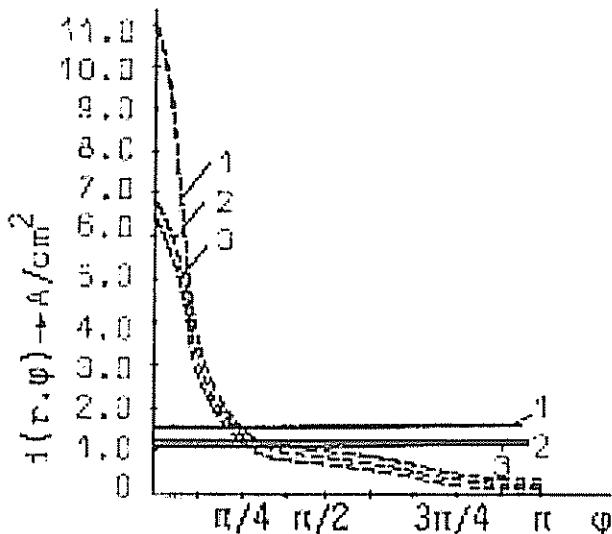


Figura 5-8: Gráfico retirado de [40].

Os números 1 ($r=0.100$), 2 ($r=0.106$) e 3 ($r=0.112$) indicam as curvas da densidade de corrente J em função do ângulo ϕ para um determinado raio fixo. As linhas contínuas que representam o efeito pelicular são obtidas quando a equação 5-3 é analisada para $n=0$. Segundo a referência [40], a Figura 5-8 foi obtida utilizando-se os seguintes parâmetros: $a=0.100$ m; $b=a*0.9$ m; $c=0.112$ m; $\mu_r=1$; $f=50$ Hz; $I=100$ A rms.

Na referência não foram citados nem o valor de σ , nem qual número de termos utilizados na equação 5-3. Buscou-se os artigos originais que propiciaram trabalhos da referência [40] e ficou constatado que a resistividade ρ usada foi 0.28×10^{-7} , logo, $\sigma = 3.5714286 \times 10^7$. Então para o mesmo grupo de parâmetros e, utilizando $n=20$ na equação 5-3, obteve-se no programa desenvolvido neste trabalho a Figura 5-9.

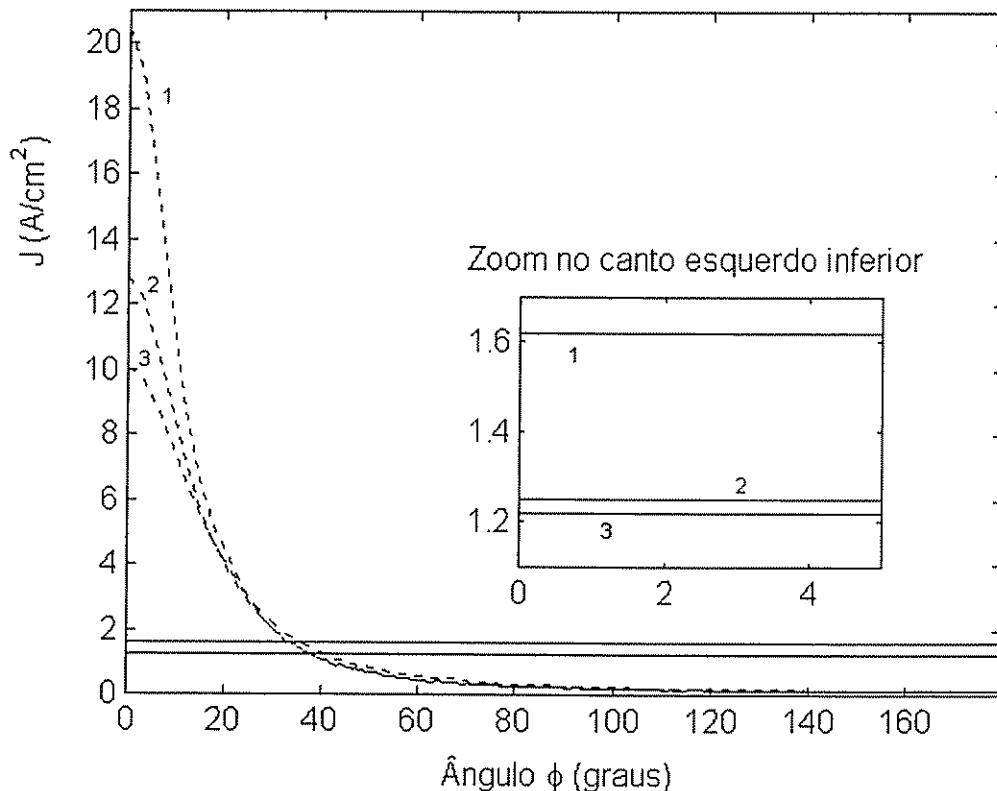


Figura 5-9: Distribuição de J na casca cilíndrica, segundo a configuração utilizada na Figura 5-8, segundo a implementação realizada nesta dissertação.

Comparando a Figura 5-8 com Figura 5-9, nota-se que, apesar dos resultados do efeito peculiar descrito pela linha contínua ($n=0$) estarem de acordo, o resultado para a corrente total induzida (20 termos na equação 5-3), representado pelas linhas pontilhadas não está. Então buscou-se outra maneira para validar quais resultados estão corretos. Partiu-se, então, de um *software* desenvolvido no CPDEE, chamado *Softwave Numerical Program* (dissertação de mestrado do aluno André Gustavo Lomônaco) que resolve vários tipos de problemas eletromagnéticos 2D, utilizando os métodos FEM (*Finite Element Method*), BEM (*Boundary Element Method*) e MoM (*Method of Moment*). O autor do *software*, juntamente com outra aluna de mestrado (Simone Aparecida Viana), implementou este caso de correntes induzidas utilizando-se do método FEM. O resultado obtido encontra-se na Figura 5-10.

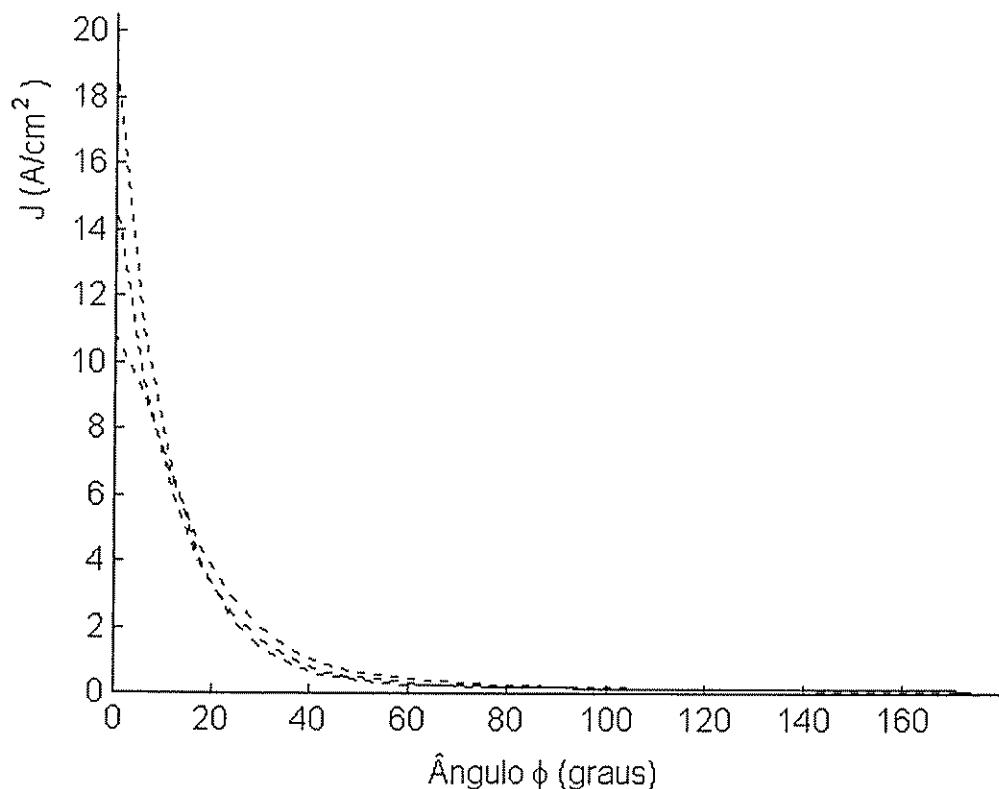


Figura 5-10: Distribuição de J na casca cilíndrica, segundo a configuração utilizada na Figura 5-8, segundo o método FEM.

Para facilitar a comparação entre o resultado da formulação analítica implementada nesta dissertação e o resultado vindo do FEM, traçou-se conjuntamente duas curvas ($r=0.100$ e $r=0.112$) para cada método e o ângulo ϕ foi variado apenas de 0° a 60° . A Figura 5-11 mostra a visualização gráfica.

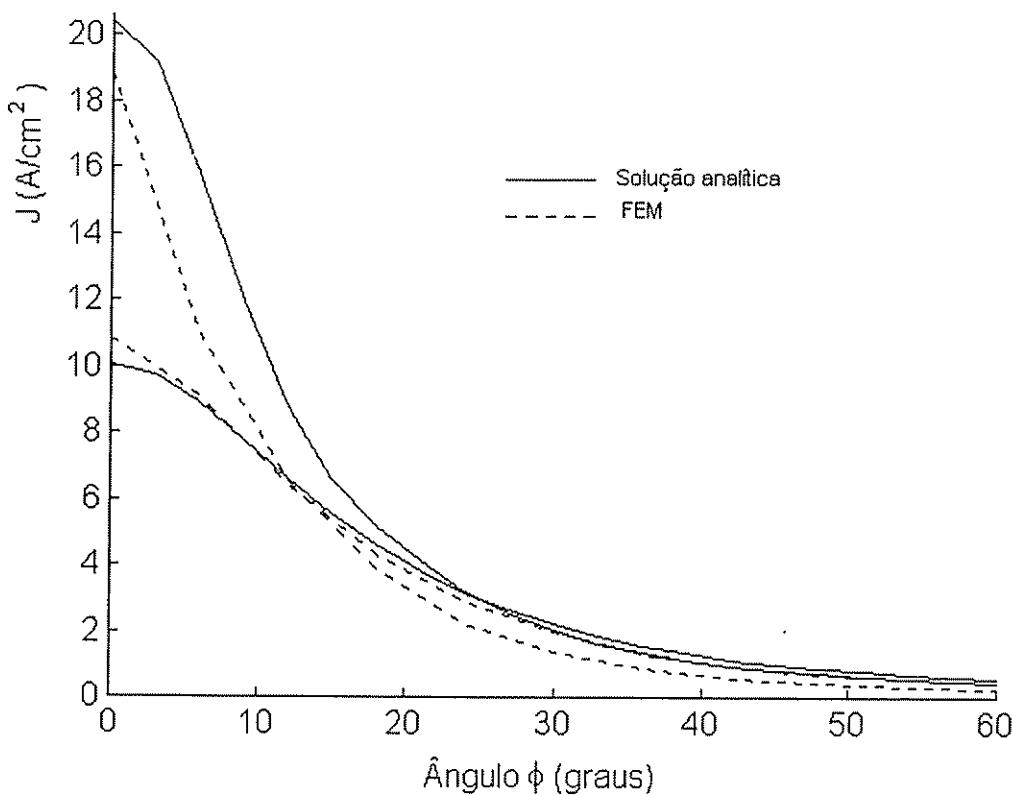


Figura 5-11: Solução analítica X FEM.

Observando a Figura 5-11, nota-se que as soluções analíticas e as obtidas através do FEM estão próximas. Portanto, concluiu-se que o gráfico da Figura 5-9 está incorreto. Resultados similares foram obtidos quando comparou-se o caso do filamento externo (Aplicação 3) com o FEM e mais uma vez os resultados da referência não coincidiram. Uma possível explicação seria o número de termos utilizados, em [40], na equação 5-3. Testou-se a equação 5-3 para várias ordens e conseguiu-se uma razoável aproximação com os resultados da referência [40], quando $n=4$.

A partir de resultados como os da Figura 5-11, as formulações de J foram consideradas implementadas corretamente.

5.4 Conclusão

Neste capítulo, o SSGA foi utilizado para resolver três problemas de otimização envolvendo correntes induzidas. As funções objetivo destas aplicações são complicadas, fazendo uso das funções de Bessel e envolvendo integrais. Estas funções são caras computacionalmente se comparadas ao tempo gasto com as funções analisadas no Capítulo 4. Por esse motivo, os parâmetros tamanho da população e número de gerações foram dimensionados de forma a deixar o SSGA

trabalhar com um pequeno número de cálculos de função, mesmo que, para isso, se perca um pouco de performance quanto ao número de execuções com sucesso em relação ao número de execuções realizadas.

6. Conclusão

Ao longo desta dissertação, procurou-se fazer um estudo aprofundado em relação aos Algoritmos Genéticos. Para tanto, no inicio da dissertação, buscou-se enfatizar a importância da otimização e, em seguida, identificar os GAs dentre os três grupos de algoritmos de otimização (Determinísticos, Enumerativos e Estocásticos). Cada grupo de algoritmos tem suas particularidades, trazendo consigo vantagens e desvantagens. Os Métodos Determinísticos normalmente requerem cálculo de derivadas, procuram por uma solução local, mas, por outro lado, são de rápida convergência. Os Enumerativos garantem a solução global, mas, para conseguir esse objetivo, precisam verificar todo espaço viável. Os Estocásticos buscam a solução global por intermédio de amostragem do espaço viável. Para cada grupo de algoritmos de otimização, foram introduzidas referências de modo a auxiliar os interessados. Enquadramento no grupo estocástico, os GAs fazem uso de técnicas de probabilidade para guiar a amostragem por toda a região de procura. Utilizando-se de amostragem, a pesquisa pela solução global é descentralizada, e portanto, é feita em mais regiões. Em seguida, apresentou-se a origem dos GAs a partir da analogia com sistemas naturais. Alguns dos termos genéticos (população, reprodução/seleção, cruzamento e mutação) foram introduzidos nesse ponto. Para facilitar o entendimento do funcionamento de um Algoritmo Genético, executou-se passo a passo um exemplo simples, como Goldberg fez em seu livro [11]. Nesse primeiro contato, a intenção foi facilitar o entendimento sobre o comportamento de um GA perante um determinado tipo de problema. Outro ponto importante no Capítulo 1, foi a apresentação das três linhas de pesquisa sobre os GAs do ponto de vista de Davis [4] (*Sistemas Classificadores, Análise de Performance e GA Paralelo*). Destas, estudou-se somente o tema *Análise de Performance*. Esta linha de pesquisa procura por mecanismos, parâmetros e outros artifícios que auxiliem os GAs a melhorarem seu desempenho na resolução de problemas. Certamente, o bom conhecimento de itens de melhoria de performance auxilia os pesquisadores de *Sistemas Classificadores e de GA Paralelo*.

Apresentado os GAs em linhas gerais no Capítulo 1, procurou-se durante o Capítulo 2 introduzir importantes conceitos para entender como os GAs realizam a busca pela melhor solução. A análise matemática foi feita sob o ponto de vista de *esquema* e também utilizando alguns mecanismos bem definidos: o cruzamento com um único ponto de corte, a mutação com um pequeno valor de probabilidade e o método da Roleta como mecanismo de seleção. O resultado desta análise está mostrado na equação 2-5 e recebe o nome de *Teorema Fundamental dos GAs*. A equação 2-5 demonstra que esquemas de pequeno *comprimento de definição*, de baixa *ordem* e alto valor de desempenho são fortes soluções parciais do problema e, portanto, terão grandes chances de prosseguirem nas próximas gerações. A partir da associação dessas soluções parciais, os GAs descobrem novas soluções (*building blocks*). Com esse conceito de "montagem" de soluções parciais, os GAs reduzem a complexidade do problema como um todo. Também sob o ponto de vista de *esquemas*, demonstrou-se que, quando os GAs avaliam o desempenho de todos os n_{pop} membros da população, analisam-se implicitamente n_{pop}^3 esquemas (*paralelismo implícito*). A visão geométrica mostrada na Figura 2-3 facilita o entendimento deste paralelismo e mostra como o algoritmo explora as semelhanças entre os diversos esquemas conduzindo a pesquisa às regiões mais diversas.

O Capítulo 3 teve a finalidade de descrever os itens que podem trazer melhorias de desempenho aos GAs, e também guiar o leitor apresentando algumas ferramentas já experimentadas por outros autores. No corpo do capítulo foram descritos:

- tipos de codificação (como o código binário e o Gray);
- como mapear as variáveis no espaço de procura;
- como transformar uma função objetivo em uma função desempenho (introdução de restrições, evitando a negatividade da função objetivo e transformando um problema de minimização em maximização);
- parâmetros dos GAs (valores quantitativos e qualitativos);
- métodos de escalonamento (Linear e Sigma);
- métodos de mutação (para códigos binário, Gray e para outros códigos que trabalham com arranjos e permutações);
- métodos de seleção (Roleta, Torneio, SRS, DS e SUS);

- métodos de cruzamento (para códigos binário e Gray: 1ponto, 2pontos, CPV, CEVI e Uniforme; para códigos que trabalham com arranjos e permutações: PMX, OX, CX e ERO);
- método da inversão
- critérios de convergência (por número de gerações, por estagnação da melhor solução, por convergência da população e também associação de mais de um critério);
- métodos de medir a performance (gráficos: f_{med} , f_{max} , *on-line performance* e *off-line performance*);
- métodos de adaptação dinâmica (PI, FF e DF);
- tipos diferentes de GAs (SGA, SSGA e RGA);
- técnicas híbridas (implementação e referências bibliográficas para o caso de acoplamento de GAs aos Métodos Determinísticos);
- técnicas de nicho e subpopulações (métodos de Goldberg e Spears);
- mecanismo de redução de intervalo

Itens como acoplamento de técnicas híbridas, formação de subpopulações e mecanismo de redução de intervalo mostram a flexibilidade dos GAs em permitirem a introdução de métodos diferentes à sua estrutura. Outro ponto importante foi o estudo da adaptação dinâmica dos operadores genéticos. Esse assunto trouxe uma perspectiva à idéia de adaptação estender-se a todos os métodos e parâmetros utilizados nos GAs.

O objetivo do Capítulo 4 foi a apresentação de resultados experimentais. No primeiro instante, testes foram realizados e, no segundo, alguns resultados de outros autores foram buscados.

Para a realização dos testes, foi escolhido o *software* GALib (MIT), a partir do qual foram implementadas as modificações necessárias. Em seguida, foram escolhidas três funções teste (Degrau, Rastrigin e Picos) sobre as quais foi feita toda a análise numérica. Também foram fixados alguns parâmetros e métodos:

- $p_c=0.6$; $p_m=0.001$, $n_{pop}=50$ (parâmetros de De Jong [7]);
- seleção: Roleta;
- cruzamento: 2pontos;

- escalonamento: Linear (1.2).

Escolhido o *software*, parâmetros, métodos e as funções teste, o próximo passo foi pesquisar quais técnicas, dentre as conhecidas na bibliografia, eram as mais robustas, observando-se o desempenho conjunto dos três GAs apresentados. Nesses testes, foram analisadas inicialmente, as técnicas do ponto de vista das funções e depois do ponto de vista dos GAs. Os primeiros testes foram em relação às características intrínsecas de cada GA, saíram-se melhor o RGA com substituição de apenas um indivíduo, SGA com elitismo, e SSGA com intervalo $G=0.8$. Depois disso, foram pesquisados outros tipos de seleção, escalonamento e cruzamento. Notaram-se que pequenas melhorias, a ponto de não se poder afirmar que este ou aquele método é melhor. Isso tendo em vista que os GAs de natureza estocástica e a existência flutuações nos resultados das execuções é perfeitamente aceitável. O que pode ser concluído é que os cruzamentos Uniforme e 2pontos são boas opções de cruzamento, o método da Roleta esteve sempre nas primeiras posições nos testes de métodos de seleção e também que o escalonamento é importante e não necessariamente deva estar dentro da faixa [1.2-2.0]. Depois dos testes de cruzamento, as técnicas de adaptação dinâmica dos operadores cruzamento e mutação foram introduzidas. Com a adaptação, tenta-se controlar a diversidade genética dentro da população, sendo este um fator importantíssimo para condicionar os GAs a encontrarem a solução principal. Os resultados com as técnicas de adaptação ajudaram efetivamente num melhor desempenho dos GAs. A Figura 4-14 e Figura 4-15 sintetizam os passos desta primeira fase de testes.

Outra fase de testes importante foi a verificação do acoplamento das técnicas híbrida (GA + BFGS), de nicho (Goldberg) e de redução de intervalo (introduzido nesta dissertação). Quanto à técnica híbrida utilizada, observou-se que ganha-se precisão na solução, mas não necessariamente há melhoria de desempenho quanto ao número de execuções com sucesso. Em relação à técnica de formação de nichos, consegue-se o objetivo que é a distribuição da população de acordo com a importância da região, mantendo-se a diversidade. Mesmo trabalhando com diversidade (população distribuída), o GA com formação de nichos, não obteve bons resultados de convergência para a solução global. Por outro lado, a técnica de redução de intervalo foi a que proporcionou melhores resultados de convergência para a solução global. O motivo é que a redução é baseada no melhor indivíduo de uma população já homogeneizada ($m_{dg} \approx 1$).

O último assunto do Capítulo 4 tratou de resultados de artigos de outros autores. A intenção foi cobrir alguns testes não implementados nesta dissertação sobre os GAs. Analisando um trabalho de Goldberg & Outros [13] em 1992, no qual conseguiram coletar aproximadamente 1200 referências e ainda observaram um crescimento do número de trabalhos sobre GAs na faixa de 37% ao ano, desde 1986, nota-se que é impossível fazer um apanhado de tudo o que já foi publicado sobre o tema até hoje.

No Capítulo 5, o SSGA (com critério de adaptação dinâmica DF) foi utilizado para resolver três problemas de otimização envolvendo correntes induzidas. O mecanismo de redução de intervalo foi inserido apenas na Aplicação 2. Os resultados foram satisfatórios, considerando-se que os parâmetros foram quantificados de modo que o número de cálculo de funções não fosse grande, pois o tempo computacional requerido para o cálculo de cada função objetivo era bem grande (veja Apêndice B), se comparado ao tempo de processamento obtido no Capítulo 4.

Finalmente, os apêndices foram inseridos com o objetivo de complementação do texto. Enquanto o Apêndice A descreve exemplos de aplicações de GAs, o Apêndice B mostra o tempo computacional gasto durante a fase de testes (Capítulo 4) e durante a fase de execução das aplicações (Capítulos 4 e 5). Já o Apêndice C lista implementações (*softwares*) de GAs que são distribuídos gratuitamente na Internet, inclusive com o código fonte. Esse último apêndice deve ser muito útil para os interessados em verificar outras implementações.

Espera-se que esta dissertação sirva como base para aqueles interessados em conhecer os Algoritmos Genéticos, aplicá-los a problemas de otimização diversos e também deixar o leitor apto à leitura e entendimento de outros trabalhos relativos ao tema.

7. Apêndice A: Algumas Aplicações que Utilizam GAs

Referência: Capítulo 12 de [4].

Tema: "Um Algoritmo Genético aplicado à geração de trajetória de um braço robô".

A geração de trajetória de robôs ou de qualquer uma de suas partes é um problema pertencente à classe dos processos dependentes de ordenação, ou seja, processos em que a seqüência executada tem efeito fundamental sobre o desempenho. Para a resolução desse problema, um GA modificado é apresentado para otimizar o caminho intermediário entre uma posição A e outra B. Essas modificações (que estão descritas no artigo com detalhes) foram feitas para melhorar as características de herança de trajetória, e também para considerar melhor as regras (de movimentação) e uma seqüência de melhor qualidade. O braço robô analisado possui três hastes e três pontos de articulação. O autor concluiu que a utilização do GA foi um sucesso.

Outros trabalhos que tratam de assuntos semelhantes são:

- F. Harashima, H.Hashimoto & D. Kang, "Path generation for mobile robot navigation using genetic algorithm", Proceedings of the 1995 IEEE IECON 21st International Conference on Industrial Electronics, Control, and Instrumentation, pp. 167-172, vol. 1, 1995.
- H.Zhuang, J.Wu & W.Huang, "Optimal Planning of Robot Calibration Experiments by Genetic Algorithms", Proceedings of the 1996 IEEE International Conference on Robotics and Automation, pp. 981-986, vol. 2, 1996.

Referência [39].

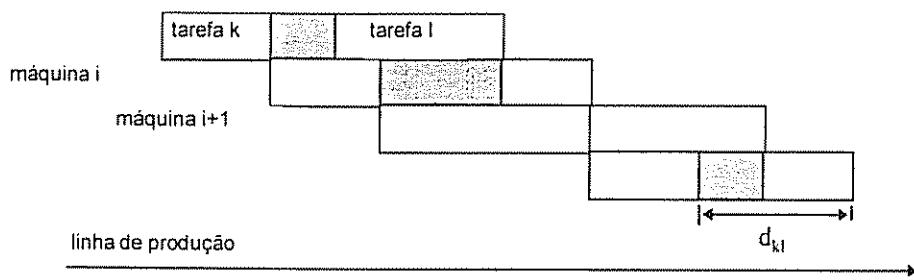
Tema: "Aplicação de um GA Paralelo na resolução de um problema de fluxo de fábrica".

Este trabalho investiga a aplicação de um GA trabalhando em paralelo (PGA) para tratar um caso especial de linha de produção, conhecido como $n/m/P/C_{max}$ (n -tarefas, m -máquinas, P -permutação e C_{max} -tempo total do programa de produção).

Neste problema deve-se respeitar as seguintes restrições:

- todos as tarefas seguem uma mesma seqüência de máquinas (1,2,3, ..., m).
- quando uma tarefa sai de uma máquina i , imediatamente ela passa à máquina $i+1$, ou seja, o tempo de espera entre duas máquinas é zero para a mesma tarefa.
- uma máquina se ocupa de apenas uma tarefa por vez. Portanto, uma tarefa só utilizará uma máquina, quando a mesma estiver desocupada.

O processo do fluxo de fábrica fica melhor explicado pela figura que segue.



O que se deseja otimizar é o tempo de espera total $d_{1, \dots, n}$ entre a primeira e a última tarefa. Então a função objetivo é a seguinte.

$$C_{\max} = \sum_{j=0}^n d_{p(j)p(j+1)}$$

Outras referências a problemas similares são:

- E.S.H.Hou, N.Ansari & H.Reng, "A genetic algorithm for multiprocessor scheduling", IEEE Transactions on Parallel and Distributed Systems, pp. 113-120, feb, 1994.
- T.Yamada, & R.Nakano, "A Genetic Algorithm Applicable to Large-Scale Job-Shop Problems", Parallel Problem Solving From Nature, 2, pp. 281-290, 1992.
- W.B.Langdon, "Scheduling Planned Maintenance of the (UK) National Grid, Available: cs.ucl.ac.uk:/genetic/papers/grid_aisb-95.ps, 1995.

Referência [44].

Tema: "GA aplicado à otimização da geometria de bucha de transformador".

Neste artigo, os autores procuram aperfeiçoar a geometria da bucha isolante de transformador, de forma que o campo em sua superfície não ultrapassasse um valor limite predeterminado. Para avaliação de cada geometria, foi utilizado o método de equações integrais de fronteira.

Os autores observaram a facilidade de aplicação de GAs em relação aos Métodos Determinísticos que fazem uso da informação do gradiente. Por outro lado, indicaram superioridade das técnicas determinísticas sobre os GAs, se analisada a quantidade final de cálculos de função. Porém, advertiram que a característica de globalidade pode valer a pena, pois o problema tratado pode ter muitas soluções locais.

Outras referências a problemas de geometria são:

- G.F.Üler, "Genetic Algorithms in Design Optimization of Eletromagnetics Devises", Ph.D. thesis, Florida International University, 1994.
- J.A.Vasconcelos, L.Krähenbühl, L.Nicolas & A.Nicolas, "Design Optimization Using the BEM Coupled with Genetic Algorithm", IEE-Proceedings CEM94, Nottingham, UK, (1994).
- O.A.Mohammed, "Practical Issues in the Implementation of Genetic Algorithms to Electromagnetics Design Problems", Segundo Congresso Brasileiro de Eletromagnetismo, nov 96, pp. 207-213, Ouro Preto, Brasil.

8. Apêndice B: Tempo Computacional

Um item importante que pode viabilizar ou não um projeto quando se trata de execuções de programas e/ou simulações é o tempo computacional (tempo de máquina). Este apêndice preocupa-se em registrar o tempo gasto em cada fase de testes. Para entendimento da apresentação desses resultados, algumas considerações devem ser feitas:

- A máquina utilizada: HP Apollo 9000 720 T.
- O registro dos tempos computacionais foi dividido em três tabelas. A primeira trata da fase de testes até os critérios de adaptação, a segunda tabela descreve os resultados provenientes da utilização das técnicas híbrida, de nicho e de redução de intervalo. A última tabela mostra o tempo gasto com as aplicações.
- Os resultados fixados estão de acordo com o procedimento descrito no texto. Por exemplo, para se construir a Tabela 4-6, que trata dos métodos de seleção na função Degrau. Sabe-se que cada GA (RGA, SGA e SSGA) foi executado 30 vezes, sendo que a cada execução no máximo foram realizados 2050 cálculos. Como são 5 os métodos de seleção analisados, o número máximo de cálculos realizados para a Tabela 4-6 são $30 \times 2050 \times 5 = 307500$. Nota-se, portanto, que é fundamental o conhecimento do procedimento.
- Os dados deste apêndice respectivos aos testes do Capítulo 4 não foram coletados durante cada experimento. Por isso, todos os testes tiveram que ser refeitos para anotar o tempo computacional gasto. Como os GAs trabalham com mecanismos estocásticos, obviamente houve variações quanto ao número de cálculos e ao tempo, as quais não devem ser significativas, já que todos os processos foram realizados 30 execuções. Quanto ao tempo das aplicações, estes foram coletados juntamente com a execução.
- O tempo computacional de algumas execuções foi grande se comparado com outras. Em alguns desses casos, teve-se dificuldade em medir o tempo computacional de uma única vez. Quando isso ocorreu, obteve-se o tempo de processamento final, fazendo-se regra de três com casos de menor magnitude.
- A Tabela 4-17 e a Tabela 4-18 apenas registram resultados de procedimentos para encontrar os melhores métodos para cada GA. Como os procedimentos foram similares aos já feitos anteriormente, ponderou-se não haver necessidade de novo registro.

Ciente destas considerações, os resultados são apresentados nas tabelas que seguem:

Tabela 8-1
Tempo computacional da primeira fase de testes (Capítulo 4).

Item	RGA		SGA		SSGA		TCT
	NCF	TC	NCF	TC	NCF	TC	
Tabela 4-3	-	-	285000	00:04:20	-	-	00:04:20
Tabela 4-4	92071	00:01:28	-	-	-	-	00:01:28
Tabela 4-6	57159	00:00:46	291450	00:01:20	114020	00:00:36	00:02:42
Tabela 4-7	90109	00:01:19	305500	00:01:19	219420	00:01:11	00:03:49
Tabela 4-8	68179	00:01:03	97500	00:00:28	93780	00:00:35	00:02:06
Tabela 4-10	107819	00:00:38	307500	00:00:59	150300	00:00:38	00:02:15
Tabela 4-11	150614	00:01:54	307500	00:02:13	245180	00:02:01	00:06:08
Tabela 4-12	73582	00:01:01	97500	00:00:56	89420	00:00:58	00:02:55
Tabela 4-16	29160817	09:00:13	54292500	04:41:48	34189140	03:42:22	17:24:23
Figura 4-4	-	-	-	-	1575042	00:11:06	00:11:06
Figura 4-8	663090	00:07:12	3751500	00:11:12	1433580	00:05:42	00:24:06
Figura 4-9	354582	00:03:30	3136500	00:09:24	812100	00:03:12	00:16:06
Figura 4-10	1241874	00:22:54	3751500	00:22:06	2994060	00:21:03	01:06:00
Figura 4-11	537066	00:09:12	3136500	00:18:30	1556340	00:11:12	00:38:54
Figura 4-12	795440	00:14:20	1189500	00:09:18	1090140	00:09:54	00:33:32
Figura 4-13	431064	00:07:24	994500	00:06:36	828180	00:06:18	00:20:18

Tabela 8-2
Tempo computacional dos testes de técnicas mais avançadas (Capítulo 4).

Número de Variáveis	BFGS		Nichos		Redução	
	NCF	TC	NCF	TC	NCF	TC
2	184500	00:01:56	61200	00:00:30	2337000	00:09:30
4	414000	00:05:43	107100	00:02:12	4548600	00:19:00
6	918000	00:29:00	137700	00:05:24	8322000	00:57:00
8	1512000	00:59:00	399300	00:22:54	11080800	01:44:30
10	2196000	02:28:00	588900	00:47:12	17955000	05:42:00
12	2970000	04:31:00	666900	01:18:18	20349000	6:29:30
14	3834000	06:45:15	1025100	02:53:24	29412000	8:33:00
16	4788000	12:09:00	1326000	04:50:36	36366000	18:31:30
18	5832000	19:41:00	1731900	05:44:00	57285000	36:34:30
20	6966000	27:29:00	2113800	08:42:00	74179800	47:39:30

Tabela 8-3
Tempo computacional das aplicações (Capítulo 5).

	NCF	TC
Aplicação 1	19600	01:29:34
Aplicação 2	18560	143:59:41
Aplicação 3	2600	19:59:53

9. Apêndice C: Softwares de GAs

Os Algoritmos Genéticos têm atraído a atenção de muitos pesquisados ([13]), alguns implementaram códigos computacionais e os distribuem gratuitamente. Então, pode-se ganhar tempo no desenvolvimento de um GA, partindo de um outro pronto. A referência [17], que pode ser obtida na Internet, lista vários códigos (comerciais ou não) e os descreve a partir de suas características principais. Neste apêndice, apenas listaram-se os códigos não comerciais.

Nome	S.O.	Ling.	Preço	Autor/Contato
BUGS	X11	C	grátis	Joshua Smith <jrs@media.mit.edu>
DGenesis	Unix	C	grátis	Erick Cantu-Paz <ecantu@lamport.rhon.itam.mx>
DOUGAL	DOS	Pascal	grátis	Brett Parker <b.s.parker@durham.ac.uk>
ESCaPaDE	Unix	C	grátis	Frank Hoffmeister <hoffmeister@ls11.informatik.uni-dortmund.de>
Evolution Machine	DOS	C	grátis	Hans-Michael Voigt <voigt@max.fb10.tu-berlin.de>
GAC, GAL	Unix	C, Lisp	grátis	William Spears <spears@aic.nrl.navy.mil>
GAGA	Unix	C	grátis	Jon Crowcroft <jon@cs.ucl.ac.uk>
GAGS	DOS, Unix	C++	grátis	JJ Merelo <jmerelo@kal-el.ugr.es>
GAlib	DOS/Win, WinNT/95, UNIX	C++	grátis	Matthew <mbwall@mit.edu>
GALOPPS	DOS, Unix	C	grátis	Erik Goodman <goodman@egr.msu.edu>
GAMusic	Win	V. Basic	\$10	Jason H. Moore <jhm@superh.hg.med.umich.edu>
GANNET	Unix	C	grátis	Darrell Duane <dduane@fame.gmu.edu>
GAucsd	Unix	C	grátis	Nici Schraudolph <GAucsd-request@cs.ucsd.edu>
GA Workbench	DOS	C++	grátis	Mark Hughes <mrh@i2ltd.demon.co.uk>
GECO	Unix, MacOS	Lisp	grátis	George Williams, Jr. <george@hsvaic.hv.boeing.com>
Genesis	DOS, Unix	C	grátis	John Grefenstette <gref@aic.nrl.navy.mil>
GENEsYs	Unix	C	grátis	Thomas Baeck <baeck@ls11.informatik.uni-dortmund.de>
GenET	Unix	C	grátis	Cezary Z. Janikow <janikow@radom.umsi.edu>
Genie	Mac Think	Pascal	grátis	Lance Chambers <pstamp@yarrow.wt.uwa.edu.au>
Genitor	Unix	C	grátis	Darrell Whitley <whitley@cs.colostate.edu>
GENlib	DOS, Unix	C	\$6	Jochen Ruhland <jochenh@neuro.informatik.uni-kassel.de>
GENOCOP	Unix	C	grátis	Zbigniew Michalewicz <zbyszek@uncc.edu>
GIGA	Unix	C	grátis	Joe Culberson <joe@cs.ualberta.ca>
GPEIST	Win, OS/2	Small-talk	grátis	Tony White <arpw@bnr.ca>
Imogene	Win	C++	grátis	Harley Davis <davis@ilog.fr>
LibGA	DOS, Unix, NeXT, Amiga	C	grátis	Art Corcoran <corcoran@wiltel.com>
LICE	DOS, Unix	C	grátis	Joachim Sprave <joe@ls11.informatik.uni-dortmund.de>

Nome	S.O.	Ling.	Preço	Autor/Contato
Matlab-GA	?	Matlab	grátis	?
mGA	Unix	C, Lisp	grátis	David Goldberg <goldberg@vmd.cso.uiuc.edu>
PARAGenesis	CM	C*	grátis	Michael van Lent <vanlent@eecs.umich.edu>
PGA	Unix	C	grátis	Peter Ross <peter@aisb.ed.ac.uk>
SGA-C, SGA-Cube	Unix	C	grátis	Robert E. Smith <rob@comec4.mh.ua.edu>
Splicer	Mac, X11	C	\$1	Steve Bayer
OLKIEN	DOS, Unix	C++	grátis	Anthony Yiu-Cheung Tang <tang028@cs.cuhk.hk>
WOLF	Unix	C	grátis	David Rogers <drolgers@msi.com>

10. Bibliografia

- [1] Bazaraa M.S. & Shetty C.M., “*Nolinear Programming-Theory and Algorithms*”, ed John Wiley & Sons, 1979.
- [2] Bertsekas D.P., “*Dynamic Programming - Deterministic and Stochastic Models*”, ed. Prentice Hall, Inc, 1987.
- [3] Bland R.G., GoldFarb D., Todd M.J., “The Ellipsoid Method-A Survey”, Operations Research, vol.29, pp.1039-1091, 1981.
- [4] Davis L., “*Handbook of Genetic Algorithms*”, Van Nostrand Reinhold, New York, 1991.
- [5] De Jong K.A. & Spears W.M., “An Analysis of the Interacting Roles of Population Size and Crossover in Genetic Algorithms”.
- [6] De Jong K.A., “Adaptive System Design: A Genetic Approach”, IEEE Transactions on Systems, Man and Cybernetics, vol.10, nº 9, 566-574, september, 1980.
- [7] De Jong K.A., “Analysis of the Behavior of a Class of Genetic Adaptive Systems”, PhD. thesis, Dep.Computer and Comunication Sciences, Univ. Michigan, Ann Arbor,1975.
- [8] Deb K. & Goldberg D.E., “An Investigation of Niche and Species Formation in Genetic Function Optimization”, Proceedings of the Third International Conference on Genetic Algorithms, pp. 42-50, 1989.
- [9] Fletcher R., “*Practical Methods of Optimization*”, Volume 1, ed. John Wiley & Sons, 1980.
- [10] Goldberg D.E. & Richardson J., “Genetic Algorithms with Sharing for Multimodal Function Optimization”, Proceedings of Second International Conference on Genetic Algorithms, pp. 41-49, 1987.
- [11] Goldberg D.E., “*Genetic Algorithms in Search, Optimization and Machine Learning*”. Reading MA: Addison Wesley, 1989.
- [12] Goldberg D.E., “Optimal Initial Population Size for binary-Coded Genetic Algorithms”, TCGA Report nº 85001, Tuscaloosa: Univer. of Alabama, the Clearinghouse for Genetic Algorithms.
- [13] Goldberg D.E., Milman K., & Tidd C., “Genetic Algorithms: A Bibliography”, Department of General Engineering, University of Illinois, Illigal report nº 92008, july, 1992.
- [14] Grefenstette J.J., “Optimization of Control Parameters for Genetic Algorithms”, IEEE Transactions on Systems, Man and Cybernetics, vol. SMC-16, nº 1, pp. 122-128, january/february, 1986.
- [15] Grefenstette J.J., Gopal R., Rosmaita B.J. & Van Guch D., “Genetic Algorithms for the Traveling Salesman Problem”, Proceedings of an International Conference on Genetic

- Algorithms and Their Applications, pp. 160-168, 1985.
- [16] Hazzan S., "Fundamentos de Matemática Elementar", vol.5 -Combinatória e Probabilidade.
 - [17] Heitkoetter J. & Beasley D., "The Hitch-Hiker's Guide to Evolutionary Computation: A list of Frequently Asked Questions(FAQ)", USENET: comp.ai.genetic. Available via anonymous FTP from rtfm.mit.edu:/pub/usenet/news.answers/ai-faq/genetic/, 1994.
 - [18] Holland J.H., "Adaptation in Natural and Artificial Systems", Univ. of Michigan Press, Ann Arbor, Mich., 1975.
 - [19] Hu N., "TABU Search Method with Random Moves for Globally Optimal Design", Int. J: Num. Meth. Engineering, vol. 35, pp. 1055-1070, 1992.
 - [20] Kazarlis S.A. & Bakirtis A.G., "A Genetic Algorithm Solution to the Unit Commitment Problem", IEEE Transactions on Power Systems, vol.11, nº 1, february, 1996.
 - [21] Kirkpatrick S. & Gelatt Jr. C.D., "Optimization by Simulated Annealing", Science, vol. 220, pp. 671-680, 1983.
 - [22] Krienitz E.E., Tsiboukis T.D., Panas S.M. & Tegopoulos J.A., "Eddy Currents: Theory and Applications", Proceedings off the IEEE, vol.80, nº 10, pp. 1559-1589, october 1992.
 - [23] Kursawe F., "Evolution Strategies for Vector Optimization", Taipei, National Chiao Tung University, pp. 187-193, 1992.
 - [24] Luenberger D.G., "Introduction to Linear and Nonlinear Programming", ed. Addison-Wesley Publishing Company, 1973.
 - [25] Man K.F., Tang K.S. & Kwong S., "Genetic Algorithms: Concepts and Applications", IEEE Transactions on Industrial Electronics, vol.43, nº 5, october, 1996.
 - [26] Mendes J.M.N., Saldanha R.R. & Vasconcelos J.A., "A Hibrid Algorithm Applied to Nonlinear Optimization Problems, Segundo Congresso Brasileiro de Eletromagnetismo, Ouro Preto, pp. 222-225, novembro, 1996.
 - [27] Mühlenbein H., "Parallel Genetic Algorithms, Population Genetics and Combinatorial Optimization", Proceedings of Third International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, Calif., pp. 416-422, 1989.
 - [28] Oliver I.M., Smith D.J., & Holland J.R.C., "A Study of Permutation Crossover Operators on the Traveling Salesman Problemm", Proceedings of Second International Conference on Genetic Algorithms, pp. 224-230, 1987.
 - [29] Ribeiro Filho J.L., Treleaven P.C. & Alippi C., "Genetic - Algorithm Programming Environments", IEEE Computer, pp. 28-43, june, 1994.
 - [30] Sawaragi T., Umemura J., katai O. & Iwai S., "Fuzing Multiple Data and Knowledge Souces for Signal Understanding by Genetic Algorithm", IEEE Transactions on Industrial Electronics, vol. 4, nº 3, june, 1996.
 - [31] Shaffer J.D., Caruana R.A., Eshelman L.J. & Das R., "A Study of Control Parameters Affecting On-line Performance of Genetic Algorithms for Function Optimization", Proceedings of Third International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, Calif., pp. 51-60, 1989.
 - [32] Soares G.L. & Vasconcelos J.A., "Adaptação Dinâmica de Operadores em Algoritmos Genéticos", Segundo Congresso Brasileiro de Eletromagnetismo, Ouro Preto, pp. 155-158, novembro, 1996.
 - [33] Spears W.M. & Anand V.A., "A Study of Crossover Operators in Genetic

Programming”, Proceedings of the International Symposium on Methodologies for Intelligent Systems, eds.R.Belew and L.Booker, San Mateo, CA: Morgan Kaufmann, pp. 230-236, 1991.

- [34] Spears W.M. & De Jong K.A., “An Analysis of Multi-point Crossover”, Proceedings of the Foundations of Genetic Algorithms Workshop, Bloomington, Indiana, 1990.
- [35] Spears W.M., “Crossover or Mutation?”, Proceedings of the Secund Foundations of Genetics Algorithms Workshop, ed D.Whitley, San Mateo, CA: Morgan Kaufmann, pp. 221-237, 1992.
- [36] Spears W.M., “Simple Subpopulation Schemes”, Proceedings of Evolutionary Programming Conference, ed. World Scientific, 1994.
- [37] Srinivas M. & Patnaik L.M., “Adaptive Probabilities Of Crossover and Mutation in Genetic Algorithms”, IEEE Transactions on Systems, Man and Cybernetics, vol.24, nº 4, april, 1994.
- [38] Srinivas M. & Patnaik L.M., “Genetic Algorithms-A Survey”. IEEE Computer, pp17-26, june, 1994.
- [39] Stöppler S. & Bierwirth C., “The Application of a Parallel Genetic Algorithm to the n/m/P/C_{max} Flowshop Problem, University of Bremen.
- [40] Tegopoulos J.A. & Krienzis E.E., “*Eddy Currents in Linear Conducting Media*”, ed. Elsevier, 1985.
- [41] Vanderplaats G.N., “*Numerical Optimization Techniques for Engineering Design*”, ed. McGraw-Hill Company, 1984.
- [42] Vasconcelos J.A., Krähenbühl L., Nicolas L. & Nicolas A., “Design Optimization Using the BEM Coupled with Genetic Algorithm”, IEE Proceedings of Second International Conference on Computation in Electromagnetics, pp. 60-63, april, 1994.
- [43] Vasconcelos J.A., “Optimisation de Forme des Structures Électromagnétiques”, Tese de Doutorado, Ecole Centrale de Lyon, Ecully, França, 1994.
- [44] Vasconcelos J.A., Saldanha R.R., Krähenbühl L. & Nicolas A., “Algoritmo Genético Aplicado à Otimização em Eletromagnetismo”, Primeiro Congresso Brasileiro de Eletromagnetismo, Florianópolis, pp. 1-6, maio, 1995.
- [45] Vasconcelos J.A., Saldanha R.R., Krähenbühl L. & Nicolas A., “Genetic Algorithm Coupled with a Deterministic Method for Optimization in Electromagnetics”, IEEE Transactions on Magnetics, vol.32, nº 2, pp. 1860-1863, march, 1997.
- [46] Vasconcelos J.A., Saldanha R.R., Krähenbühl L. & Nicolas A., “Genetic Algorithm Coupled with Deterministic Method for Optimization in Electromagnetics”, IEEE CEFC’ 96, Okayama, pp. 18-20, march, 1996.
- [47] Whitley D., “GENITOR: A Different Genetic Algorithm”, Proceedings of the Rocky Mountain Conference on Artificial Intelligence, Denver, 1988.
- [48] Yao L., Sethares W.A. & Kammer D.C., “Sensor Placement for On-Orbit Modal Identification via Genetic Algorithm”, AIAA Journal, vol.31, nº 10, october, 1993.