MINISTÉRIO DA CIÊNCIA E TECNOLOGIA
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

INPE-7513-TDI/726

O USO DE ALGORITMOS GENÉTICOS NA DECOMPOSIÇÃO MORFOLÓGICA DE OPERADORES INVARIANTES EM TRANSLAÇÃO APLICADOS A IMAGENS DIGITAIS

João Ricardo de Freitas Oliveira

Tese de Doutorado em Computação Aplicada, orientada pelo Dr. Gerald Jean Francis Banon, aprovada em 17 de dezembro de 1998.

INPE
São José dos Campos
2000

621.376.5

OLIVEIRA, J. R. F.
O uso de algoritmos genéticos na decomposição morfológica de operadores invariantes em translação aplicados a imagens digitais / J. R. F. Oliveira. – São José dos Campos: INPE, 1998.
109p. – (INPE-7513-TDI/726).

1.Morfologia matemática. 2.Algoritmos genéticos. 3.Decomposição de Operador. 4.Filtragem. 5.Intervalos fechados. 6.Treinamento. I.Título.

Aprovado pela Banca Examinadora em cumprimento a requisito exigido para a obtenção do Título de **Doutor** em **Computação Aplicada.**

Dr. Solon Venâncio de Carvalho — Presidente

Dr. Gerald Jean Francis Banon — Orientador

Drª Sandra Aparecida Sandri — Membro da Banca

Dr. José Antônio Gonçalves Pereira — Membro da Banca

Dr. Roberto Vieira da Fonseca Lopes — Membro da Banca

Dr. Junior Barrera — Membro da Banca Convidado

Dr. Roberto de Alencar Lotufo — Membro da Banca Convidado

Candidato (a): João Ricardo de Freitas Oliveira

São José dos Campos, 17 de dezembro de 1998.

Tout problème profane un mystère; à son tour, le problème est profané par sa solution. Emile-Michel Cioran 1911- (Syllogismes de l'amertume).

Il n'y a pas de problème; il n'y a que des solutions. L'esprit de l'homme invente ensuite le problème. André Gide 1869-1951 (Journal).

Les pires choses en général sont faites des meilleures qui ont mal tourné. Les diables sont faits d'anges. Victor Hugo 1802-1885 (Fragments).

J'embrasse mon rival, mais c'est pour l'étouffer. Jean Racine 1639-1699 (Britannicus, IV, 3, Néron).

Si tout ici-bas était excellent, il n'y aurait rien d'excellent. Denis Diderot 1713-1784 (Le Neveu de Rameau).

Este trabalho é dedicado aos meus pais

João Damásio e Maria Aracy,

aos meus filhos

Paula,

Sandra,

Flávio,

e à minha mulher

Bete.

AGRADECIMENTOS

Eu gostaria de agradecer ao principal responsável pela concepção deste trabalho, meu orientador Dr. Gerald Jean Francis Banon, pela atenção concedida em todas as vezes que o procurei para discutirmos sobre assuntos relativos a este trabalho, assim como pelas idéias, sugestões, revisões e ensinamentos colocados à minha disposição. Quero agradecê-lo, ainda, pela confiança, pela paciência, e pelo profissionalismo sempre demonstrados.

Agradeço aos professores da pós-graduação da Computação Aplicada do INPE, em especial ao Dr. Tatuó Nakanishi, por seu apoio e incentivo constantes.

Agradeço ao Dr. Junior Barrera (USP), ao Dr. Roberto Vieira da Fonseca Lopes, à Dra. Sandra Aparecida Sandri, ao Dr. Solon Venâncio de Carvalho, e ao Dr. José Antônio Gonçalves Pereira pelas valiosas sugestões, idéias e revisões, sempre no intento de melhorar o conteúdo e a forma de apresentação deste trabalho.

Agradeço ao Dr. Roberto de Alencar Lotufo (UNICAMP) pelas idéias, sugestões e apoio irrestrito, e pela disponibilidade de uso da excelente biblioteca de rotinas de computador MMach 1.4 de sua autoria, que permitiu que a implementação computacional deste trabalho fosse simplificada através do uso de funções robustas de Morfologia Matemática e de manipulação de imagens.

Agradeço ao colega Sérgio Donizete Faria pelo apoio, principalmente no uso do KHOROS e no fornecimento da imagem binária utilizada neste trabalho. Agradeço ao colega Ronei Marcos de Moraes pela instalação e bom funcionamento do KHOROS na DPI/INPE.

Agradeço ao apoio dos companheiros de trabalho e da direção da Divisão de Processamento de Imagens do INPE (DPI-INPE), nas pessoas do Eng. Ri-

cardo Cartaxo Modesto de Souza, Dr. Gilberto Câmara Neto, e Eng. Ubirajara Moura de Freitas.

Agradeço à minha amiga Dra. Ana Lúcia Bezerra Candeias e aos meus amigos Dr. Hélio Koiti Kuga, Dr. Celso Luiz Mendes, Sérgio Rosim, Leonardo Sant'anna Bins, Eduardo Celso Gerbi Camargo e Mário Lopes Crossetti, pela amizade e pelo apoio incondicional e irrestrito recebido ao longo de vários anos.

Agradeço aos colegas do curso de pós-graduação em Computação Aplicada do INPE, Lilia, Liliane, Míriam, Mônica e Ronei pelo apoio e pelos bons momentos passados juntos.

Agradeço aos meus pais pela vida digna que me deram. Agradeço ao meu irmão pelo exemplo e ensinamentos.

Agradeço à minha mulher e aos meus três filhos pelo amor, pela compreensão, e por darem à minha vida um enorme sentido.

RESUMO

Este trabalho usa um algoritmo genético na decomposição de operadores invariantes em translação para imagens digitais. Esta decomposição tem a forma de uma união (resp. interseção) de operadores sup-geradores (resp. inf-geradores). Esses operadores são construídos a partir dos quatro operadores elementares da Morfologia Matemática que, por sua vez, são definidos por elementos estruturantes. A tarefa mais difícil nesta formulação é encontrar os elementos estruturantes que implementam a decomposição desejada. Para este propósito, um algoritmo genético é usado para encontrar uma solução sub-ótima em um espaço de soluções enorme. Vários testes computacionais são realizados e os resultados são suficientemente bons para justificar mais pesquisas nesta direção.

THE USE OF GENETIC ALGORITHMS IN THE MORPHOLOGICAL DECOMPOSITION OF TRANSLATION INVARIANT OPERATORS APPLIED TO DIGITAL IMAGES

ABSTRACT

This work uses a genetic algorithm for the morphological decomposition of translation invariant operators on digital images. This decomposition has the form of a union (resp. intersection) of sup-generating (resp. inf-generating) operators. These operators are built from the four morphological elementary operators which, in turn, are defined by structuring elements. The harder task in using this formulation is to find the structuring elements that ultimately implement the desired decomposition. For this purpose, a genetic algorithm is used to find a sub-optimal solution in a huge solution space. Several computational tests are done and the results are good enough to justify further research in this direction.

SUMÁRIO

Pág.

LISTA DE FIGURAS 15
LISTA DE SÍMBOLOS 19
CAPÍTULO 1 - INTRODUÇÃO 23
CAPÍTULO 2 - OS ALGORITMOS GENÉTICOS 29
2.1 - Características e Aplicações 29
2.2 - Descrição do Método 33
CAPÍTULO 3 - DECOMPOSIÇÃO DE OPERADORES EM MORFOLOGIA MATEMÁTICA 41
3.1 - Morfologia Matemática e Análise de Imagens 41
3.2 - Reticulado das imagens binárias 42
3.3 - Operadores sobre subconjuntos 43
3.4 - Operadores elementares 45
3.5 - Operadores invariantes em translação e de janela 45
3.6 - Construção dos operadores elementares invariantes em translação 47
3.7 - Operadores sup-gerador e inf-gerador 48
3.8 - Decomposição dos operadores invariantes em translação 50
CAPÍTULO 4 - IMPLEMENTAÇÃO EM COMPUTADOR E RESULTADOS 53
4.1 - Codificação em cadeias binárias 53
4.2 - Função de adaptação 55
4.3 - Parâmetros adotados 55
4.4 - Programação em computador 56
4.5 - Decomposição da dilatação 58
4.6 - Decomposição do extrator de bordas 65
4.7 - Decomposição dos filtros de ruído sal-e-pimenta e de ruído gaussiano 67
4.8 - Decomposição de fechamentos e aberturas morfológicos i.t. 72

4.9 - Aplicação usando imagens em níveis de cinza 79
4.10 - Limitações no uso do método 87
CAPÍTULO 5 - COMENTÁRIOS FINAIS 91
REFERÊNCIAS BIBLIOGRÁFICAS 93

LISTA DE FIGURAS

Pág.

2.1 - Comparação qualitativa de métodos diferentes.....29

2.2 - Diagrama de fluxo de um algoritmo genético simples.....35

2.3 - Representação equivalente da Função de Adaptação.....36

2.4 - Roleta de seleção ponderada pela adaptação. 37

2.5 - Operação de cruzamento. 37

2.6 - Operação de mutação. 38

3.1 - Representações equivalentes do operador sup-gerador de parâmetros A e B. 50

3.2 - Representações equivalentes de um operador i.t.....52

4.1 - Realização do operador $\hat{\psi}$ 57

4.2 - Decomposição da dilatação: **a**) X', **b**) Y', **c**) X, **d**) Y, **e**) $\hat{Y}$.....58

4.3 - Decomposição da dilatação (**a** e **b** ampliadas 1,8x): **a**) Y, **b**) $\hat{Y}$, **c**) diferença simétrica entre Y e $\hat{Y}$. 59

4.4 - Imagens resultantes da aplicação dos 7 sup-geradores em X, para a dilatação.....60

4.5 - Intervalos fechados que parametrizam os 7 sup-geradores para a dilatação.....60

4.6 - Evolução da adaptação ao longo das gerações para a decomposição da dilatação: melhor indivíduo e média da população. 62

4.7 - Dispersão da adaptação na população inicial.....63

4.8 - Dispersão da adaptação na população da 10^{a} geração. 63

4.9 - Dispersão da adaptação na população da 30^{a} geração. 64

4.10 - Dispersão da adaptação na população da 50^{a} geração. 64

4.11 - Extrator de borda. 65

4.12 - Decomposição do extrator de bordas: **a**) X , **b**) Y^c , **c**) $\hat{Y}^c$ 65

4.13 - Decomposição do extrator de bordas (ampliadas 2x): **a**) Y^c, **b**) $\hat{Y}^c$ 66

4.14 - Imagens resultantes da aplicação dos 5 sup-geradores em X, para o extrator de bordas. 66

4.15 - Intervalos fechados que parametrizam os 5 sup-geradores para o extrator de bordas. 66

4.16 - Evolução da adaptação ao longo das gerações para a decomposição do extrator de bordas: melhor indivíduo e média da população. 67

4.17 - Decomposição do filtro de ruído sal-e-pimenta: **a**) X, **b**) Y, **c**) $\hat{Y}$ 68

4.18 - Decomposição do filtro de ruído sal-e-pimenta (ampliadas 2x): **a**) X, **b**) Y, **c**) $\hat{Y}$ 68

4.19 - Imagens resultantes da aplicação dos 6 sup-geradores em X, para o filtro de ruído sal-e-pimenta. 69

4.20 - Intervalos fechados que parametrizam os 6 sup-geradores para o filtro de ruído sal-e-pimenta. 69

4.21 - Decomposição do filtro de ruído gaussiano: **a**) X, **b**) Y, **c**) $\hat{Y}$ 70

4.22 - Imagens resultantes da aplicação dos 7 sup-geradores em X, para o filtro de ruído gaussiano. 70

4.23 - Intervalos fechados que parametrizam os 7 sup-geradores para o filtro de ruído gaussiano. 71

4.24 - Decomposição do filtro de ruído sal: **a**) X, **b**) Y, **c**) $\hat{Y}$ 71

4.25 - Intervalos fechados que parametrizam os 6 sup-geradores para o filtro de ruído sal 71

4.26 - Decomposição do filtro de ruído pimenta: **a**) X, **b**) Y, **c**) $\hat{Y}$ 72

4.27 - Intervalos fechados que parametrizam os 6 sup-geradores para o filtro de ruído pimenta. 72

4.28 - Imagens de amostra para a decomposição do fechamento: **a**) X', **b**) Y' 73

4.29 - Imagens resultantes da decomposição do fechamento morfológico i.t. 73

4.30 - Intervalos fechados que parametrizam os 4 sup-geradores do fechamento 74

4.31 - Imagens de amostra para a decomposição da abertura: **a**) X', **b**) Y' 74

4.32 - Imagens resultantes da decomposição da abertura morfológica i.t. 74

4.33 - Intervalos fechados que parametrizam os 4 sup-geradores da abertura 75

4.34 - Estratégia de cruzamento por módulo de sup-gerador 75

4.35 - Evolução da adaptação para a decomposição da abertura com população de 100 indivíduos: cruzamento normal e por módulo 76

4.36 - Evolução da adaptação para a decomposição da abertura com população de 200 indivíduos: cruzamento normal e por módulo 76

4.37 - Evolução da adaptação para a decomposição da abertura com população de 300 indivíduos: cruzamento normal e por módulo 77

4.38 - Variação do valor de convergência em função do tamanho da população: cruzamento normal e cruzamento por módulo 78

4.39 - Variação do valor máximo obtido em função do tamanho da população: cruzamento normal e cruzamento por módulo 78

4.40 - Variação do tempo de convergência em função do tamanho da população: cruzamento normal e cruzamento por módulo 79

4.41 - Decomposição do filtro de ruído sal-e-pimenta para imagens em níveis de cinza: **a**) X, **b**) Y, **c**) $\hat{Y}$. 80

4.42 - Imagem SPOT original: cidade de Manaus 81

4.43 - Imagem borrada por um filtro de média 82

4.44 - Imagens de amostra e resultado da decomposição do filtro de média: caso 1 83

4.45 - Imagens de amostra e resultado da decomposição do filtro de média: caso 2 84

4.46 - Imagens de amostra e resultado da decomposição do filtro de restauração da nitidez 85

4.47 - Imagens de amostra e resultado da decomposição da dilatação 86

4.48 - Imagens de amostra e resultado da decomposição do extrator de bordas 87

4.49 - Decomposição do operador SKIZ: **a**) X, **b**) Y, **c**) $\hat{Y}$ 88

4.50 - Intervalos fechados que parametrizam os 4 sup-geradores para o operador SKIZ 88

4.51 - Evolução da adaptação ao longo das gerações para a decomposição do operador SKIZ: melhor indivíduo e média da população. 89

LISTA DE SÍMBOLOS

A, B, C	designação genérica de subconjuntos de E; imagens digitais.
adapt	função de adaptação.
c	coeficiente de normalização.
E	conjunto das posições de todos os pixels que compõem uma imagem binária sobre E.
$\mathcal{K}(\psi)$	núcleo de um operador pertencente a $\mathbf{\Psi}^{\mathbf{it}}$.
o	origem de E.
obj	função objetivo.
$\mathcal{P}(E)$	coleção de todos os subconjuntos de E.
r	expoente de realce.
W	subconjunto de E tomado como janela.
X, Y	designação genérica de subconjuntos de E; imagens digitais.
X', Y'	imagens digitais de amostra.
X^c	complemento do subconjunto X.
X^t	transposto do subconjunto X.
X_u	translado do subconjunto X por u.
$\#X$	número de elementos do subconjunto X.
$\mathcal{X}$	subcoleção não-vazia de $\mathcal{P}(E)$.

x	ponto pertencente a E; posição de um pixel.
$\hat{Y}$	aproximação do subconjunto Y.
$\mathbf{D}$	conjunto das dilatações sobre $\mathcal{P}(E)$.
d	dilatação sobre $\mathcal{P}(E)$.
δ_B	dilatação pelo subconjunto B.
$\mathbf{D}^{\mathrm{a}}$	conjunto das anti-dilatações sobre $\mathcal{P}(E)$.
d^{a}	anti-dilatação sobre $\mathcal{P}(E)$.
δ_B^{a}	anti-dilatação pelo subconjunto B.
$\mathbf{E}$	conjunto das erosões sobre $\mathcal{P}(E)$.
e	erosão sobre $\mathcal{P}(E)$.
ε_B	erosão pelo subconjunto B.
$\mathbf{E}^{\mathrm{a}}$	conjunto das anti-erosões sobre $\mathcal{P}(E)$.
e^{a}	anti-erosão sobre $\mathcal{P}(E)$.
$\varepsilon_B^{\mathrm{a}}$	anti-erosão pelo subconjunto B.
γ_B	abertura morfológica pelo subconjunto B.
ϕ_B	fechamento morfológico pelo subconjunto B.
$l_{A,B}$	operador sup-gerador sobre $\mathcal{P}(E)$ de parâmetros A e B.
$m_{A,B}$	operador inf-gerador sobre $\mathcal{P}(E)$ de parâmetros A e B.
$\mathbf{Q}$	subconjunto de $\mathbf{Y}$.

q	operador sobre $\mathcal{P}$, pertencente a $\mathbf{Q}$.
$\mathbf{Y}$	conjunto de todos os operadores sobre $\mathcal{P}(E)$.
$\mathbf{Y}^{\mathbf{it}}$	subconjunto de $\mathbf{Y}$, correspondente aos mapeamentos invariantes em translação.
y	operador sobre $\mathcal{P}(E)$.
ψ^*	mapeamento dual do operador y.
$\hat{\psi}$	aproximação do operador y.
$\psi(\mathcal{X})$	imagem de $\mathcal{X}$ através de y.
$\bigvee\mathcal{X}$	supremo de $\mathcal{X}$ em $\mathcal{P}(E)$.
$\bigwedge\mathcal{X}$	ínfimo de $\mathcal{X}$ em $\mathcal{P}(E)$.
[A, B]	intervalo fechado; coleção de todos os subconjuntos que contêm A e estão contidos em B.
Å	adição de Minkowski.
§	subtração de Minkowski.
Æ	conjunto vazio.

CAPÍTULO 1

INTRODUÇÃO

Desde os seus primórdios, o homem se ocupou em resolver problemas: fossem de natureza de sobrevivência básica (casa, comida), de defesa contra inimigos e feras, ou simples curiosidade, havia sempre um problema à espera de uma solução. Essa qualidade de resolver problemas revelou-se um fator relevante na evolução e desenvolvimento da inteligência humana.

Tal habilidade proporcionou ao homem uma melhoria constante em sua qualidade de vida, motivando-o mais e mais a enfrentar novos desafios, estando, para isto, melhor capacitado a cada vez (aprendizagem). Esta realimentação, agindo de forma construtiva ao longo do tempo, levou o homem, em última instância, a galgar os degraus do conhecimento humano até o estágio atual de desenvolvimento tecnológico, cultural e social.

Esta qualidade inerente ao ser humano de resolver problemas continua sendo extremamente importante na caracterização do homem moderno. A área de engenharia, por exemplo, tem por atribuição primordial a colocação e a solução de problemas. Dentro deste contexto, problemas do tipo "Como construir uma ponte?" são muito mais relevantes do que problemas do tipo "Como reconstruir uma ponte que caiu?". Enquanto que o primeiro tipo configura uma situação inédita, propícia para a criação e desenvolvimento de novas técnicas, no segundo tipo a principal atividade para a solução é a obtenção de recursos (considerando-se que um aprendizado anterior já tenha ocorrido).

Através da aprendizagem o homem pôde acumular um grande número de técnicas e metodologias usadas na resolução de diversos tipos de problemas. Há problemas que, uma vez resolvidos, não impõem novos desafios (Qual a fórmula para encontrar as raízes de uma equação do segundo grau?). Existem outros problemas, entretanto, que, a cada vez, surgem com nova roupagem, exigindo o uso de técnicas engenhosas, não raro forçando a experimentação de diferentes paradigmas e algoritmos na busca de uma solução aceitável dentro de um tempo útil.

Existe uma classe de problemas que é caracterizada por possuir um elevado número de soluções possíveis (ou, equivalentemente, pontos no espaço de soluções). Esta classe de problemas, que inclui os problemas multi-modais (presença de vários máximos locais), é muito comum, e conseqüentemente muito importante, estando presente nas mais diversas áreas do conhecimento humano, incluindo-se aí as atividades do dia-a-dia. É este tipo de problema que deve ser resolvido quando se joga xadrez, se dirige um carro na cidade, ou se planeja as férias de verão.

Problemas como a alocação e o uso de recursos limitados, como os que aparecem em companhias transportadoras ou de transporte coletivo, também pertencem a esta classe de problemas. Nesse caso, como em vários outros, o uso da melhor solução pode representar ganhos significativos no desempenho econômico das empresas.

Nas áreas científica e de engenharia, problemas com um número gigantesco de soluções candidatas aparecem com freqüência associados ao surgimento de novas tecnologias ou de novas metodologias. Com o advento do computador, soluções para esses problemas puderam ser obtidas através da busca exaustiva (ou força-bruta), ainda que apenas para os casos mais simples.

Mesmo com o avanço tecnológico dos computadores atuais, existem e sempre existirão os problemas em que a solução ótima não pode ser obtida pela busca exaustiva dentro de um intervalo de tempo útil. Um computador que jogue xadrez não seria muito útil se levasse, por exemplo, 1 ano para fazer uma jogada. Por outro lado, se a cura do câncer houvesse demorado 10 anos para ser obtida, isso seria um tempo curto o suficiente para ser considerado útil. Assim, dependendo do problema em questão, o significado de "tempo útil" pode variar de frações de segundo a dezenas de anos.

Muitas vezes, dependendo das características do problema abordado, é suficiente encontrar uma solução sub-ótima, desde que seja respeitado o tempo hábil. Neste caso, vale a máxima popular que diz que "o bom é inimigo do ótimo". Quantas vezes, no cotidiano, tomam-se decisões que, embora se saiba não serem as ótimas, são as melhores disponíveis dentro do intervalo de tempo que se tem para tomar estas decisões?

Desse modo, vários métodos e heurísticas foram desenvolvidos e testados ao longo do tempo, com maior ou menor êxito, dependendo da natureza do problema (domínio do problema). Em 1975 John Holland e seus colaboradores na Universidade de Michigan desenvolveram um método de busca inspirado nos processos de adaptações de sistemas naturais (Holland, 1975), chamado de "algoritmo genético" (abreviadamente, GA, do termo inglês *Genetic Algorithm*).

Em um primeiro momento, a simplicidade deste algoritmo obliterou o seu potencial de uso efetivo. Some-se a isto um certo excesso de zelo daqueles que consideravam esta proposta de se usar mecanismos naturais em procedimentos de busca como um caminho pouco promissor. Mas não tardou para que os primeiros frutos começassem a aparecer. Vários trabalhos e teses foram desenvolvidos usando o paradigma genético com excelentes resultados práticos, firmando o seu uso como uma ferramenta poderosa, tão boa ou até melhor que os métodos conhecidos e utilizados até aquele momento.

É interessante notar que, em relação ao ser humano, o sentido da visão sempre foi o canal mais importante na tarefa de suprir o cérebro com informações (entrada) do mundo exterior, tanto para decidir quais ações executar como para acompanhar a boa execução destas ações (realimentação pela visão, que nos permite, por exemplo, manter um carro em sua faixa de pista em uma estrada sinuosa).

Era de se supor, em conseqüência, que tão logo estivessem disponíveis máquinas (computadores) com capacidades de processamento e armazenamento adequadas, propiciando a implementação prática de algoritmos, a área de Processamento de Imagens (PI) experimentaria um grande desenvolvimento teórico e prático, com aplicações em um vasto espectro de atividades e conhecimentos do ser humano. De fato, a partir de meados da década de 60, a área de PI (realce de imagens, reconhecimentos de padrões, e análise de imagens e visão por computador) apresentou um forte crescimento, tendo sido objeto de estudos interdisciplinares em áreas como fisiologia, matemática, física, engenharia eletrônica, ciência da informação, e ciência da computação. Esses esforços levaram ao estabelecimento de um grande número de técnicas e algoritmos, aplicáveis a uma enorme variedade de problemas, auxiliando de forma inequívoca o

desenvolvimento de áreas como: medicina, odontologia, biologia, geologia, ciência dos materiais, meteorologia, astronomia, manufatura e engenharia de produção, robótica, física, química, macro-economia, direito, arquitetura, artes, arqueologia, segurança, telecomunicações digitais, sistemas sensores inteligentes, e sensoriamento remoto.

Estes estudos basearam-se em duas linhas principais: o processamento linear clássico, e a morfologia matemática. No primeiro caso são utilizadas a convolução e a representação dos sistemas no domínio da freqüência como ferramentas básicas na busca de soluções como, por exemplo, eliminar distorções e reconstruir a função (sinal) original. Como qualquer função do tempo pode ser representada por uma soma de senóides, uma função corrompida por ruídos e distorções pode ser estudada no domínio da freqüência de modo a permitir que sejam desenvolvidos métodos para eliminar de forma ótima o efeito destas distorções.

Em relação à Morfologia Matemática (MM), o conceito de *forma* tem papel fundamental no estudo de algoritmos para processamento de imagens, sendo a forma uma portadora primordial de informações. Um sistema de operações morfológicas, como a MM, é poderoso porque seus operadores mais simples podem ser combinados formando operadores mais complexos que, ao atuarem em dados de imagens, são capazes de decompor formas complexas em suas partes essenciais, separando-as de suas partes ruidosas.

Na grande maioria dos casos, a boa aplicação da MM depende da escolha de um ou mais elementos estruturantes adequados. Essa seleção de elementos estruturantes feita manualmente por seres humanos não é prática na maioria das situações, requerendo um trabalho braçal grande mesmo para elementos estruturantes pequenos, exceção feita aos casos mais simples. Um dos temas atuais de grande interesse na aplicação da MM em PI reside no projeto automático de elementos estruturantes.

Neste trabalho será usado um GA para se encontrar elementos estruturantes que irão compor um filtro morfológico de imagens, de propósito geral, utilizando-se uma expressão canônica deduzida através de ferramentas teóricas da MM. Acredita-se que o méto-

do apresentado neste trabalho tenha características inovadoras, tendo sido desenvolvido em decorrência de um trabalho inicial (Oliveira, 1996) que indicou a possibilidade de um bom desempenho da metodologia aqui adotada

É interessante ressaltar a extensa aplicabilidade do método delineado neste trabalho uma vez que: a) PI é ferramenta importante de inúmeras áreas; b) a expressão morfológica canônica utilizada pode decompor qualquer operador i.t.; e c) os algoritmos genéticos (GAs) podem ser ajustados para apresentar um bom desempenho em, virtualmente, todos os problemas combinatorialmente complexos.

Outros trabalhos publicados correlacionados a este são descritos a seguir: Dougherty e Haralick (1991) utilizam um método de estimação estatística que emprega um procedimento espectral baseado em formas para restauração de imagens; Dougherty et al. (1991) apresentam um algoritmo que deriva um filtro morfológico ótimo que minimiza o erro quadrático médio a partir da esperança condicional; Dougherty e Loce (1992) desenvolveram um trabalho que trata vínculos e utiliza um biblioteca de elementos estruturantes com objetivo de acelerar a obtenção de soluções; os trabalhos de Barrera et al. (1995) e Kim (1997) utilizam a teoria de aprendizagem computacional PAC (Provavelmente Aproximadamente Correta) para o projeto automático de operadores morfológicos; Harvey e Marshall (1994) utilizam um GA para projetos de filtros morfológicos usando encadeamento (composição de operadores) de erosões e dilatações.

No Capítulo 2 deste trabalho são apresentadas as características gerais de um GA e algumas aplicações. Em seguida é feita uma descrição detalhada de um GA específico. O Capítulo 3 apresenta o embasamento necessário de MM para apresentar a forma canônica de decomposição de operadores i.t. A forma como o problema foi codificado para uso de um GA, bem como os parâmetros utilizados e os resultados obtidos estão apresentados no Capítulo 4. Por fim, no Capítulo 5 são feitos alguns comentários finais.

CAPÍTULO 2

OS ALGORITMOS GENÉTICOS

2.1 - Características e Aplicações

Um algoritmo genético (GA) é uma técnica de busca multi-agente, tradicionalmente usado como ferramenta pela comunidade de inteligência artificial. Um GA representa uma promissora classe de processos de busca com grande robustez e paralelismo intrínseco, adequado à resolução de um largo espectro de problemas de otimização e aprendizagem por máquinas. Sua simplicidade de enfoque orientada a combinações de variáveis discretas tornam os GAs atrativos em comparação a outros métodos matemáticos mais complexos.

Um método especializado pode ter um desempenho (em termos de eficiência) superior que um GA em um problema específico, por utilizar informações e heurísticas inerentes a esse problema. Um GA, por outro lado, apresenta um desempenho consistentemente robusto ao longo de um largo espectro de domínios de problemas. Os métodos de enumeração (ou busca exaustiva) apresentam sempre um baixo desempenho. A Figura 2.1 ilustra essas considerações de forma qualitativa.

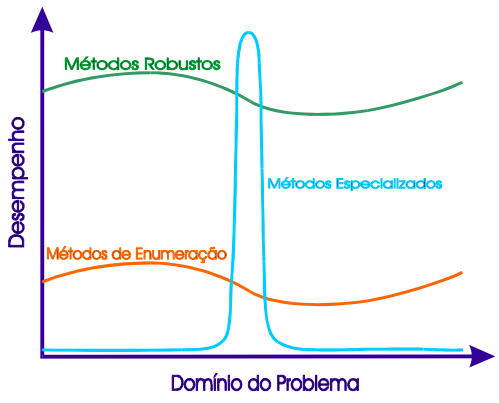


Figura 2.1 - Comparação qualitativa de métodos diferentes.

Um GA apresenta um comportamento versátil porque depende pouco do domínio do problema. Há na literatura específica muitos exemplos que ilustram a sua utilização em uma grande variedade de problemas, nas mais diversas áreas do conhecimento humano. Como ilustração dessa diversidade pode-se citar o balanceamento de carga em computadores paralelos (Baumgartner e Cook, 1994), a extração de petróleo (Martinez et al., 1993; Martinez et al., 1994; Fang et al., 1993), o reconhecimento de moléculas (Payne e Glen, 1993), a busca em banco de dados (Cui et al., 1993; Horng, 1994), o controle de tráfego aéreo (Alliot, 1993), o controle de mísseis (Hull e Johnson, 1994), e a composição de músicas (Mc Intyre, 1994). Outra característica vantajosa dos GAs é a robustez, atestada por excelentes desempenhos, na presença de ruídos (Huang e Bian, 1993).

Outros fatores decisivos no crescente uso do método são a sua eficácia e a sua eficiência. É eficaz pois geralmente encontra em tempo hábil a solução ótima-global, ou uma solução próxima da ótima-global, de problemas combinatórios complexos. Normalmente oferece baixas taxas de erro, e um GA já encontrou a melhor solução conhecida até aqui de alguns problemas para teste de desempenho (Muehlenbein et al., 1988; Muehlenbein et al., 1991; Thangiah et al., 1990). O que decorre disto é que os GAs apresentam um potencial significativo na produção de soluções para problemas difíceis. Um GA é eficiente pois normalmente apresenta rápida convergência para soluções próximas da ótima-global em espaços de busca mal comportados (Karr e Goldberg, 1990; Androulakis e Venkatasubramanian, 1991; Murdock et al., 1991).

O uso do método é indicado em problemas em que é necessário rapidez na obtenção de soluções. Um GA também pode ser aplicado com vantagens em sistemas que apresentem mudanças súbitas em sua dinâmica (separação do primeiro estágio de um foguete), em sistemas fortemente não-lineares, em sistemas instáveis, e em sistemas com atrasos intrínsecos no tempo (Hoptroff et al., 1990; Wieland, 1992) (por exemplo, comandos para sondas interplanetárias). Além disso, existem problemas que são beneficiados pela capacidade de um GA em produzir, dentro de certas condições, várias soluções próximas da ótima-global (Alliot et al., 1993; Bickel e Bickel, 1990; Le Riche

e Haftka, 1993); essas soluções estariam à disposição de um operador humano a quem caberia a decisão final de qual utilizar, a partir de parâmetros não modelados.

Vários resultados demonstrando a conveniência do uso da técnica podem ser encontrados na literatura especializada. Um GA pode ser empregado quando as técnicas tradicionais de Newton, enumeração, regressão não-linear, busca pelo gradiente, e busca binária se apresentam lentas, instáveis ou não confiáveis (Deb, 1990; Davidor, 1990; Thangiah e Nygard, 1992; Cartwright e Harris, 1993; Dandy et al., 1993; Clark et al., 1994; Mansour e Fox, 1994). Em muitos casos foi relatado o melhor desempenho de um GA em relação a algoritmos de alto desempenho de estimação de parâmetros sob condições difíceis (Sharman e Mc Clurkin, 1989). O método mais usualmente comparado em desempenho com um GA é o "simulated annealing", com favorecimento para o primeiro (Cohoon et al., 1988; Yang, 1993; Mutalik et al., 1992; Stuckman, 1991; Sen et al., 1992; Wen e Han, 1994; Sheung et al., 1993; Kwok e Sheng, 1994; Bhandarkar et al., 1994; Vemuri e Vemuri, 1994). Algumas vezes, quando se quer mais velocidade ou precisão na obtenção de soluções, algoritmos hibridizados têm sido utilizados, em geral com um GA em seu cerne (Huntley e Brown, 1991; Bellgard e Tsang, 1991; Lin et al., 1993; Esbensen e Mazumder, 1994; Hadi e Wallace, 1993; Schultz, 1994; Renders e Bersini, 1994; Patton e Liu, 1994).

Consideradas as características que tipificam o problema de projeto de redes neurais, a saber, espaços de busca de muitas dimensões com descontinuidades e ruídos (Harp e Samad, 1992), não é surpresa o grande número existente de trabalhos publicados em que se utiliza um GA no projeto de redes neurais. Um GA pode ser usado tanto para estimar os pesos de uma rede neural, como para determinar a melhor arquitetura para uma rede neural (Hintz e Spofford, 1990; Koza e Rice, 1992; Williams et al., 1994; Patel e Maniezzo, 1994; Ikuno et al., 1994). Esta combinação de técnicas possibilita a construção de redes neurais dinâmicas e hierárquicas, arcabouço do que é chamado de "sistema nervoso artificial" (de Garis, 1990; de Garis, 1991). Outras áreas de aplicação onde os GAs têm sido extensivamente utilizados são lógica nebulosa (Mohammadian e Stonier 1994a e 1994b; Fukuda et al., 1994; Bezdek e Hathaway, 1994; Kinzel et al.,

1994; Kim et al., 1994) e PI, cujos trabalhos são descritos a seguir: Mc Aulay e Oh (1989) examinam aspectos de aprendizagem computacional para um sistema classificador que usa um GA; Mandava et al. (1989) descrevem uma técnica para registro de imagens médicas, com considerações físicas que limitam o espaço de busca de um GA; Bala e Wechsler (1992) propõem uma maneira de combinar processamento morfológico e um GA de modo a gerar operadores de alto desempenho de discriminação de formas; Trenkle et al. (1991) investigam o uso de um GA na seleção otimizada de características de conjuntos com o fim de discriminar grandes conjuntos de caracteres; Chu e Kottapalli (1991) usam um GA para projetar matrizes de pontos dispersos para implementar meio-tons; Jacq e Roux (1993a) apresentam um método para registro automático de imagens médicas seqüenciais com uso de um GA; em Jacq e Roux (1993b) o mesmo método é apresentado para imagens 3D; Bhandari et al. (1993) usam uma função de adaptação nebulosa para realce de imagens; Dasgupta e Mc Gregor (1992) utilizam cromossomos com múltiplas camadas e mecanismos de ativação de genes com o fim de registro automático de imagens digitais; Seetharaman et al. (1992) utilizam operadores modificados de cruzamento e de mutação para uso em segmentação de imagens; Pal (1992) discute vários aspectos da aplicação da teoria de conjuntos nebulosos em PI, combinado com um GA e com redes neurais; Pal et al. (1994) demonstraram o uso de um GA na busca da solução global ótima na seleção automática de um operador para realce de imagem; Roth e Levine (1994) usaram um GA para a extração de primitivas geométricas a partir de dados de sensor de geometria para uso em visão por máquinas; Harvey e Marshall (1994) utilizam um GA para projetar filtros morfológicos otimizados através da composição de operadores morfológicos elementares; Oliveira (1996) mostra a viabilidade do uso eficiente de um GA em problemas de classificação de imagens.

Com a perspectiva de um crescente acesso às máquinas de processamento paralelo, os GAs se apresentam como uma ferramenta bem adequada a essas arquiteturas (Cui et al., 1993; Dorigo, 1992; Logar et al., 1992), tornando-se útil o uso de plataformas para o estudo da paralelização dos GAs (Idlebi e Mignot, 1994), bem como o desenvolvimento de um ambiente de programação paralela orientada a objetos (Kingdon e Dekker, 1994).

Com tantas qualidades interessantes, é previsível uma maior utilização do método nos anos que virão.

Os sistemas de programação usados para a implementação de um GA podem ser agrupados em três classes: sistemas orientados à aplicação (Kanet e Sridharan, 1991), elaborados para atender a uma aplicação específica; sistemas orientados aos algoritmos (Wilke, 1994), usados para o desenvolvimento de novas técnicas aplicáveis aos GAs; e sistemas de programação geral (Bayer e Wang, 1992), que podem ser utilizados por várias aplicações diferentes.

2.2 - Descrição do Método

Um algoritmo genético (GA) baseia-se nas teorias evolutivas para seres vivos, e portanto utiliza-se dos mecanismos de seleção natural e de recombinação de informação genética, onde os indivíduos de uma população que melhor se adaptarem ao meio ambiente têm maior probabilidade de sobreviver e gerar descendentes que retransmitirão suas heranças genéticas às novas gerações. A conseqüência global desse processo é a melhoria da média da adaptação das populações posteriores. Da mesma forma, é melhor a adaptação dos melhores indivíduos de uma população em relação aos melhores indivíduos das gerações anteriores.

Em termos computacionais, um GA mapeia (codifica) os parâmetros de um dado problema para um conjunto de caracteres, usualmente *cadeias* binárias, sendo que cada cadeia, também chamada de *cromossomo*, representa uma solução candidata. Os GAs então manipulam os cromossomos mais promissores em busca de soluções melhores.

O modelo mais simples de um GA compõe-se de um conjunto de indivíduos (a população, de tamanho fixo ao longo das gerações), e de um conjunto de operadores genéticos (a seleção, o cruzamento e a mutação). Será descrito a seguir, de forma mais detalhada, o GA utilizado neste trabalho.

O processo começa com a criação de uma população inicial (primeira geração), o que pode ser feito de modo aleatório. Em seguida à geração de uma população, inicial ou

subseqüente, é feita a avaliação da adaptação de todos os indivíduos que a compõem através de uma *função objetivo* que represente o critério de otimização desejado. Em seguida, o processo de seleção escolhe os pais que serão responsáveis pela criação da próxima geração. Essa escolha é feita de modo que a chance de cada indivíduo ser selecionado (com reposição) é diretamente proporcional à sua adaptação. Os casais assim selecionados passam pelo processo de cruzamento, ocasião em que ocorre a troca de informação genética, e pelo processo de mutação, responsável pela introdução de novas informações, gerando os filhos que irão compor a próxima geração. O processo termina quando se encontra uma solução "suficientemente boa", segundo um critério de parada escolhido (pode ser, por exemplo, um critério de precisão numérica, de tempo ou de número máximo de gerações).

O procedimento iterativo de um GA pode ser visto na Figura 2.2:

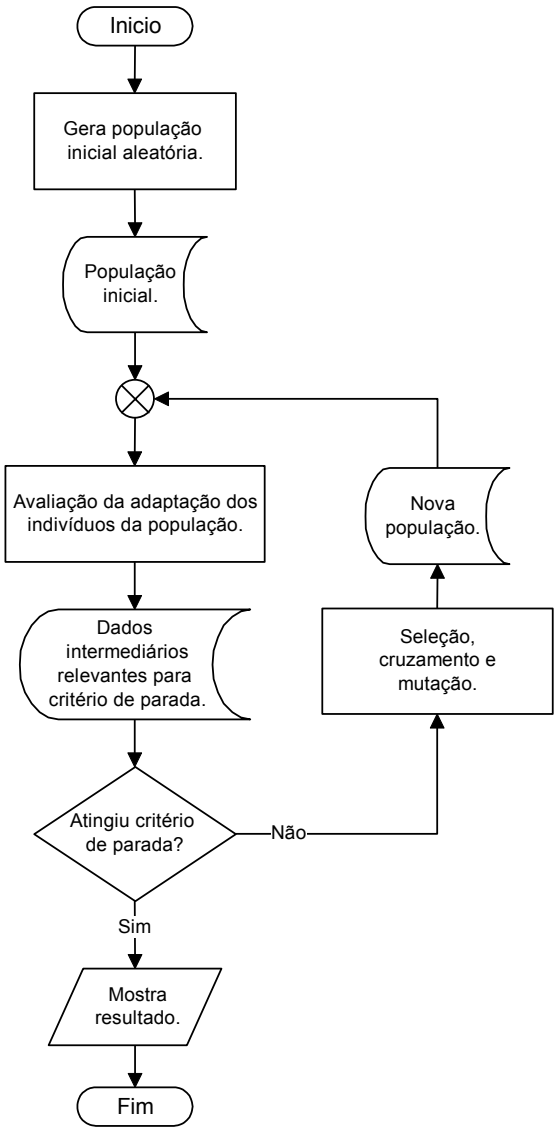


Figura 2.2 - Diagrama de fluxo de um algoritmo genético simples.

O bom desempenho de um GA depende fortemente da sua capacidade de avaliar corretamente a adaptação dos indivíduos de uma população. A característica de um GA de independer de parâmetros específicos do domínio do problema determina a qualidade de robustez do algoritmo. A única informação usada para dirigir o processo de busca é dada pela função de adaptação, e por isso o desempenho de um GA está condicionado à qualidade da avaliação da adaptação dos indivíduos.

A função de adaptação pode ser vista como uma caixa-preta onde a entrada é uma cadeia binária que codifica um indivíduo, e a saída é um valor correspondente à adaptação deste indivíduo, como ilustra a Figura 2.3.

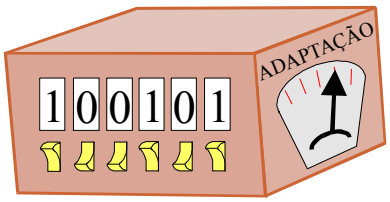


Figura 2.3 - Representação equivalente da Função de Adaptação.

Para realizar a avaliação da adaptação dos indivíduos de uma população, necessitamos inicialmente definir a função objetivo *obj* a ser otimizada. O domínio desta função corresponde ao conjunto das cadeias codificadas. Para assegurar uma uniformidade de tratamento nos parâmetros de ajuste para diversos domínios de problemas, a função de adaptação $adapt(cadeia)$ é usualmente obtida através da normalização da função objetivo $obj(cadeia)$ por um coeficiente *c*, seguida de uma exponenciação por um expoente *r* para ressaltar o valor relativo dos melhores indivíduos, ou seja:

$$adapt(cadeia) = \left(\frac{obj(cadeia)}{c}\right)^r$$

O processo de seleção usado neste trabalho escolhe ao acaso, com reposição, dois indivíduos que irão compor um casal. Cada indivíduo, porém, tem a probabilidade de ser escolhido proporcionalmente à sua adaptação: um indivíduo melhor adaptado tem mais chance de gerar descendentes do que um indivíduo menos adaptado.

Este processo de seleção pode ser representado por uma roleta com áreas desiguais para cada indivíduo, de modo a ponderar as chances pelo valor da adaptação, como ilustra a Figura 2.4.

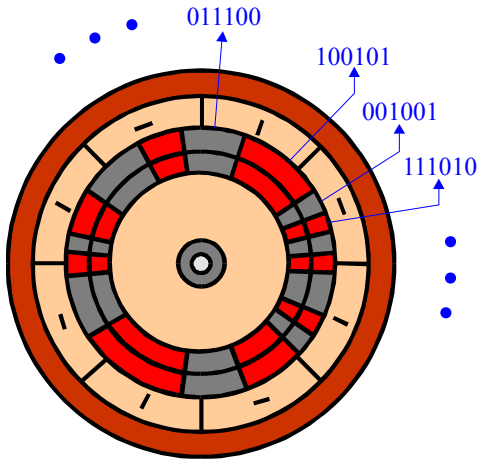


Figura 2.4 - Roleta de seleção ponderada pela adaptação.

Um casal de pais, P1 e P2, selecionados para o cruzamento, geram um casal de filhos, F1 e F2, que irão compor a próxima geração. A operação de cruzamento inicia-se com a escolha de um ponto de cruzamento, o que normalmente é feito de forma aleatória, embora métodos heurísticos possam ser utilizados aqui. Então, as partes secionadas dos cromossomos do casal de pais são intercambiadas gerando assim um casal de filhos, como ilustra a Figura 2.5.

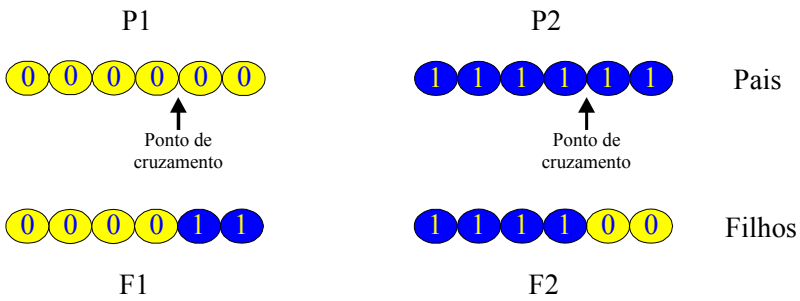


Figura 2.5 - Operação de cruzamento.

A operação de mutação é realizada para evitar a convergência prematura do algoritmo, introduzindo na busca novas regiões do espaço de soluções. Esta operação de mutação é feita com probabilidade baixa para cada elo da cadeia (bit), algo em torno de um para mil, para não descaracterizar as boas soluções encontradas até aquele momento. Quando um elo da cadeia é escolhido para sofrer a operação de mutação, o seu valor (binário) é complementado, de acordo com a Figura 2.6.

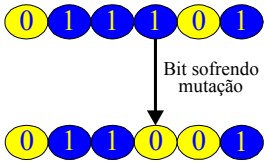


Figura 2.6 - Operação de mutação.

Adicionalmente às técnicas básicas para implementar um GA, várias outras possibilidades de melhoria do método têm sido estudadas. A técnica mais usada na tentativa de melhorar o desempenho de um GA é o aperfeiçoamento dos operadores ou a criação de novos operadores (Shahookar e Mazumder, 1990; Khuri, 1990; Moed et al., 1992; Shang e Li, 1992; Potter et al., 1991; Ansari et al., 1990; Yao, 1993; Lin e Hajela, 1993; Yang e Nygard, 1993; Konstam et al., 1992; Hartley e Konstam, 1993; Corcoran e Wainwright, 1992; Falkenauer e Delchambre, 1992; Bhandari et al., 1994; Park et al., 1993; Potts et al., 1994; Miller et al., 1993; Yamamura et al., 1994; Prahlada et al., 1994).

A estratégia de penalização aplicada à função objetivo para lidar com vínculos (restrições) é muitas vezes utilizada (Rajeev e Krishnamoorthy, 1992; Ball et al., 1993; Lee e Takagi, 1993; Homaifar et al., 1994; Joines e Houck, 1994; Baeck e Khuri, 1994; Oliveira, 1996), porém há casos de utilização de estratégia alternativa como a de gerar apenas indivíduos que satisfaçam os vínculos (Sun e Wang, 1994; Sakane et al., 1994), ou através do "conserto" de alguns cromossomos "ilegais" de uma população (Orvosh e Davis, 1994).

Em muitos casos pode ser conveniente alterar dinamicamente a probabilidade de cruzamento e de mutação dos indivíduos, ao longo das gerações (Srinivas e Patnaik, 1994).

Quanto à estrutura, os cromossomos podem ser de tamanho variável (Tanaka et al., 1994), bidimensionais (Bhandarkar et al., 1994), ou com múltiplas camadas (Dasgupta e Mc Gregor, 1992; Dasgupta e Mc Gregor, 1994). Todas essas variações buscam um processo eficaz de codificação dos parâmetros do problema.

Em relação à população há possibilidades de uso de população variável (Arabas et al., 1994), ou de geração dirigida (não-aleatória) da população inicial (Yang e Nygard, 1993); para ambientes com arquitetura paralela, é útil o uso de múltiplas populações (Elketroussi e Fan, 1994), ou de populações atualizadas indivíduo a indivíduo, com indivíduos de gerações diferentes pertencendo à mesma população (Stoffa e Sen, 1991; Husbands, 1994).

Existem estudos sobre a estimação do número de gerações necessárias para obter a convergência de um GA (Chakraborty e Dastidar, 1993), como também sobre a obtenção de uma função de adaptação eficaz a partir da função objetivo (Kreinovich et al., 1993). Há ainda a possibilidade de tentar seguir o exemplo biológico o mais próximo possível, o que é proposto em (Juric, 1994).

Por outro lado, há estudos abordando problemas em que falha o paradigma genético, conhecidos por "GA deceptive problems" (Vose, 1991; Forrest e Mitchell, 1993; Dasgupta, 1994). Uma maneira de tentar contornar estes problemas reside na escolha apropriada da forma que os parâmetros do problema serão codificados(Altenberg, 1994): a escolha da codificação apropriada pode proporcionar a convergência do método.

CAPÍTULO 3

DECOMPOSIÇÃO DE OPERADORES EM MORFOLOGIA MATEMÁTICA

3.1 - Morfologia Matemática e Análise de Imagens

A área de Análise de Imagens por computador foi objeto de grande desenvolvimento ao longo das últimas três décadas, justificado pelas inúmeras aplicações práticas que possibilitou nas mais diversas áreas, como Sensoriamento Remoto, Visão Robótica, e Imagens Médicas, para citar algumas. Uma das ferramentas mais poderosas usada em Análise de Imagens é a Morfologia Matemática (MM), desenvolvida a partir do trabalho pioneiro de George Matheron (1967) e Jean Serra (1982; 1988), e usada na extração de informações a partir das formas contidas em uma imagem. A MM possui sólida formalização matemática e permite abordar de modo unificado a Análise de Imagens.

No caso de invariância por translação, a Morfologia Matemática para imagens binárias fundamenta-se em operadores elementares parametrizados por "elementos estruturantes", que são tipicamente pequenos padrões (comparativamente ao tamanho da imagem) que agem como sondas na detecção e tratamento de formas em imagens. Um dos resultados significativos da MM é a decomposição dos operadores em termos dos operadores elementares da MM (Banon e Barrera, 1991, 1993). Esse resultado, embora poderoso em termos de possibilidades de aplicações, apresenta uma grande dificuldade de implementação: quais são os elementos estruturantes que caracterizam os operadores elementares que irão compor um dado operador? Em geral esta tarefa engloba a busca de uma solução quase-ótima em um espaço de soluções gigantesco, desautorizando qualquer tentativa de uso de métodos de enumeração. Por outro lado, o uso de um GA parece indicado para a obtenção de soluções para este problema. Este trabalho tenta mostrar a aplicabilidade de um GA na decomposição de operadores invariantes em translação.

Com este objetivo, os conceitos básicos da MM para a análise de imagens binárias serão relembrados. Os pixels podem assumir dois estados, "aceso" e "apagado", representados

pelos valores 1 e 0, respectivamente. Porém, como será visto mais adiante, os procedimentos utilizados aqui para as imagens binárias poderão ser estendidos para as imagens em níveis de cinza.

3.2 - Reticulado das imagens binárias

Seja E um conjunto finito, não-vazio, de pontos $x \in E$. Cada ponto x corresponde à posição de um pixel de uma imagem binária definida sobre E. Dessa forma, E representa também o conjunto de posições de todos os pixels que compõem uma imagem binária sobre E.

Um subconjunto X de E ($X \subset E$) é um modelo de uma imagem definida sobre E se e somente se (sse) X é o subconjunto das posições dos pixels acesos.

A coleção (conjunto de subconjuntos) de todos os subconjuntos de E será denotada por $\mathcal{P}(E)$, ou seja, $X \in E \Leftrightarrow X \in \mathcal{P}(E)$. Desse modo, $\mathcal{P}(E)$ é a coleção de todas as imagens que podem ser definidas sobre E.

A relação de inclusão de conjuntos, $\subset$, é uma relação de ordem parcial pois, para todo A, B e $C \in \mathcal{P}(E)$, é uma relação reflexiva ($A \subset A$), anti-simétrica ($A \subset B$ e $B \subset A \Rightarrow A = B$), e transitiva ($A \subset B$ e $B \subset C \Rightarrow A \subset C$).

A coleção $\mathcal{P}(E)$ dotada de uma relação de ordem parcial ($\subset$), forma um conjunto parcialmente ordenado, denotado por ($\mathcal{P}(E), \subset$).

Seja $\mathcal{X}$ uma subcoleção não-vazia de $\mathcal{P}(E)$, isto é, $\mathcal{X} \subset \mathcal{P}(E)$, e seja A um elemento de $\mathcal{P}(E)$, $A \in \mathcal{P}(E)$. Se $X \subset A$ para todo $X \in \mathcal{X}$, então A é um *limitante superior* de $\mathcal{X}$ em $\mathcal{P}(E)$. O *supremo* de $\mathcal{X}$ em $\mathcal{P}(E)$ é, se existir, o menor dos limitantes superiores de $\mathcal{X}$ em $\mathcal{P}(E)$, e será denotado por $\bigvee \mathcal{X}$. Isto significa que, para todo $X \in \mathcal{P}(E)$,

$$X \text{ é limitante superior de } \mathcal{X} \Leftrightarrow \bigvee \mathcal{X} \subset X.$$

Definem-se analogamente os conceitos de *limitante inferior* e *ínfimo* ($\bigwedge\mathscr{X}$), de forma que, para todo $X \in \mathscr{P}(E)$,

X é limitante inferior de $\mathscr{X} \Leftrightarrow X \subset \bigwedge\mathscr{X}$.

Convenientemente são definidos $\bigvee\varnothing = \varnothing$ e $\bigwedge\varnothing = E$, onde $\varnothing$ simboliza o conjunto vazio.

A partir da definição de supremo e ínfimo de $\mathscr{X} \subset \mathscr{P}(E)$ definem-se, respectivamente, as operações de *união* ¿ e *interseção* Ç entre conjuntos: $\forall A, B \in \mathscr{P}(E)$,

$A \bigcup B = \bigvee\mathscr{X}$ e

$A \bigcap B = \bigwedge\mathscr{X}$

onde

$\mathscr{X} = \{ A, B \}$ se A e B são distintos, e

$\mathscr{X} = \{ A \}$ se $A = B$.

O conjunto parcialmente ordenado ($\mathscr{P}(E), \subset$) é um *reticulado completo* pois toda sub-coleção $\mathscr{X}$ de $\mathscr{P}(E)$ possui um supremo e um ínfimo em $\mathscr{P}(E)$.

O maior e o menor elemento do reticulado completo ($\mathscr{P}(E), \subset$) são, respectivamente, o conjunto E e o conjunto vazio $\varnothing$.

3.3 - Operadores sobre subconjuntos

Um operador ψ sobre $\mathscr{P}(E)$ é um mapeamento de $\mathscr{P}(E)$ em $\mathscr{P}(E)$. O conjunto de todos os operadores sobre $\mathscr{P}(E)$ será denotado por Ψ, $\psi \in \Psi$ Û $\psi : \mathscr{P}(E)$ ® $\mathscr{P}(E)$. Se o subcon-

junto $X \in \mathcal{P}(E)$ é transformado em $Y \in \mathcal{P}(E)$ através da aplicação do operador ψ, escreve-se $Y = \psi(X)$.

Um operador ψ sobre $\mathcal{P}(E)$ é crescente sse, para todo $A, B \in \mathcal{P}(E)$,

$$A \subset B \Rightarrow \psi(A) \subset \psi(B),$$

e decrescente sse, para todo $A, B \in \mathcal{P}(E)$,

$$A \subset B \Rightarrow \psi(B) \subset \psi(A).$$

Será denotado por $\psi(\mathcal{X})$ a imagem de $\mathcal{X}$ ($\mathcal{X} \subset \mathcal{P}(E)$) através de ψ:

$$\psi(\mathcal{X}) = \{Y \in \mathcal{P}(E): \exists X \in \mathcal{X}, Y = \psi(X)\}.$$

Um operador ψ_1 é menor que um operador ψ_2 (denota-se por $\psi_1 \leq \psi_2$) sse $\psi_1(X) \subset \psi_2(X)$, para todo $X \in \mathcal{P}(E)$, ou de outro modo,

$$\psi_1 \leq \psi_2 \Leftrightarrow (\psi_1(X) \subset \psi_2(X) \ \ (X \in \mathcal{P}(E))).$$

Dessa forma, o conjunto $\mathbf{\Psi}$ herda a estrutura de reticulado completo de $(\mathcal{P}(E), \subset)$. O supremo e o ínfimo de um subconjunto $\mathbf{Q}$ do reticulado completo $(\mathbf{\Psi}, <)$ satisfazem, respectivamente, para todo $X \in \mathcal{P}(E)$,

$$(\vee \mathbf{Q})(X) = \cup \{\psi(X): \psi \in \mathbf{Q}\}, \text{ e}$$

$$(\wedge \mathbf{Q})(X) = \cap \{\psi(X): \psi \in \mathbf{Q}\}.$$

O *complemento* de um subconjunto $X \in \mathcal{P}(E)$ é o subconjunto X^c definido por $X^c = \{x \in E: x \notin X\}$. Assim, as igualdades $X \cup X^c = E$ e $X \cap X^c = \varnothing$ são verificadas.

O *mapeamento dual* de $\psi \in \mathbf{\Psi}$ é denotado por ψ^*, e é definido por

$$\psi^*(X) = (\psi(X^c))^c, \ (X \in \mathcal{P}(E)).$$

Observe-se que $\psi^* \in \mathbf{\Psi}$.

3.4 - Operadores elementares

Existem quatro conjuntos fundamentais de operadores sobre $\mathcal{P}(E)$: o conjunto **E** das erosões, o conjunto **D** das dilatações, o conjunto $\mathbf{E}^a$ das anti-erosões, e o conjunto $\mathbf{D}^a$ das anti-dilatações. Esses operadores são chamados de operadores elementares da Morfologia Matemática (MM) pois qualquer operador pode ser decomposto em termos desses operadores básicos (Banon e Barrera, 1991, 1993, 1994, 1998).

Um operador ψ sobre $\mathcal{P}(E)$ é,

uma *erosão* sse $\psi(\wedge\mathcal{X}) = \wedge\psi(\mathcal{X})$,

uma *dilatação* sse $\psi(\vee\mathcal{X}) = \vee\psi(\mathcal{X})$,

uma *anti-erosão* sse $\psi(\wedge\mathcal{X}) = \vee\psi(\mathcal{X})$,

uma *anti-dilatação* sse $\psi(\vee\mathcal{X}) = \wedge\psi(\mathcal{X})$, para todo $\mathcal{X} \in \mathcal{P}(E)$.

Considerando-se $\mathcal{X} = \varnothing$ e lembrando-se que $\vee\varnothing = \varnothing$ e $\wedge\varnothing = E$, temos as seguintes igualdades: para todo $\varepsilon \in \mathbf{E}$, $\varepsilon(E) = E$; para todo $\delta \in \mathbf{D}$, $\delta(\varnothing) = \varnothing$; para todo $\varepsilon^a \in \mathbf{E}^a$, $\varepsilon^a(E) = \varnothing$; e para todo $\delta^a \in \mathbf{D}^a$, $\delta^a(\varnothing) = E$.

As erosões e as dilatações são operadores crescentes, enquanto que as anti-erosões e as anti-dilatações são operadores decrescentes.

3.5 - Operadores invariantes em translação e de janela

Em processamento de imagens é muito utilizado o subconjunto de **Y** que corresponde aos mapeamentos invariantes em translação (i.t.). Este subconjunto será denotado por

$\mathbf{Y}^{\mathbf{it}}$. Para isto, será assumido que (E, +) é um grupo Abeliano em relação à operação binária denotada por +. O elemento zero de (E, +) é denotado por o e chamado de origem de E.

Para todo $u \in E$ e $X \subseteq E$, o subconjunto

$$X_u = \{ y \in E : \ y = x + u, \ x \in X \}$$

é chamado de *translado* de X por u.

Um mapeamento ψ sobre $\mathcal{P}(E)$ é *i.t.* sse ele verifica a relação

$$\psi (X_h) = (\psi (X))_h , \ (X \in \mathcal{P}(E), \ h \in E).$$

O *transposto* (em relação à origem) de $X \subseteq E$ é o subconjunto

$$X^t = \{ x \in E : \ - x \in X \}.$$

Um subconjunto $X \subseteq E$ é simétrico em relação à origem se $X = X^t$.

Notar também que, para $A, B \in \mathcal{P}(E)$,

$$A \subseteq B \cup A^t \subseteq B^t.$$

O conjunto parcialmente ordenado ($\mathbf{Y}^{\mathbf{it}}$, <) é um sub-reticulado completo de ($\mathbf{Y}$, <).

Seja W um subconjunto de E. Um operador ψ é chamado de *operador de janela* W sse, para todo $x \in E$,

$$x \in \psi(X) \Leftrightarrow x \in \psi\left(X \cap W_x\right).$$

Um operador ψ é chamado de *W-operador* sse ele for, ao mesmo tempo, i.t. e operador de janela W (Banon, 1996; Barrera e Hashimoto, 1998).

3.6 - Construção dos operadores elementares invariantes em translação

Com o objetivo de se obter uma definição construtiva dos operadores elementares i.t., usa-se a *adição* (Å) e a *subtração* (§) de Minkowski, definidas, para todo A e $B \in \mathcal{P}(E)$, por:

$$A \oplus B = \{ x \in E : (B^{\vee})_x \cap A \neq \varnothing \},$$

$$A \ominus B = \{ x \in E : B_x \subseteq A \}.$$

Os operadores elementares i.t. são caracterizados por subconjuntos que definem os *elementos estruturantes*. Assim, a expressão $\varepsilon_B(X) = Y$ ($Y \subseteq \mathcal{P}(E)$) será usada para indicar que Y é a erosão de X pelo subconjunto B, para todo B e $X \in \mathcal{P}(E)$. Os elementos estruturantes agem como sondas de formas na análise de imagens. No caso da erosão, o elemento estruturante é o próprio subconjunto B.

Os subconjuntos $B \in \mathcal{P}(E)$ que definem os elementos estruturantes são representados usando-se a notação matricial, sendo que o elemento posicionado na origem é grafado em negrito. Em geral, estes subconjuntos caracterizam-se por apresentarem muitos 0's com alguns poucos 1's agrupados, ou exatamente o contrário, como em

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & \mathbf{1} & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \text{ou} \quad B = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & \mathbf{1} & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Nesses casos é útil representar-se apenas as partes relevantes, suprimindo-se a repetição do algarismo abundante, o qual será representado apenas por um subscrito à direita da matriz, como está mostrado em

$$B = \begin{bmatrix} 0 & 1 & 0 \\ 1 & \mathbf{1} & 1 \\ 0 & 1 & 0 \end{bmatrix}_0 \quad \text{ou} \quad B = \begin{bmatrix} 1 & 0 & 1 \\ 0 & \mathbf{1} & 0 \\ 1 & 0 & 1 \end{bmatrix}_1.$$

Prova-se (Banon e Barrera, 1994, 1998) que os operadores elementares i.t. sobre $\mathcal{P}(E)$ são caracterizados por subconjuntos de $\mathcal{P}(E)$ e podem ser escritos em termos da adição e da subtração de Minkowski:

$$\varepsilon_B(X) = X \ominus B$$

$$\delta_B(X) = X \oplus B$$

$$\varepsilon_B^{a}(X) = (X \ominus B)^c$$

$$\delta_B^{a}(X) = (X \oplus B^c)^c$$

3.7 - Operadores sup-gerador e inf-gerador

Dados dois subconjuntos $A, B \in \mathcal{P}(E)$, tais que $A \subseteq B$, define-se como *intervalo fechado*, denotado por $[A, B]$, a coleção de todos os subconjuntos de E que contêm A e estão contidos em B. Isto pode ser expresso por:

$$[A, B] = \{ X \in \mathcal{P}(E):\ A \subseteq X \subseteq B \}$$

Os intervalos fechados também são representados usando-se a notação matricial. Os elementos representados pelo símbolo · e os elementos não representados na matriz podem assumir indiferentemente os valores 0 ou 1. Assim, o intervalo fechado dado por

$$[A,B] = \begin{bmatrix} 0 & 1 & \cdot \\ \cdot & \mathbf{1} & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

corresponde ao intervalo fechado compreendido entre

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & \mathbf{1} & 1 \\ 1 & 1 & 0 \end{bmatrix}_0 \quad \text{e} \quad B = \begin{bmatrix} 0 & 1 & 1 \\ 1 & \mathbf{1} & 1 \\ 1 & 1 & 0 \end{bmatrix}_1 .$$

Os operadores *sup-gerador* ($l_{A,B}$) e *inf-gerador* ($m_{A,B}$) sobre $\mathcal{P}(E)$ de parâmetros A, B $\in \mathcal{P}(E)$, tal que $A \subseteq B$, são definidos por:

$$\lambda_{A,B} = \varepsilon_A \wedge \delta_{B^t}^{a}$$

$$\mu_{A,B} = \delta_A \vee \varepsilon_{B^{ct}}^{a}$$

O transformado de um subconjunto $X \in \mathcal{P}(E)$ pelo operador sup-gerador de parâmetros A e B é o subconjunto dado por:

$$l_{A,B}(X) = \{ x \in E : A_x \subseteq X \subseteq B_x \},$$

e pelo operador inf-gerador:

$$m_{A,B}(X) = \{ x \in E : (A^t)_x \cap X^{1} \neq \emptyset \text{ ou } (B^t)_x \supseteq X^{1} E \}.$$

É interessante notar (Banon e Barrera, 1991) que

$$\mu_{A,B} = \lambda^{*}_{A^t,B^t}.$$

O operador sup-gerador pode ser representado esquematicamente do modo ilustrado na Figura 3.1.

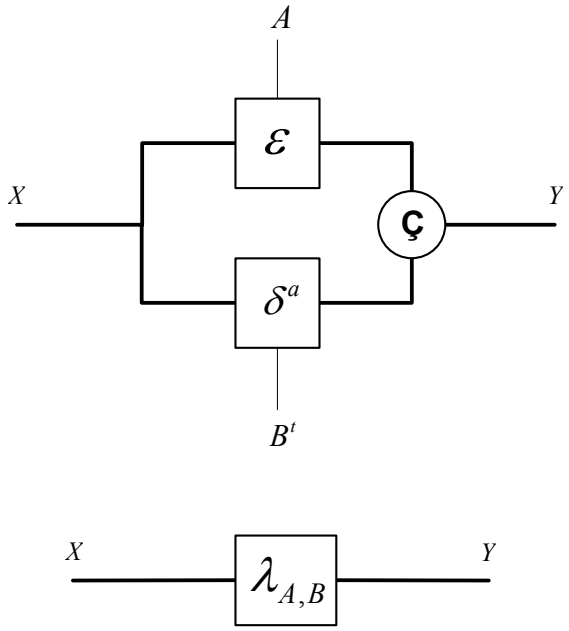


Figura 3.1 - Representações equivalentes do operador sup-gerador de parâmetros *A* e *B*.

Os operadores sup-geradores e inf-geradores definidos desse modo são i.t.

3.8 - Decomposição dos operadores invariantes em translação

O *núcleo* (*kernel*, em inglês) $\mathcal{K}(\psi)$ de um mapeamento i.t. $\psi \in \mathbf{Y}^{it}$, é a subcoleção de $\mathcal{P}(E)$ definida por

$$\mathcal{K}(\psi) = \{ X \in \mathcal{P}(E): \ o \in \psi(X) \},$$

onde *o* é a origem de E.

Se $\psi(X) \neq \emptyset$ para algum $X \in \mathcal{P}(E)$, e $u \in \psi(X)$, então $o \in \psi(X_{-u})$, e em conseqüência, $X_{-u} \in \mathcal{K}(\psi)$. Ou seja, o núcleo de um operador i.t. é vazio sse ψ faz o mapeamento de todo subconjunto $X \in \mathcal{P}(E)$ para $\emptyset$.

Todo operador i.t. $\psi \in \mathbf{Y}^{it}$ pode ser representado pelo supremo de sup-geradores, ou de forma equivalente, pelo ínfimo de inf-geradores (Banon e Barrera, 1991):

$$\psi = \bigvee \{ \iota_{A,B} : [A, B] \in \mathcal{K}(\psi) \} \text{ ou}$$

$$\psi = \bigwedge \{ \mu_{A',B'} : [A, B] \in \mathcal{K}(\psi^*) \}.$$

É interessante notar que no caso da decomposição em termos do supremo de sup-geradores (resp. ínfimo de inf-geradores), se ψ (resp. ψ^*) for W-operador então, para todo $[A, B]$ incluso no núcleo,

$$[A, B] \in \mathcal{P}(W).$$

Sendo E finito, os intervalos fechados de $\mathcal{K}(\psi)$ são enumeráveis, e a decomposição de um operador ψ em termos do supremo de sup-geradores pode ser ilustrado como na Figura 3.2.

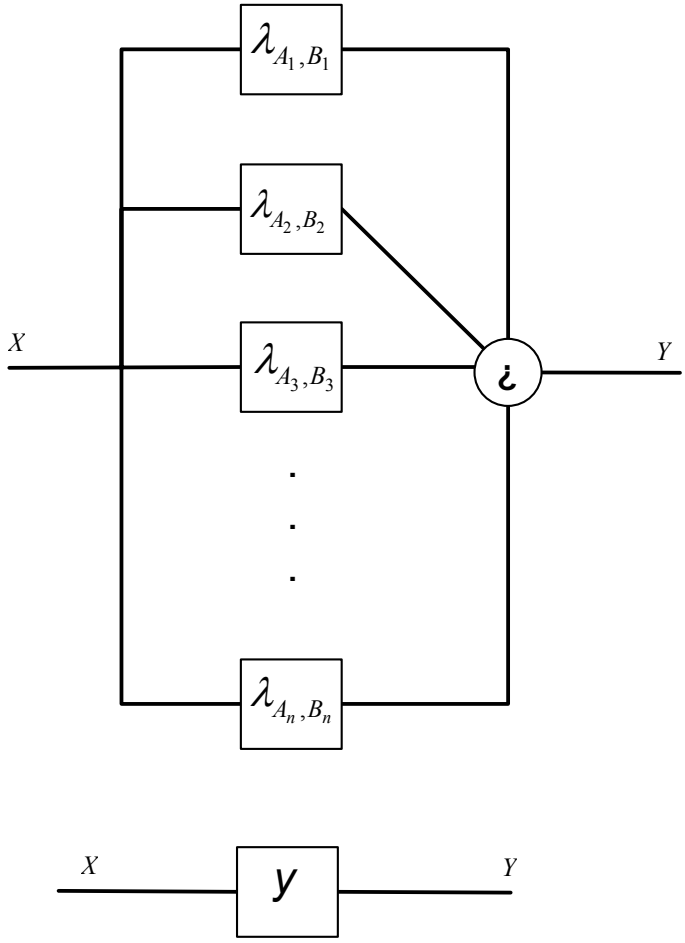


Figura 3.2 - Representações equivalentes de um operador i.t.

Dado um operador i.t. $y \in \mathbf{Y}^{it}$, sua decomposição em operadores sup-geradores (ou inf-geradores) é determinada pelos intervalos fechados [A_1, B_1], [A_2, B_2], ... , [A_n, B_n], representados genericamente por [A_i, B_i].

Embora o número de intervalos fechados n possa ser, em teoria, um número extremamente grande, neste trabalho verifica-se a possibilidade prática de utilizar-se um valor pequeno para n, sem que se perca a eficácia de uso desta forma de decomposição de operadores morfológicos invariantes em translação.

O problema que este trabalho tenta resolver, com o auxílio de um GA, pode ser colocado da seguinte forma: "Dados X, $Y \in \wp(E)$, quais são os intervalos fechados [A_i, B_i] que caracterizam um operador i.t. $y \in \mathbf{Y}^{it}$, tal que $Y = y(X)$?".

CAPÍTULO 4

IMPLEMENTAÇÃO EM COMPUTADOR E RESULTADOS

"Dados X, $Y \hat{I} \mathcal{P}(E)$, quais são os intervalos fechados [A_i, B_i] que caracterizam um operador i.t. $\psi \hat{I} \mathbf{Y}^{it}$, tal que $Y = \psi (X)$?"

A decomposição de um operador $\psi \hat{I} \mathbf{Y}^{it}$ será investigada usando-se a forma dada pela união de operadores sup-geradores. Para cada operador sup-gerador deve-se determinar o intervalo fechado [A, B] que o parametriza. Com o objetivo de se usar um GA que utilize cadeias binárias de comprimento fixo, é necessário fixar o número de operadores sup-geradores que irão decompor ψ. Adicionalmente, só serão considerados como candidatos os intervalos fechados que possam ser representados por matrizes 3x3, com a origem o posicionada no centro da matriz.[1]

Em decorrência, a melhor solução que pode ser encontrada para um ψ, em geral, se limita a uma aproximação $\hat{\psi}$, que pode ser aceitável ou não, dependendo dos pré-requisitos do problema de aplicação. Inicialmente decidiu-se usar oito sup-geradores para observar o comportamento do método sob essa configuração. Como, no decorrer dos testes, esse número não ofereceu nenhuma restrição ao método, acabou sendo adotado ao longo deste trabalho. Desta forma, neste trabalho foram usados oito operadores sup-geradores para a investigação da decomposição de um operador i.t. com a ajuda de um GA.

4.1 - Codificação em cadeias binárias

Para a implementação da busca de uma solução, os oito intervalos fechados [A_i, B_i] (i = 1, ... , 8) que parametrizam os oito sup-geradores serão codificados em cadeias binárias. Para isto será feita a seguinte codificação:

[1] A limitação imposta pelo uso desses intervalos fechados tem seu lado positivo por seu potencial de reutilização, em outras imagens, do operador decomposto, uma vez que as transformações serão de pequena vizinhança..

Codificação	Significado
00	0
01	1
10	·
11	·

Tabela 4.1 – Conteúdo das células da matriz que representa o intervalo fechado

A redundância incorporada desta forma na codificação, apesar de aumentar artificialmente o espaço de busca, permite que os operadores genéticos de cruzamento e mutação produzam sempre cadeias binárias válidas, economizando o tempo que seria gasto, de outra forma, em operações para validar e consertar cadeias binárias. Em geral este tipo de redundância não penaliza o GA, sendo adotada neste trabalho após constatação do bom desempenho demonstrado por uma codificação com redundância similar no trabalho de Harvey e Marshall (1994).

A alternativa apresentada pelo uso de um alfabeto ternário para codificação em cadeias poderia ter sido experimentada neste caso. Todavia, como tal procedimento não é normalmente recomendado em virtude de apresentar, em geral, um desempenho inferior em relação à codificação em cadeias binárias (Goldberg,1989), as rotinas computacionais que implementam o GA foram todas otimizadas para o uso de cadeias binárias, o que as torna robustas e de uso geral. Assim, optou-se pela codificação em cadeias binárias.

Como serão utilizados intervalos fechados definidos por janelas 3x3, cada intervalo fechado usará 18 bits para ser representado. Conforme a cadeia binária é varrida da esquerda para a direita, a matriz deve ser percorrida da esquerda para a direita, da linha de cima até a linha de baixo. Como exemplo, a cadeia binária

$$\overset{1}{0}\overset{}{0}\underset{2}{0}\overset{3}{1}\overset{}{0}\underset{4}{1}\overset{5}{1}\overset{}{0}\underset{6}{1}\overset{7}{0}\overset{}{0}\underset{8}{1}\overset{9}{0}\overset{}{0}$$

representa o intervalo fechado

$$[A,B] = \begin{bmatrix} \overset{1}{0} & \overset{2}{1} & \overset{3}{1} \\ \overset{4}{\cdot} & \overset{\mathbf{5}}{\mathbf{1}} & \overset{6}{0} \\ \overset{7}{\cdot} & \overset{8}{\cdot} & \overset{9}{0} \end{bmatrix}.$$

Uma vez que cada operador será decomposto em 8 sup-geradores, o GA usará populações de indivíduos representados por cadeias binárias de comprimento de 144 bits.

4.2 - Função de adaptação

Os operadores a serem encontrados serão definidos a partir de duas imagens de amostra, chamadas deste ponto em diante de X' e Y', de modo que $Y' = \psi(X')$.

Como cada indivíduo representa uma aproximação $\hat{\psi}$ de um operador ψ, deve-se comparar $\hat{Y}' = \hat{\psi}(X')$ com o subconjunto Y' dado, avaliando-se com isso a adaptação de cada indivíduo. Essa comparação entre imagens binárias (subconjuntos) de mesmas dimensões espaciais é feita pela contagem do número de pixels iguais em posições correspondentes das duas imagens, isto é,

$$obj(\hat{\psi}) = \#\left(\left(\hat{Y}' \cap Y'\right) \cup \left(\hat{Y}'^{c} \cap Y'^{c}\right)\right),$$

sendo usada a expressão #X para representar o número de elementos do conjunto X. O objetivo do GA é encontrar $\hat{\psi}$ que maximize

$$adapt(\hat{\psi}) = \left(\frac{obj(\hat{\psi})}{c}\right)^{r}.$$

4.3 - Parâmetros adotados

Os diversos parâmetros que envolvem a implementação de um GA em computador precisam ser ajustados de modo a otimizar o desempenho do algoritmo . Em geral este

ajuste é feito por tentativa-e-erro através da realização de alguns testes. Assim, após uma tentativa inicial, com fraco desempenho, usando-se uma população de 50 indivíduos, a obtenção de um bom desempenho na segunda tentativa, com 90 indivíduos na população, foi suficiente para escolher esse valor para a maioria dos testes feitos a seguir. Em um dos testes posteriores, quando se objetivou encontrar a solução exata, foram utilizados os valores de 100, 200, e finalmente 300 indivíduos para a população, quando o resultado procurado foi encontrado.

Do mesmo modo, inicialmente foi usado para a probabilidade de cruzamento o valor de 0,90. Em seguida, testes realizados com o valor 0,85 mostraram-se melhores. Novos testes com valor de 0,80 voltaram a degradar o desempenho. Foi então escolhido o valor de 0,85 para a probabilidade de cruzamento em todos os casos estudados neste trabalho.

Os primeiros testes foram feitos usando-se o valor de 0,01 para a probabilidade de mutação de um bit. Estes testes indicaram uma convergência por volta de 50 gerações, o que levou a se adotar a seguinte estratégia: um valor de 0,1 para a probabilidade de mutação de um bit nas 20 primeiras gerações, visando a obtenção da maior diversidade possível de indivíduos avaliados nas gerações iniciais; um valor de 0,01 até a geração 40; e um valor de 0,001 para as gerações restantes, objetivando a convergência fina do algoritmo. Para a função de adaptação usou-se o valor 8 para o expoente r ; o valor de c variou de caso para caso, tendo sido escolhido de forma que os valores finais da função de adaptação ficassem entre 0 e 100.

4.4 - Programação em computador

Foram usadas as rotinas computacionais da biblioteca MMach 1.4 (Lotufo, 1997) para manipular imagens e aplicar os operadores morfológicos às imagens. Foi feito o uso da plataforma KHOROS para a entrada de dados (duas imagens), visualização das imagens, e como plataforma de desenvolvimento de programas. Todo o trabalho computacional foi desenvolvido usando-se a linguagem de programação C, e o GA implementado, descrito na seção 2.2 do Capítulo 2, foi baseado no “simple genetic

algorithm" (SGA) apresentado em Goldberg (1989). O tempo médio de convergência do GA para os casos estudados foi de 7 minutos em uma Sun Sparc station 20.

Ao longo dos resultados que vêm a seguir, serão adotadas as seguintes convenções: X é a imagem a ser transformada (filtrada, por exemplo) pelo operador ψ, fornecendo a imagem Y desejada, ou seja, $Y = \psi(X)$; X' e Y', que se relacionam por $Y' = \psi(X')$, servirão como imagens de entrada para o GA realizar a busca dos $[A_i, B_i]$ (neste trabalho X' e Y' foram extraídos do quarto superior esquerdo de X e Y, respectivamente); e $\hat{Y}$ é a imagem resultante da aplicação do operador $\hat{\psi}$ (definido pelos intervalos fechados $[A_i, B_i]$ encontrados pelo GA) em X, $\hat{Y} = \hat{\psi}(X)$. Deseja-se que $\hat{Y}$ seja o mais próximo possível de Y. A Figura 4.1 ilustra o uso dessas convenções na realização de $\hat{\psi}$.

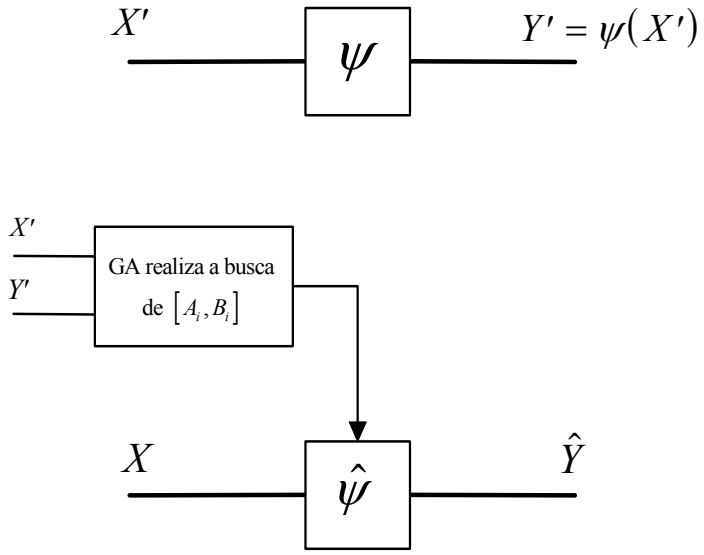


Figura 4.1 - Realização do operador $\hat{\psi}$.

4.5 - Decomposição da dilatação

Nesta seção será investigada a decomposição da dilatação pelo subconjunto $B = \begin{bmatrix} 0 & 1 & 0 \\ 1 & \mathbf{1} & 1 \\ 0 & 1 & 0 \end{bmatrix}$, $Y = \delta_B(X)$.

A Figura 4.2 mostra as imagens X', Y', X e Y usadas na decomposição da dilatação, e a imagem $\hat{Y}$ resultante da aplicação do operador decomposto (e aproximado) $\hat{\psi}$.

Figura 4.2 - Decomposição da dilatação: **a)** X', **b)** Y', **c)** X, **d)** Y, **e)** $\hat{Y}$.

A Figura 4.3 mostra em **a** e **b** as imagens Y e $\hat{Y}$ ampliadas de um fator 1,8 para melhor visualização das diferenças, e em **c**, a diferença simétrica entre Y e $\hat{Y}$, ou seja, a imagem que mostra os pontos que, em posições correspondentes, diferem nas duas imagens. Para quantificar a qualidade dos resultados, usa-se a porcentagem de pixels idênticos nas duas imagens, Y e $\hat{Y}$. Este valor foi de 99,688% para o caso desta dilatação estudada, atestando o bom desempenho do método.

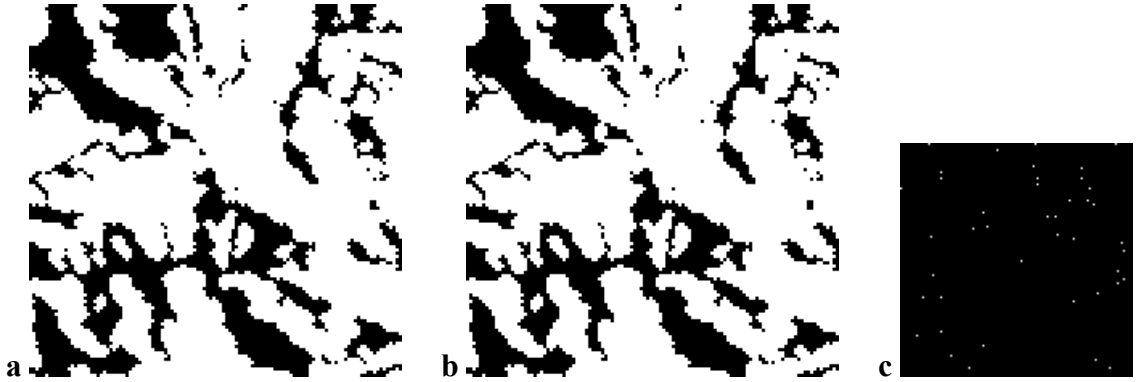


Figura 4.3 - Decomposição da dilatação (**a** e **b** ampliadas 1,8x): **a**) Y, **b**) $\hat{Y}$, **c**) diferença simétrica entre Y e $\hat{Y}$.

É interessante notar que em nenhum dos casos abordados neste trabalho o GA precisou usar os 8 operadores sup-geradores para convergir. No caso presente da dilatação, por exemplo, apenas 7 sup-geradores foram utilizados pelo GA.

A Figura 4.4 mostra as imagens correspondentes aos $\lambda_{A_i,B_i}(X)$ (i = 1, ... , 7) para solução encontrada. A união dessas 7 imagens fornece $\hat{Y}$. A Figura 4.5 mostra os intervalos fechados correspondentes.

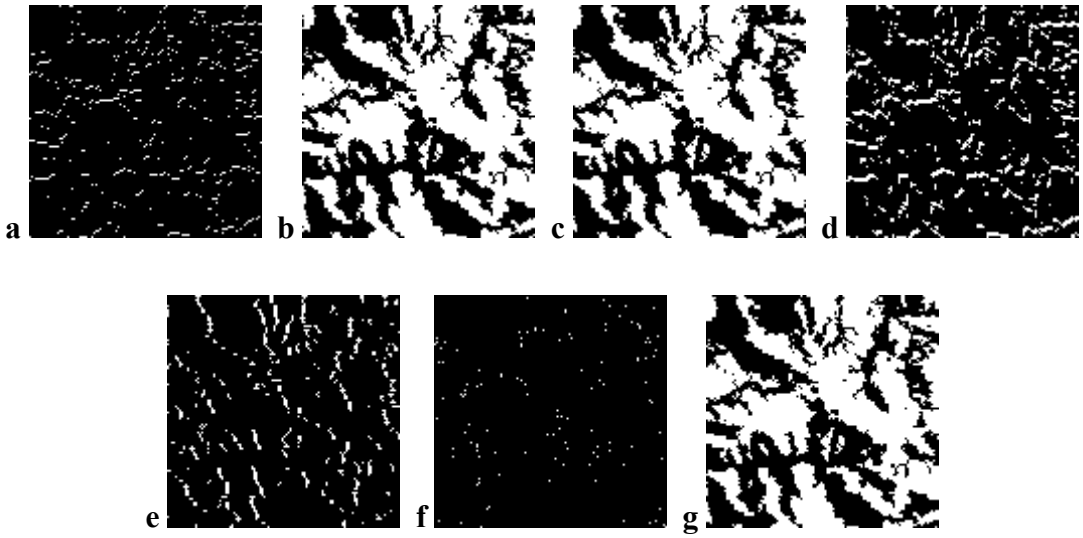


Figura 4.4 - Imagens resultantes da aplicação dos 7 sup-geradores em *X*, para a dilatação.

$$\mathbf{a}\begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \mathbf{0} & \cdot \\ \cdot & 1 & 1 \end{bmatrix} \quad \mathbf{b}\begin{bmatrix} \cdot & \cdot & \cdot \\ 1 & \bullet & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix} \quad \mathbf{c}\begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \bullet & 1 \\ \cdot & \cdot & \cdot \end{bmatrix} \quad \mathbf{d}\begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \bullet & \cdot \\ \cdot & 1 & 0 \end{bmatrix}$$

$$\mathbf{e}\begin{bmatrix} \cdot & \cdot & 1 \\ \cdot & \bullet & 0 \\ \cdot & \cdot & 1 \end{bmatrix} \quad \mathbf{f}\begin{bmatrix} \cdot & \cdot & \cdot \\ 0 & \mathbf{1} & 0 \\ \cdot & \cdot & \cdot \end{bmatrix} \quad \mathbf{g}\begin{bmatrix} \cdot & 1 & \cdot \\ \cdot & \bullet & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix}$$

Figura 4.5 - Intervalos fechados que parametrizam os 7 sup-geradores para a dilatação.

A solução exata para a decomposição desta dilatação é dada pelos 5 sup-geradores caracterizados pelos intervalos fechados $\begin{bmatrix} \cdot & 1 & \cdot \\ \cdot & \bullet & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix}$, $\begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \bullet & 1 \\ \cdot & \cdot & \cdot \end{bmatrix}$, $\begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \bullet & \cdot \\ \cdot & 1 & \cdot \end{bmatrix}$, $\begin{bmatrix} \cdot & \cdot & \cdot \\ 1 & \bullet & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix}$ e $\begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \mathbf{1} & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix}$.

Desse modo, os intervalos fechados dados pelas letras **b**, **c**, **d** e **g** da Figura 4.5 representam a parte relevante da solução encontrada pelo GA. É oportuno lembrar que a contribuição do último sup-gerador da solução exata dada acima é pequeno ou nulo em

muitos casos. Isto ocorre porque o supremo dos quatro primeiros sup-geradores da solução exata, em geral, engloba a parcela que corresponderia ao quinto sup-gerador.

Pode acontecer também que algum dos sup-geradores que compõem a solução encontrada pelo GA contribua negativamente, isto é, a solução encontrada seria melhor sem a contribuição de tal sup-gerador. Isso ocorre, neste caso, com o sup-gerador dado pelo intervalo fechado apresentado pela letra **e** da Figura 4.5 acima. A porcentagem de pixels idênticos entre Y e $\hat{Y}$ seria de 99,911% se este sup-gerador não fosse incluído na solução encontada.

Outro ponto a ser ressaltado é que a solução exata pode ser encontrada, neste e em outros casos estudados, através do ajuste do GA. Um modo disto ser feito, como será visto mais adiante, é aumentar o tamanho da população e o número de gerações.

Para se ter uma noção mais clara a respeito da convergência do método, a Figura 4.6 mostra o comportamento da adaptação do melhor indivíduo e da média da população em função da geração. Notar que o valor de 0,1 utilizado para a probabilidade de mutação de um bit nas 20 primeiras gerações é alto demais, não permitindo que o GA tenha um bom desenvolvimento evolutivo durante este período. No entanto, a utilização de um valor bem pequeno (0,001) para a probabilidade de mutação de um bit para as últimas 20 gerações parece ser uma boa estratégia para garantir uma convergência estável do algoritmo.

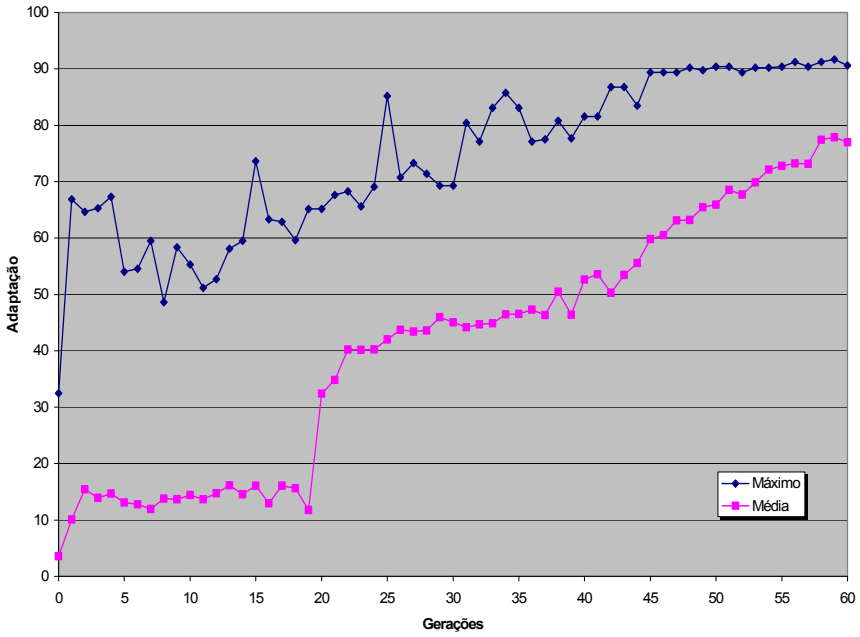


Figura 4.6 - Evolução da adaptação ao longo das gerações para a decomposição da dilatação: melhor indivíduo e média da população.

Gráficos de dispersão da adaptação entre os indivíduos da população inicial, e das populações da 10^{a}, 30^{a}, e 50^{a} gerações são apresentados a seguir. A partir desses gráficos pode-se observar a evolução qualitativa dos indivíduos da população ao longo das gerações. Vê-se na Figura 4.7 o perfil da adaptação da população inicial gerada de forma aleatória: aproximadamente 80% dos indivíduos estão situados abaixo da média da população; os 20% restantes serão os pais da maior parte dos indivíduos da população seguinte. Depois de 10 gerações, pode-se ver na Figura 4.8 que aproximadamente metade da população já está acima da média, que cresceu 4 vezes; no mesmo intervalo de tempo, o melhor indivíduo quase dobrou o valor de sua adaptação. Mas ainda boa parte da população, por volta de 40%, continua com a adaptação muito ruim. Após 30 gerações, no entanto, o quadro já se apresenta bem diferente, como mostra a Figura 4.9: a maior parte dos indivíduos situa-se acima da média, e são bem poucos os indivíduos com adaptação muito fraca. A média já aumentou mais de 12 vezes até este ponto. Próximo à convergência, na geração de número 50, pode-se ver na Figura 4.10 que vários indivíduos disputam o posto de melhor adaptado.

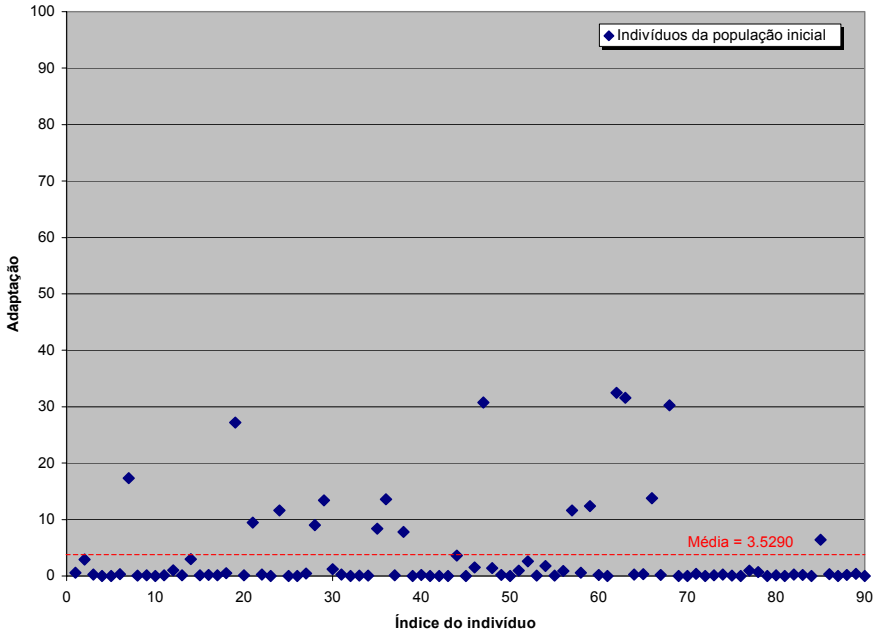


Figura 4.7 - Dispersão da adaptação na população inicial.

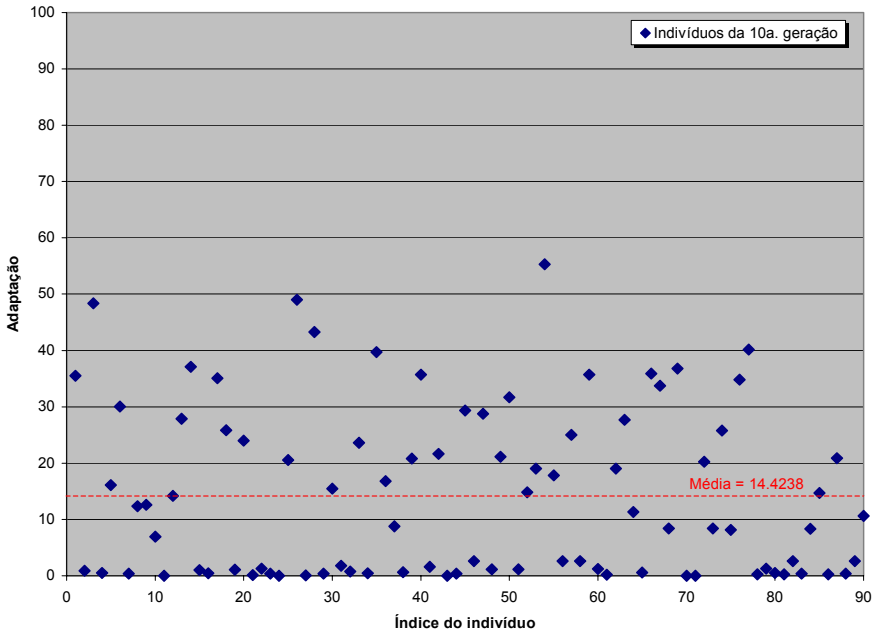


Figura 4.8 - Dispersão da adaptação na população da 10ª geração.

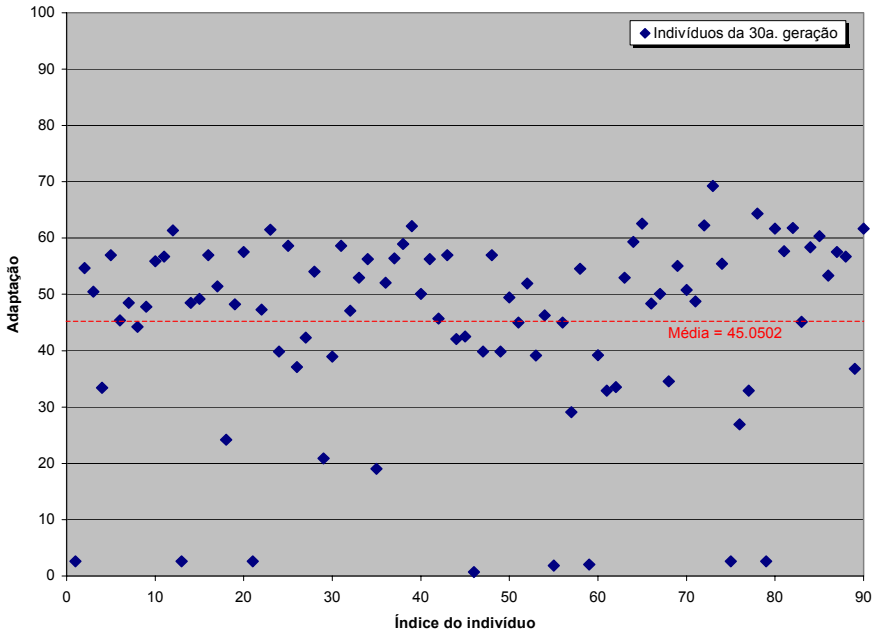


Figura 4.9 - Dispersão da adaptação na população da 30ª geração.

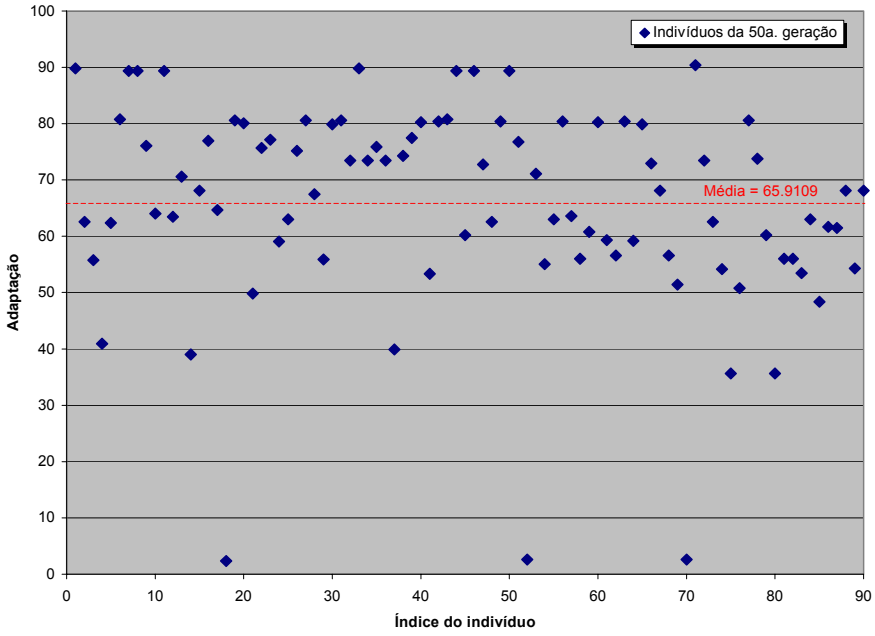


Figura 4.10 - Dispersão da adaptação na população da 50ª geração.

4.6 - Decomposição do extrator de bordas

Lembrando-se que a diferença entre dois subconjuntos A e B é definida como $A - B = A \cap B^c$, a imagem Y das bordas da imagem X pode ser obtida fazendo-se $Y = X - \varepsilon_B(X)$, conforme ilustrado na Figura 4.11.

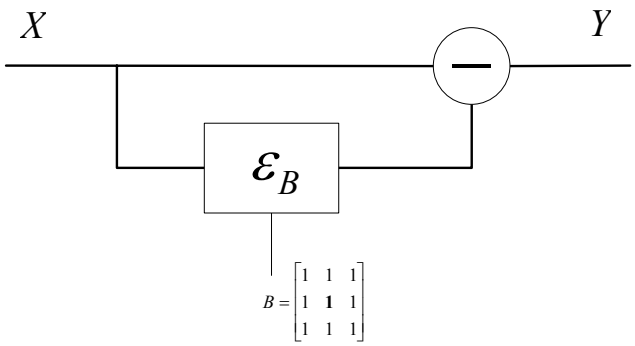


Figura 4.11 - Extrator de borda.

As imagens X, Y^c e $\hat{Y}^c$ para o caso do extrator de bordas são mostradas na Figura 4.12. Neste caso utilizou-se o complemento de várias imagens, como Y^c e $\hat{Y}^c$, visando-se uma qualidade melhor de impressão em papel.

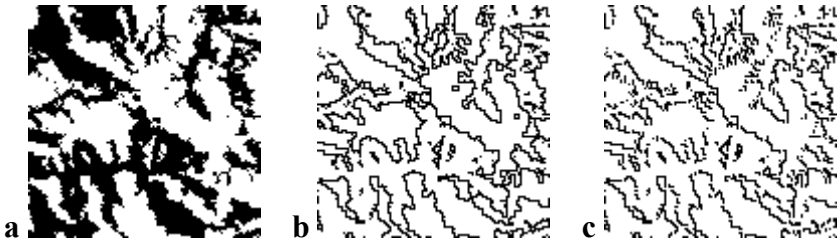


Figura 4.12 - Decomposição do extrator de bordas: **a**) X, **b**) Y^c, **c**) $\hat{Y}^c$.

Como antes, as imagens Y^c e $\hat{Y}^c$ são mostradas com ampliação de fator 2 pela Figura 4.13. A porcentagem de pixels idênticos neste caso foi medida em 98,573%.

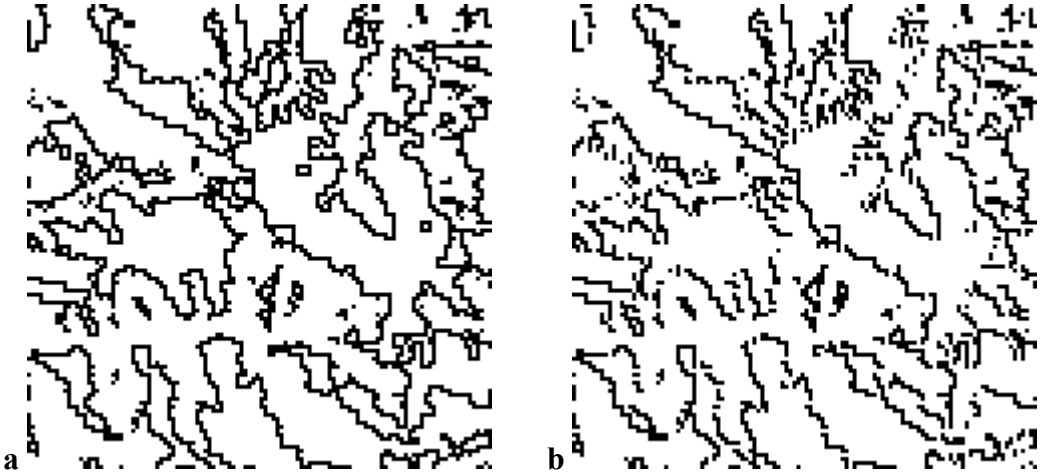


Figura 4.13 - Decomposição do extrator de bordas (ampliadas 2x): **a)** Y^c, **b)** $\hat{Y}^c$.

A Figura 4.14 exibe os complementos das imagens geradas pelos 5 sup-geradores obtidos pelo GA, e a Figura 4.15 lista os intervalos fechados correspondentes.

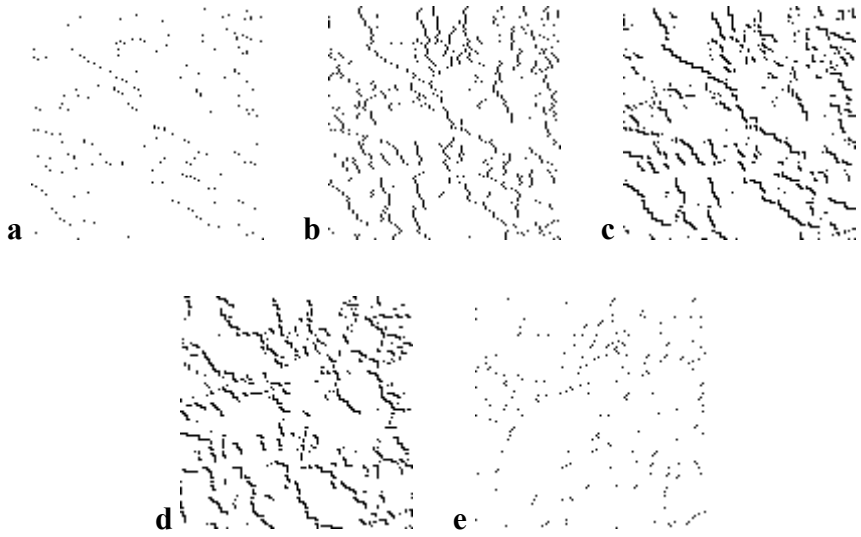


Figura 4.14 - Imagens resultantes da aplicação dos 5 sup-geradores em X, para o extrator de bordas.

$$\mathbf{a}\begin{bmatrix} 0 & \cdot & 0 \\ \cdot & \mathbf{1} & 0 \\ \cdot & \cdot & \cdot \end{bmatrix} \quad \mathbf{b}\begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \mathbf{1} & 0 \\ \cdot & \cdot & \cdot \end{bmatrix} \quad \mathbf{c}\begin{bmatrix} \cdot & \cdot & 0 \\ \cdot & \mathbf{1} & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix} \quad \mathbf{d}\begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \mathbf{1} & \cdot \\ 0 & \cdot & \cdot \end{bmatrix} \quad \mathbf{e}\begin{bmatrix} \cdot & 1 & 1 \\ 1 & \bullet & \cdot \\ \cdot & \cdot & 0 \end{bmatrix}$$

Figura 4.15 - Intervalos fechados que parametrizam os 5 sup-geradores para a dilatação.

Pode-se ver na Figura 4.16 a convergência dos valores máximo e médio da adaptação ao longo das gerações.

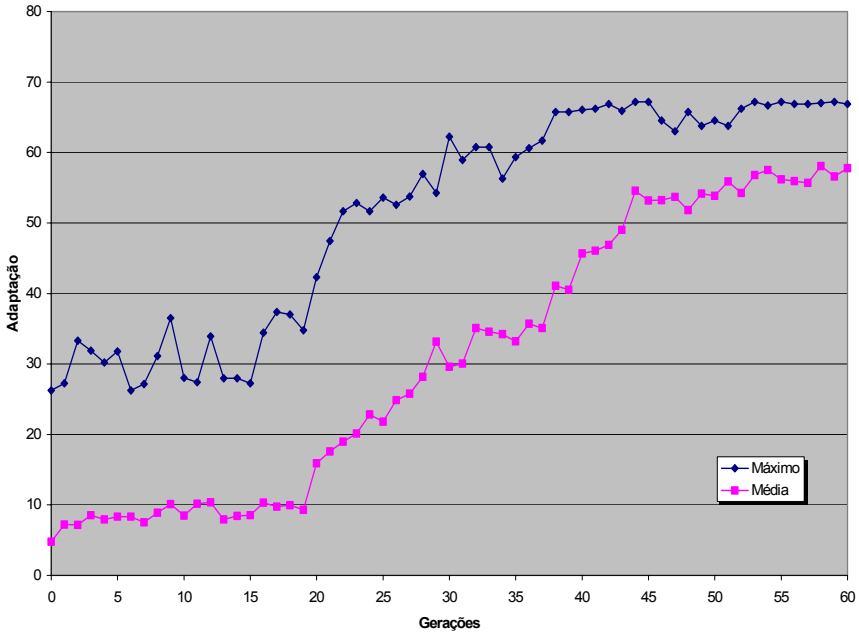


Figura 4.16 - Evolução da adaptação ao longo das gerações para a decomposição do extrator de bordas: melhor indivíduo e média da população.

O resultado obtido com a decomposição do extrator de bordas mostrou-se muito bom.

4.7 - Decomposição dos filtros de ruído sal-e-pimenta e de ruído gaussiano

A construção de um filtro para imagens corrompidas com ruído do tipo sal-e-pimenta apresenta alguns desafios. É interessante observar como o modelo de decomposição proposto se comporta neste caso. A Figura 4.17 mostra as imagens X, Y e $\hat{Y}$, no caso de um ruído sal-e-pimenta. A porcentagem de pixels idênticos de Y e $\hat{Y}$ neste caso chegou a 98,551%.

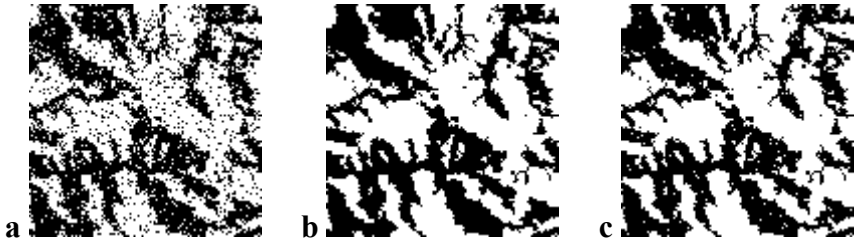


Figura 4.17 - Decomposição do filtro de ruído sal-e-pimenta: **a**) X, **b**) Y, **c**) $\hat{Y}$.

As mesmas imagens aparecem ampliadas por um fator 2 na Figura 4.18, para a visualização de detalhes.

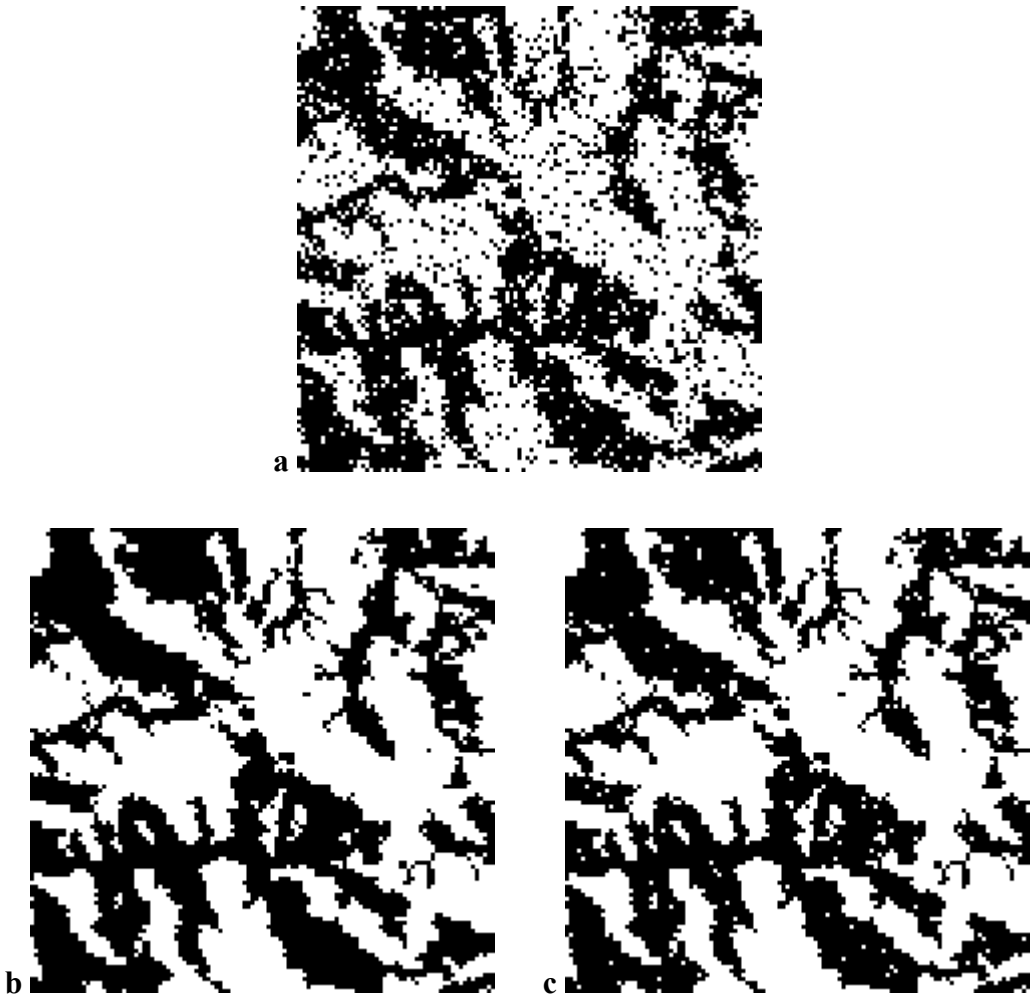


Figura 4.18 - Decomposição do filtro de ruído sal-e-pimenta (ampliadas 2x): **a**) X, **b**) Y, **c**) $\hat{Y}$.

Os seis operadores sup-geradores encontrados pelo GA fornecem como saída da imagem X as imagens apresentadas na Figura 4.19. Os intervalos fechados correspondentes são mostrados na Figura 4.20.

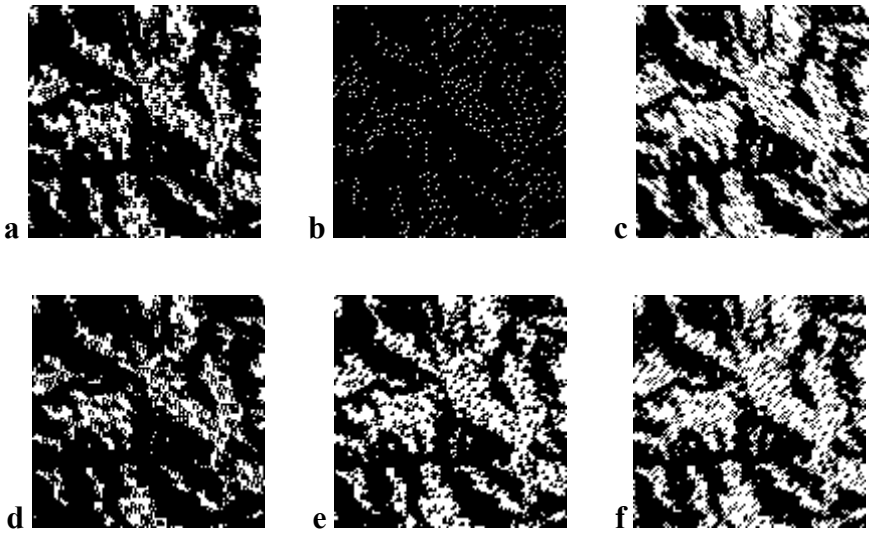


Figura 4.19 - Imagens resultantes da aplicação dos 6 sup-geradores em X, para o filtro de ruído sal-e-pimenta.

$$\mathbf{a}\begin{bmatrix}1 & 1 & \cdot\\ 1 & \bullet & \cdot\\ 1 & 1 & \cdot\end{bmatrix}\quad \mathbf{b}\begin{bmatrix}\cdot & 1 & \cdot\\ 1 & \mathbf{1} & 1\\ \cdot & \cdot & 0\end{bmatrix}\quad \mathbf{c}\begin{bmatrix}\cdot & \cdot & \cdot\\ \cdot & \mathbf{1} & \cdot\\ \cdot & \cdot & 1\end{bmatrix}\quad \mathbf{d}\begin{bmatrix}1 & \cdot & 1\\ 1 & \bullet & 1\\ \cdot & 1 & 1\end{bmatrix}\quad \mathbf{e}\begin{bmatrix}\cdot & 1 & 1\\ \cdot & \mathbf{1} & \cdot\\ \cdot & \cdot & \cdot\end{bmatrix}\quad \mathbf{f}\begin{bmatrix}\cdot & \cdot & \cdot\\ \cdot & \mathbf{1} & \cdot\\ 1 & \cdot & \cdot\end{bmatrix}$$

Figura 4.20 - Intervalos fechados que parametrizam os 6 sup-geradores para o filtro de ruído sal-e-pimenta.

É fácil constatar o bom desempenho do método pela análise dos resultados obtidos. Para o ruído pimenta o filtro teve um comportamento excelente, próximo do ideal; para o ruído sal, o desempenho foi bom, porém inferior ao ruído pimenta. Essa discrepância de comportamento para os casos sal e pimenta deve ser atribuída a uma soma de fatores: uma maior área coberta por branco do que por preto nas imagens (portanto maior quantidade de ruído pimenta para usar como informação); e a formação de um maior número de artefatos brancos em contraste com uma maior quantidade de pontos pretos isolados.

Pode-se melhorar o desempenho do método usando-se imagens de maior dimensão espacial para X' e Y', ou então, como mostrado a seguir, aumentando-se a quantidade de ruído em X'. Este aparente paradoxo se justifica no método adotado de comparação de duas imagens, pois uma maior quantidade de ruído corresponde a uma maior quantidade de informação. A Figura 4.21 ilustra este fato a partir de uma imagem X' corrompida por um forte ruído gaussiano. Para este caso, porcentagem de pixels idênticos de Y e $\hat{Y}$ foi calculada em 90,807%. As Figuras 4.22 e 4.23 mostram os 7 sup-geradores da solução encontrada, e os intervalos fechados correspondentes, respectivamente.

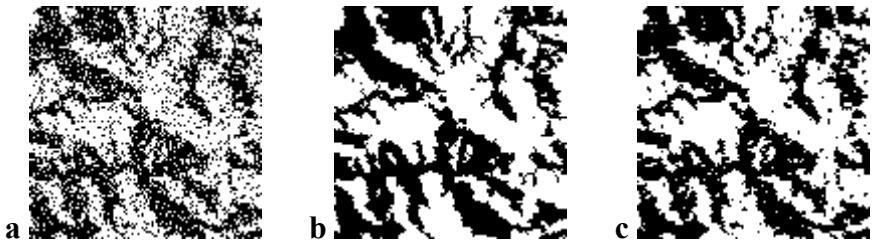


Figura 4.21 - Decomposição do filtro de ruído gaussiano: **a**) X, **b**) Y, **c**) $\hat{Y}$.

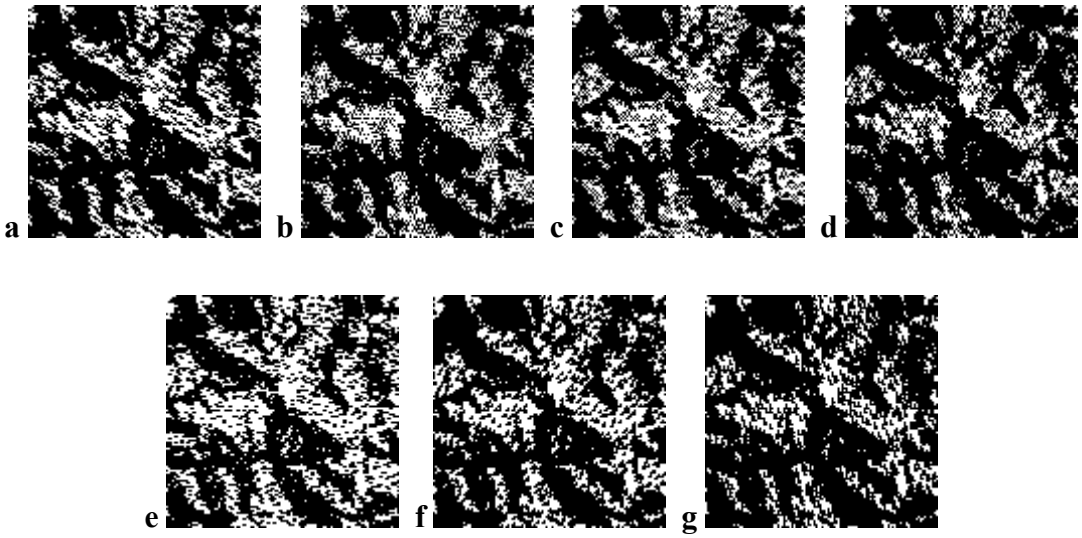


Figura 4.22 - Imagens resultantes da aplicação dos 7 sup-geradores em X, para o filtro de ruído gaussiano.

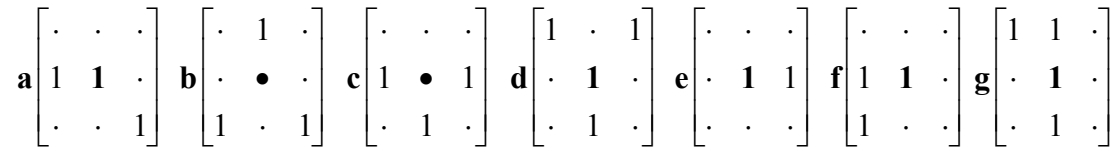


Figura 4.23 - Intervalos fechados que parametrizam os 7 sup-geradores para o filtro de ruído gaussiano.

Em adição foi testada a decomposição dos filtros para os ruídos sal e pimenta separadamente. A Figura 4.24 mostra as imagens X, Y e $\hat{Y}$ para um ruído sal, e a Figura 4.25 apresenta os intervalos fechados que geraram $\hat{Y}$. A porcentagem de pixels idênticos de Y e $\hat{Y}$ atinge aqui 99,101%. As Figuras 4.26 e 4.27 mostram os mesmos resultados para um ruído pimenta, com a porcentagem de pixels idênticos de Y e $\hat{Y}$ chegando a 99,339%.

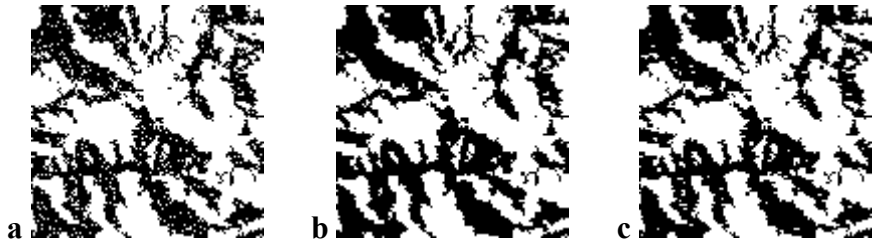


Figura 4.24 - Decomposição do filtro de ruído sal: **a**) X, **b**) Y, **c**) $\hat{Y}$.

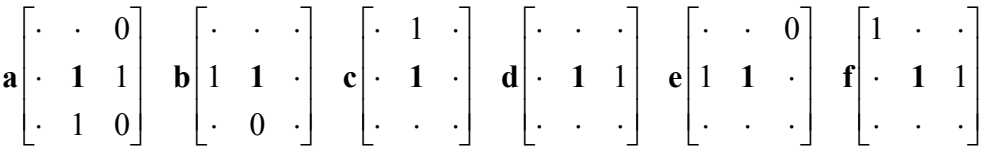


Figura 4.25 - Intervalos fechados que parametrizam os 6 sup-geradores para o filtro de ruído sal.

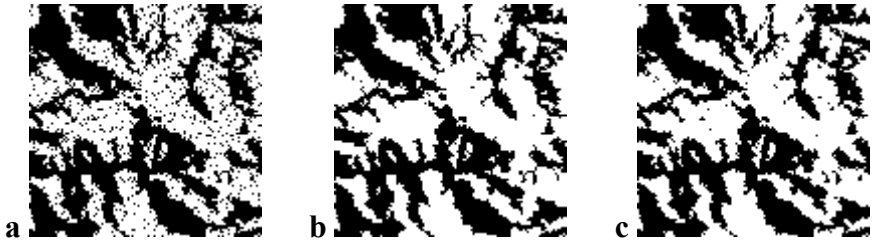


Figura 4.26 - Decomposição do filtro de ruído pimenta: **a**) X, **b**) Y, **c**) $\hat{Y}$.

$$\mathbf{a}\begin{bmatrix} \cdot & \cdot & 1 \\ 1 & \bullet & \cdot \\ \cdot & 1 & \cdot \end{bmatrix} \quad \mathbf{b}\begin{bmatrix} 1 & \cdot & \cdot \\ 0 & \mathbf{1} & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix} \quad \mathbf{c}\begin{bmatrix} 1 & 1 & \cdot \\ \cdot & \bullet & 1 \\ \cdot & \cdot & 1 \end{bmatrix} \quad \mathbf{d}\begin{bmatrix} 0 & \cdot & \cdot \\ \cdot & \mathbf{1} & 1 \\ \cdot & 0 & \cdot \end{bmatrix} \quad \mathbf{e}\begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \mathbf{1} & \cdot \\ \cdot & \cdot & 0 \end{bmatrix} \quad \mathbf{f}\begin{bmatrix} 1 & \cdot & \cdot \\ \cdot & \mathbf{1} & \cdot \\ 1 & \cdot & \cdot \end{bmatrix}$$

Figura 4.27 - Intervalos fechados que parametrizam os 6 sup-geradores para o filtro de ruído pimenta.

Os resultados obtidos mostram o ótimo desempenho do método proposto na decomposição do filtro para ruído do tipo sal-e-pimenta.

4.8 – Decomposição de fechamentos e aberturas morfológicos i.t.

Duas classes importantes de operadores i.t. (Banon e Barrera, 1994, 1998) são a classe dos *fechamentos morfológicos* e a classe das *aberturas morfológicas*. O fechamento e a abertura morfológicos por um subconjunto B definidos, respectivamente, para todo $X \in \mathcal{P}(E)$, por:

$$\phi_B(X) = (X \oplus B) \ominus B,$$

$$\gamma_B(X) = (X \ominus B) \oplus B.$$

Esses operadores possuem a importante propriedade da idempotência, isto é,

$$\phi_B\phi_B = \phi_B,$$

$$\gamma_B\gamma_B = \gamma_B.$$

No entanto, para que a idempotência seja preservada é importante, nesse caso, que a decomposição seja exata. Para isto é melhor utilizar-se de imagens de amostra apropriadas, como o par que define o fechamento ϕ_B apresentado na Figura 4.28, para $B = \begin{bmatrix} \mathbf{1} & 1 \\ 1 & 1 \end{bmatrix}$.

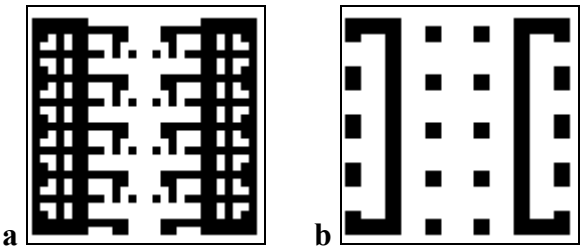


Figura 4.28 – Imagens de amostra para a decomposição do fechamento: **a**) X', **b**) Y'.

Aqui o método mostrou-se eficiente e preciso, convergindo fortemente para a solução exata, que está apresentada no trabalho de Maragos (1985), na expressão (5-26) à página 142.

As imagens resultantes da decomposição do fechamento são mostradas na Figura 4.29, e os intervalos fechados correspondentes, na Figura 4.30.

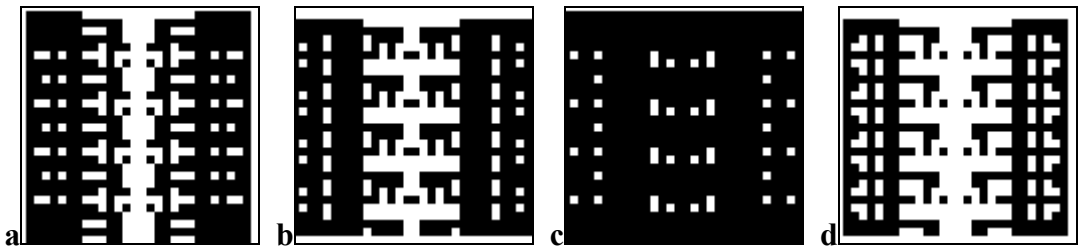


Figura 4.29 – Imagens resultantes da decomposição do fechamento morfológico i.t.

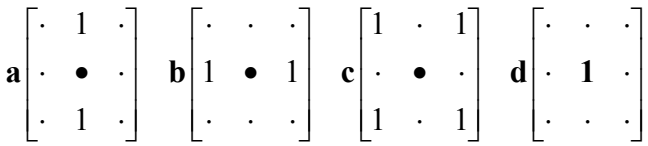


Figura 4.30 – Intervalos fechados que parametrizam os 4 sup-geradores do fechamento.

O mesmo teste foi feito com as aberturas γ_B, utilizando-se do mesmo B, com as imagens de amostra dadas pela Figura 4.31.

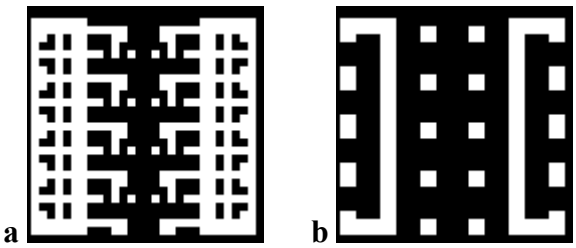


Figura 4.31 - Imagens de amostra para a decomposição da abertura: **a**) X', **b**) Y'.

Neste caso, no entanto, não houve convergência para a resposta exata com a população de 90 indivíduos usada nos testes deste trabalho. Também não houve convergência para populações de 100 e de 200 indivíduos, o que acabou finalmente ocorrendo para uma população de 300 indivíduos. A solução exata assim obtida é mostrada nas Figuras 4.32 e 4.33.

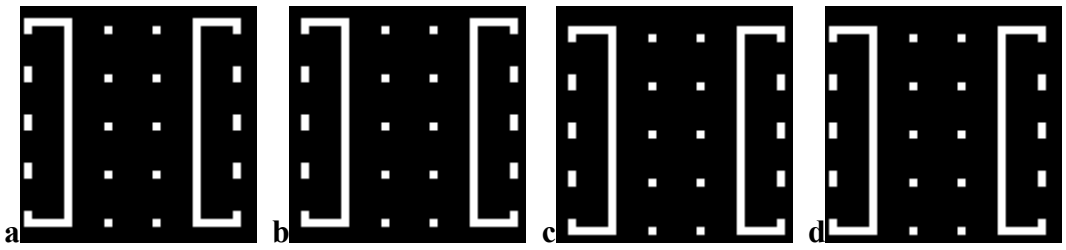


Figura 4.32 – Imagens resultantes da decomposição da abertura morfológica i.t.

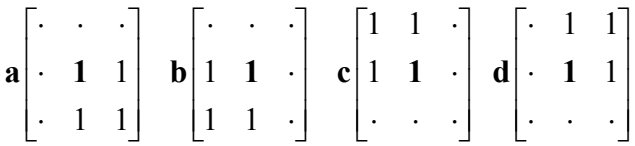


Figura 4.33 – Intervalos fechados que parametrizam os 4 sup-geradores da abertura.

Dado o grau de dificuldade que o método encontrou neste caso, é feita aqui uma experiência visando uma possível melhoria do desempenho do método. Para isto utiliza-se uma operação de cruzamento para o GA diferente da que foi utilizada em todo este trabalho. Em vez de haver apenas um ponto de cruzamento para a cadeia binária completa, haverá um ponto de cruzamento para cada módulo sup-gerador, perfazendo um total de 8 pontos de cruzamento para cada casal de cromossomos; as informações genéticas são intercambiadas apenas entre cada módulo sup-gerador correspondente dos pais, como mostra a figura 4.34.

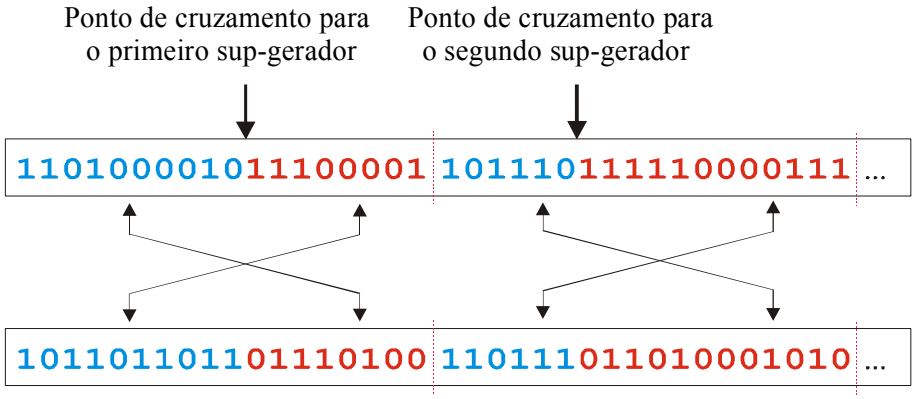


Figura 4.34 – Estratégia de cruzamento por módulo de sup-gerador.

Ao cabo de três testes, com populações de 10, 200 e 300 indivíduos, o método não convergiu para a resposta exata, não tendo havido, portanto, melhora no desempenho do método para este caso. No entanto, observando-se as Figuras 4.35, 4.36 e 4.37 nota-se que a estratégia de cruzamento por módulo pode ser útil quando o tempo, e não a precisão, é o fator primordial, pois o GA convergiu mais rapidamente para boas soluções usando-se esta estratégia.

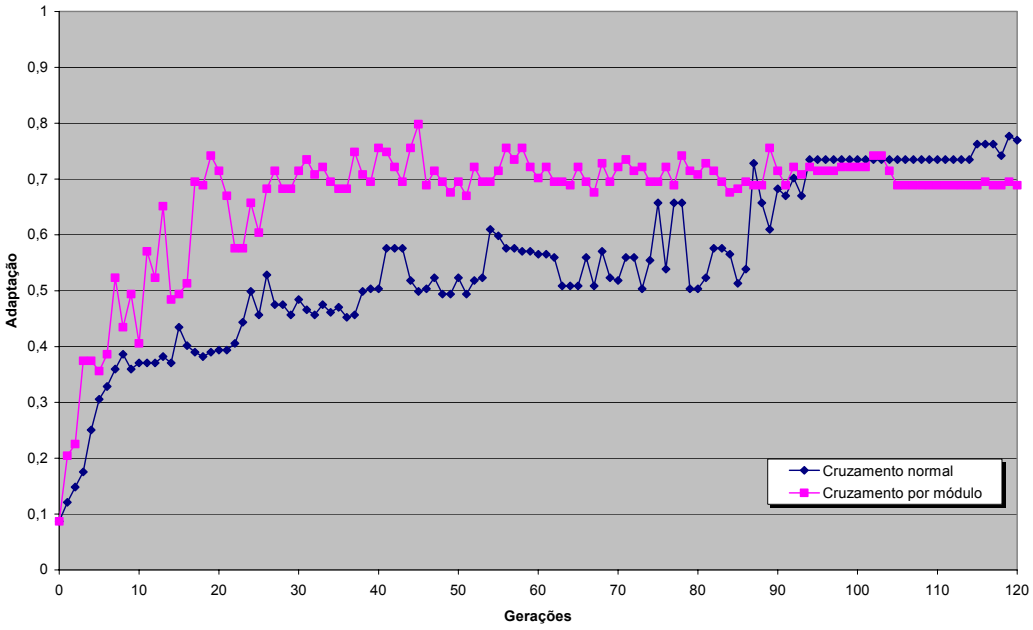


Figura 4.35 - Evolução da adaptação do melhor indivíduo para a decomposição da abertura com população de 100 indivíduos: cruzamento normal e por módulo.

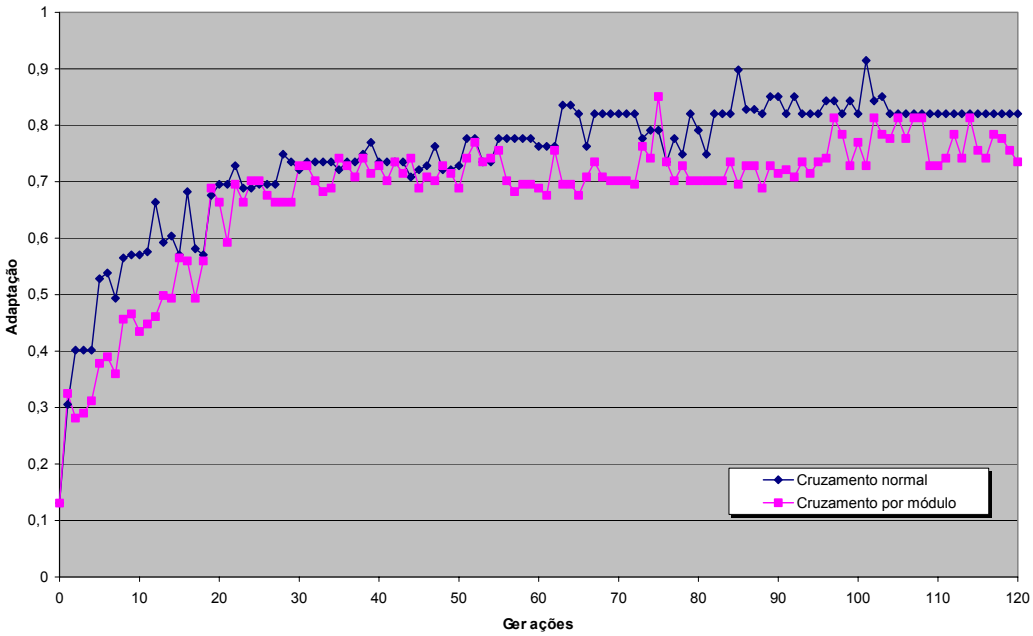


Figura 4.36 - Evolução da adaptação do melhor indivíduo para a decomposição da abertura com população de 200 indivíduos: cruzamento normal e por módulo.

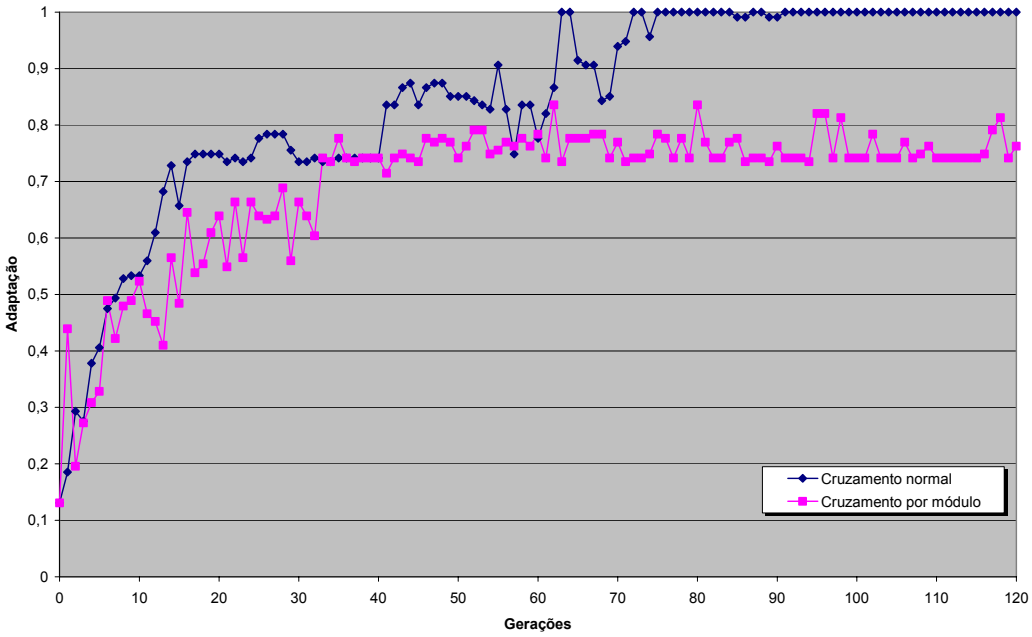


Figura 4.37 - Evolução da adaptação do melhor indivíduo para a decomposição da abertura com população de 300 indivíduos: cruzamento normal e cruzamento por módulo.

As Figuras 4.38, 4.39 e 4.40 ilustram de modo mais claro como variam o valor de convergência, o valor máximo e o tempo de convergência em função do número de indivíduos na população.

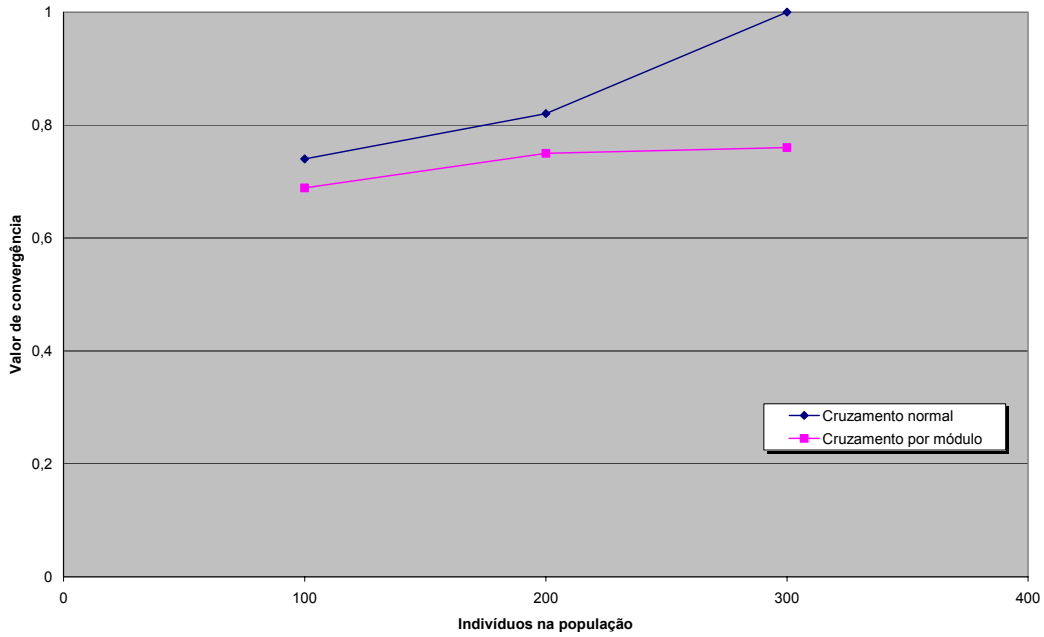


Figura 4.38 – Variação do valor de convergência em função do tamanho da população: cruzamento normal e cruzamento por módulo.

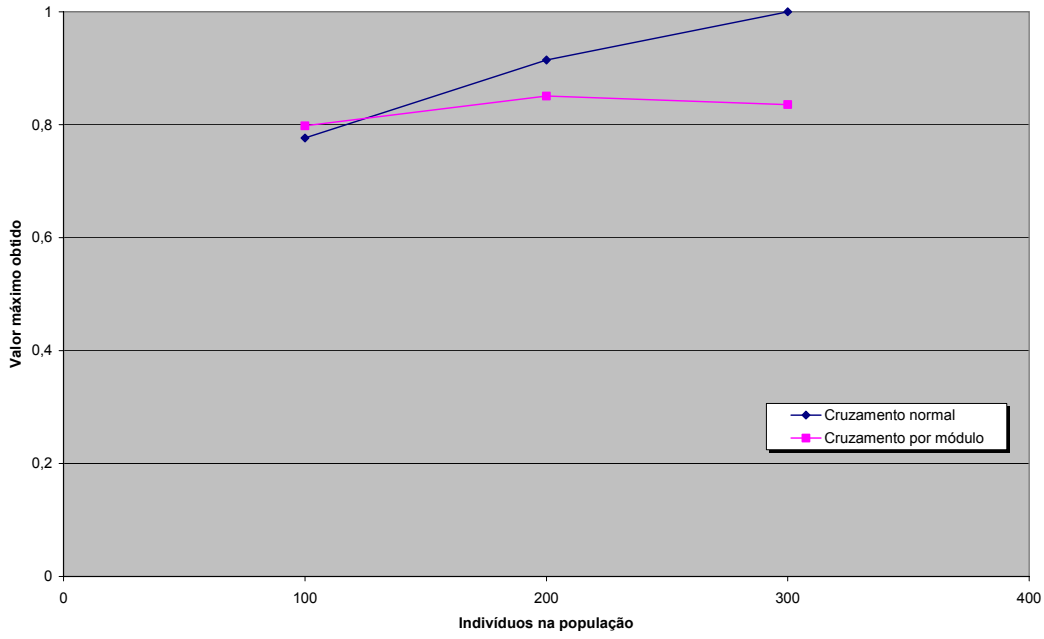


Figura 4.39 - Variação do valor máximo obtido em função do tamanho da população: cruzamento normal e cruzamento por módulo.

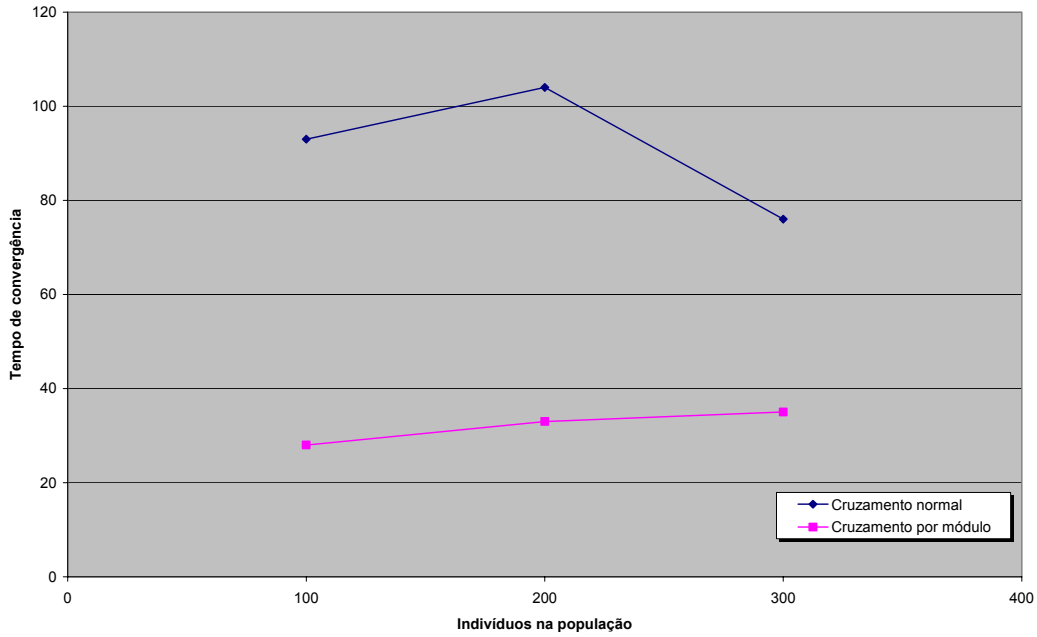


Figura 4.40 - Variação do tempo de convergência em função do tamanho da população: cruzamento normal e cruzamento por módulo.

4.9 - Aplicação usando imagens em níveis de cinza

Foram realizados alguns testes com imagens em níveis de cinza para verificar a extensibilidade do método. Foram usados subconjuntos definindo elementos estruturantes planares, ou seja, definidos por 0's e 1's, como anteriormente. A função objetivo utilizada nestes testes com imagens em níveis de cinza foi obtida através do cálculo do erro dos mínimos quadrados entre as imagens Y e $\hat{Y}$.

Foi obtido um resultado muito promissor para o ruído sal-e-pimenta, conforme mostra a Figura 4.41, para X, Y e $\hat{Y}$ em níveis de cinza.

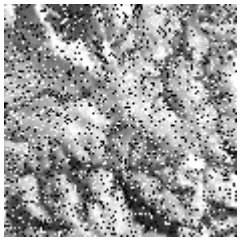
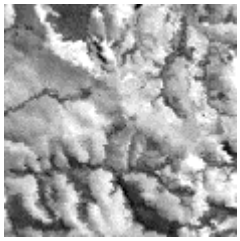
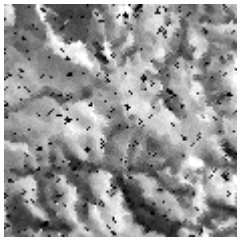
a  **b**  **c** 

Figura 4.41 - Decomposição do filtro de ruído sal-e-pimenta para imagens em níveis de cinza: **a**) X, **b**) Y, **c**) $\hat{Y}$.

Nos testes realizados a seguir foi utilizada uma imagem de satélite em níveis de cinza que mostra uma cidade com grande quantidade de detalhes. Na Figura 4.42 é apresentada a imagem original, e a Figura 4.43 mostra a mesma imagem borrada por um filtro de média de janela $\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$.

Figura 4.42 – Imagem SPOT original: cidade de Manaus.

Figura 4.43 – Imagem borrada por um filtro de média.

Inicialmente é realizada a decomposição do filtro de média que foi usado para borrar a imagem original. Para isto serão usadas duas amostras extraídas das imagens dadas. Essas amostras, assim como o resultado obtido, estão mostrados na Figura 4.44. Notar que as imagens de amostra contêm pontos claros sobre uma região escurecida.

É interessante ressaltar que, dependendo das características que aparecem nas imagens de amostra, os resultados obtidos podem ser bem diferentes entre si. Isto é ilustrado através da resolução do mesmo problema usando-se imagens de amostra de outra região, sem pontos claros nas imagens. O resultado está mostrado na Figura 4.45.

a  **b**  **c** 

Figura 4.44 – Imagens de amostra e resultado da decomposição do filtro de média: caso 1.

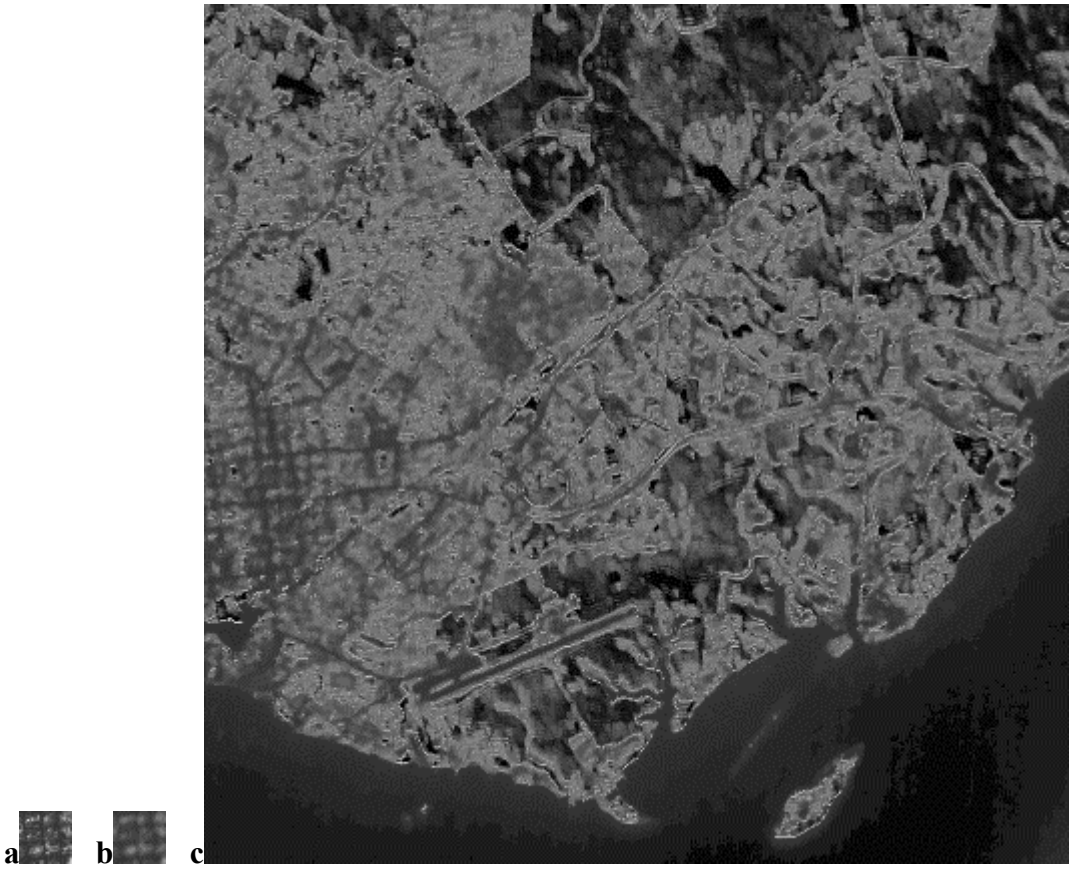


Figura 4.45 – Imagens de amostra e resultado da decomposição do filtro de média: caso 2.

Foi também realizado um teste, muito mais difícil, no sentido contrário: tentar decompor um operador que filtra uma imagem borrada, restaurando a nitidez original da imagem. As imagens de amostra utilizadas, bem como o resultado obtido, são mostrados pela Figura 4.46.

a  **b**  **c** 

Figura 4.46 – Imagens de amostra e resultado da decomposição do filtro de restauração da nitidez.

Outros testes em imagens em níveis de cinza, que foram bem sucedidos, são mostrados para o caso da dilatação, na Figura 4.47, e para o caso do extrator de bordas, na Figura 4.48.

a  **b**  **c** 

Figura 4.47 – Imagens de amostra e resultado da decomposição da dilatação.

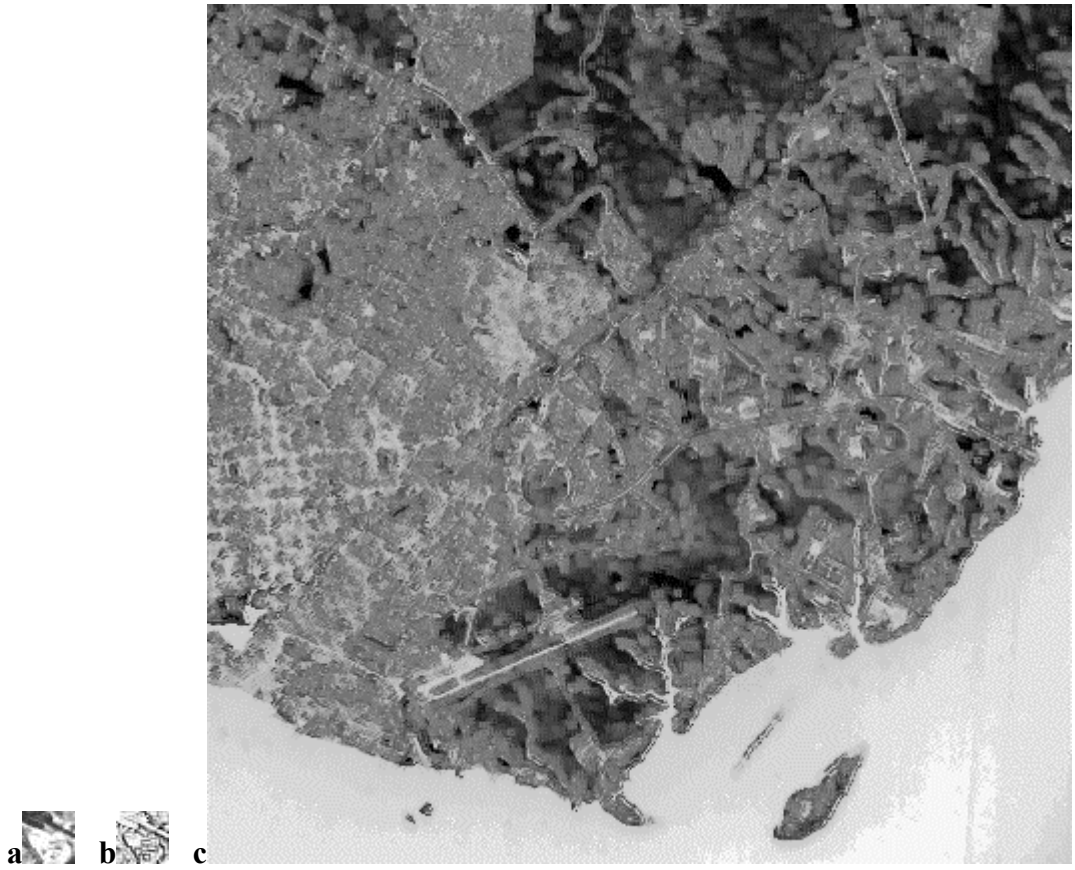


Figura 4.48 – Imagens de amostra e resultado da decomposição do extrator de bordas.

Conclui-se que, mesmo utilizando-se de elementos estruturantes planares, o método funciona bem para alguns casos de imagens em níveis de cinza. Para os casos em que o método não apresentou resultados satisfatórios, existem boas possibilidades de melhora no desempenho do método com a utilização de elementos estruturantes não-planares (em geral definidos por valores entre 0 e 255), procedimento não desenvolvido neste trabalho.

4.10 - Limitações no uso do método

Quando as imagens X' e Y' representam operadores definidos por janelas maiores que 3x3, o uso do método utilizado neste trabalho não funciona pois seria necessário utilizar-se de subconjuntos maiores para definir os intervalos fechados, o que não foi implementado nesse desenvolvimento. Ainda que fossem utilizados tais subconjuntos, o

problema poderia ser resolvido com o auxílio de um número maior de sup-geradores, mas eventualmente só seria útil para transformar X' em $\hat{Y}'$, e não serviria para nenhuma outra imagem, limitando seriamente a possibilidade de seu uso em aplicações úteis.

Para ilustrar o comportamento do método em um destes casos, a decomposição do operador SKIZ[2] será experimentada com o método proposto. A Figura 4.49 mostra as imagens X, Y e $\hat{Y}$ para este caso, a Figura 4.50 os intervalos fechados correspondentes.

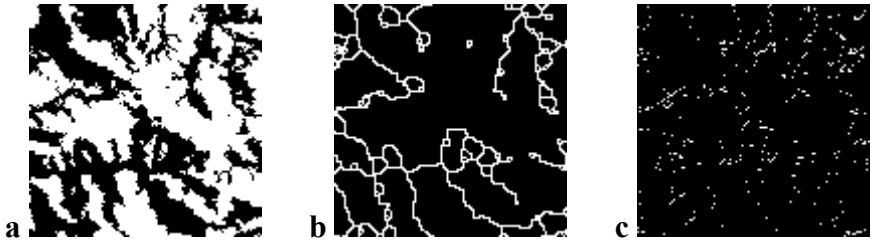


Figura 4.49 - Decomposição do operador SKIZ: **a**) X, **b**) Y, **c**) $\hat{Y}$.

$$\mathbf{a}\begin{bmatrix} \cdot & 1 & \cdot \\ \cdot & \bullet & 0 \\ 0 & \cdot & 0 \end{bmatrix} \quad \mathbf{b}\begin{bmatrix} 0 & 0 & 1 \\ 1 & \bullet & \cdot \\ \cdot & \cdot & 0 \end{bmatrix} \quad \mathbf{c}\begin{bmatrix} 1 & \cdot & \cdot \\ \cdot & \mathbf{0} & 0 \\ \cdot & 0 & \cdot \end{bmatrix} \quad \mathbf{d}\begin{bmatrix} \cdot & 1 & \cdot \\ \cdot & \mathbf{0} & 1 \\ 0 & 1 & \cdot \end{bmatrix}$$

Figura 4.50 - Intervalos fechados que parametrizam os 4 sup-geradores para o operador SKIZ

É interessante notar que o desempenho do método neste caso é tão deficitário que a melhor solução encontrada só se utilizou de 4 sup-geradores

A figura 4.51 ilustra como o GA se comporta nesse caso em que o método é impossibilitado de fornecer uma resposta razoável.

[2] O operador SKIZ define uma partição na imagem de saída de forma que, a cada componente conexa da imagem de entrada, corresponda a um gomo na partição da imagem de saída, chamado de *zona de influência da componente conexa* (Banon e Barrera, 1994, 1998).

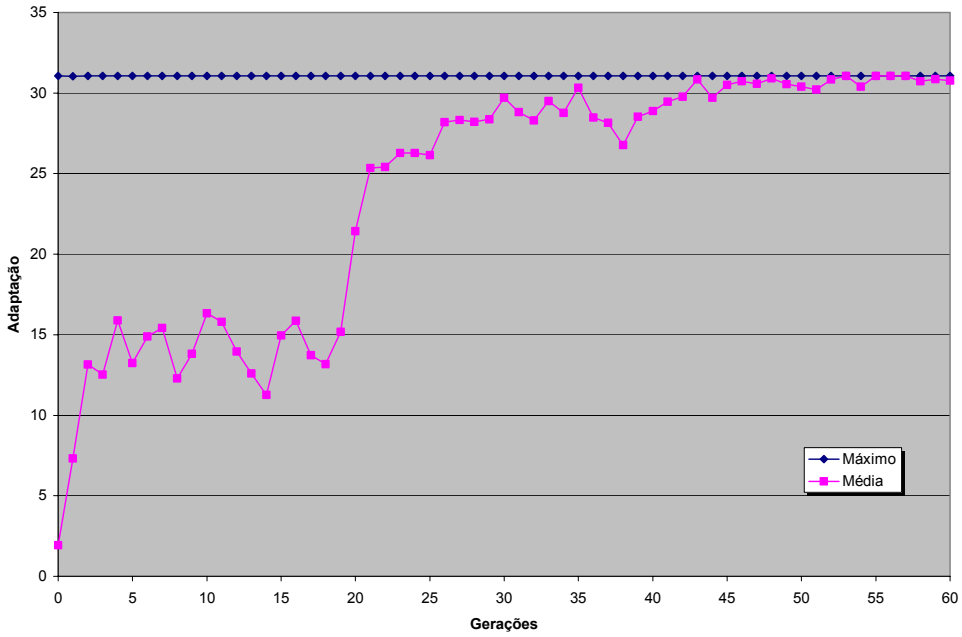


Figura 4.51 - Evolução da adaptação ao longo das gerações para a decomposição do operador SKIZ: melhor indivíduo e média da população.

Notar que desde a população inicial, que é gerada de forma aleatória, a "solução" já está disponível. Não há evolução do melhor indivíduo da população ao longo das gerações, e a evolução da média tende a transformar toda a população em clones do "melhor" indivíduo. O valor relativamente alto (88,622%) para a porcentagem de pixels idênticos de Y e $\hat{Y}$ ocorre devido ao grande número de pixels apagados (pixels pretos) que elas possuem, apresentando relativamente poucos detalhes. Como exemplo, se $\hat{Y}$ fosse uma imagem completamente negra, ao ser comparada com Y ela teria uma porcentagem de pixels idênticos igual a 89,120%, melhor, portanto, que a solução encontrada pelo GA.

CAPÍTULO 5

COMENTÁRIOS FINAIS

Este trabalho procurou alcançar dois propósitos: estabelecer uma forma de implementação prática da decomposição morfológica de operadores i.t. pelo supremo de sup-geradores, e desenvolver o potencial de uso de um GA em outros problemas que o seu uso pareça promissor.

O método apresentado neste trabalho mostrou-se viável, com excelentes perspectivas de obtenção de bons resultados na decomposição de outros operadores i.t. não abordados aqui. Esta linha de pesquisa indica possibilidades recompensadoras para um estudo continuado no assunto. As limitações encontradas não comprometem o uso prático do método em vários casos. É interessante notar que o método apresentado pode ser utilizado em qualquer tipo de imagens digitais, não apenas em imagens de satélites. Vale ainda ressaltar que o número utilizado de módulos sup-geradores (8) não comprometeu os resultados dos casos abordados. Isto ocorreu porque nesses casos as soluções teóricas procuradas não devem necessitar de um número muito maior que 8 sup-geradores para definir os respectivos operadores.

O espaço de soluções para 8 módulos sup-geradores parametrizados por intervalos fechados definidos por janelas 3x3 tem tamanho $\frac{(3^9)!}{(3^9-8)!\ 8!} \cong 5{,}58\text{x}10^{29}$. Como a codificação utilizada não impede a ocorrência de módulos sup-geradores idênticos, este número cresce para $(3^9)^8 = 3^{72} \cong 2{,}253\text{x}10^{34}$. Para representar o espaço de soluções em cadeias binárias, um GA aceita as redundâncias que forem necessárias. No presente trabalho esta acomodação de redundância acarretou em uma cadeia binária com 144 dígitos, resultando num espaço de soluções virtualmente aumentado para $2^{144} \cong 2{,}23\text{x}10^{43}$. No entanto, para a grande maioria dos testes realizados neste trabalho, com 60 gerações de populações com 90 indivíduos, um total de 60x90 = 5400 soluções foram investigadas para a aceitação de uma solução, ou seja, o GA necessitou examinar apenas uma parcela ínfima do espaço de soluções para encontrar boas soluções. Mesmo no caso em que foi

necessário um maior esforço para se obter a solução exata, que foi o caso da abertura morfológica, foi necessário o exame de, aproximadamente, 24000 soluções, um número (ainda) irrisório se comparado a 10^{34} .

O GA mostrou a sua força ao encontrar boas soluções de forma eficiente e robusta. Outras aplicações deverão contemplar o seu uso visando soluções mais rápidas e mais precisas. Constatou-se que é possível melhorar o resultado aumentando-se a população e o número de gerações. Observou-se também que uma probabilidade de mutação muito alta não favorece o GA. Há ainda bons indícios que a utilização de um esquema de cruzamento parâmetro a parâmetro converge mais rapidamente para soluções sub-ótimas. Uma outra forma de melhorar o desempenho do GA seria excluir das soluções encontradas os sup-geradores que não colaboram na melhoria da adaptação e, ao invés, acabam por introduzir ruído nas soluções.

Uma continuação natural deste trabalho seria a investigação comparativa da decomposição de operadores i.t. pelo ínfimo de inf-geradores, assim como um estudo mais aprofundado no caso de imagens em níveis de cinza utilizando-se de elementos estruturantes não-planares, que aceitam valores entre 0 e 255, e não apenas 0 e 1 como no caso planar. Outras possibilidades de trabalho seriam: investigar o desempenho do método para operadores que necessitem de mais de oito módulos; explorar o desempenho do método para elementos estruturantes maiores que os de tamanho 3x3 utilizados neste trabalho, propiciando a decomposição de operadores de vizinhança maior que os operadores aqui estudados; investigar qual a influência, no método, do tamanho e do conteúdo das imagens de amostra; pesquisar as limitações do algoritmo; comparar os diversos métodos encontrados na literatura, incluindo este, usando-se um mesmo computador. Uma outra possibilidade ainda seria a utilização do método aqui apresentado dentro de um processo de produção.

REFERÊNCIAS BIBLIOGRÁFICAS

Alliot, J. M.; Gruber, H.; Joly, G., Schoenauer, M. Genetic algorithms for solving air traffic control conflicts. In: Conference on Artificial Intelligence for Applications, 9, 1993. **Proceedings**. p 338-344, 1993.

Altenberg, L. Evolving better representations through selective genome growth. In: Conference on Evolutionary Computation, 1.,1994. **Proceedings**. IEEE v.1, p 182-187, 1994.

Androulakis, I. P.; Venkatasubramanian, V. Genetic algorithmic framework for process design and optimization. **Computers & Chemical Engineering**, v.15, n. 4, p 217-228, 1991.

Ansari, N.; Chen, M. H.; Hou, E. S. H. Point pattern matching by a genetic algorithm. In:Annual Conference of IEEE Industrial Electronics Society - IECON'90, 16.,1990. **Proceedings.** IEEE, v.2, p.1233-1238, 1991.

Arabas, J.; Michalewicz, Z.; Mulawka, J. GAVaPS - A genetic algorithm with varying population size. In:IEEE Conference on Evolutionary Computation, 1.,1994. **Proceedings.** IEEE, v.1, p. 73-78, 1994.

Baeck, T.; Khuri, S. Evolutionary heuristic for the maximum independent set problem. In: IEEE Conference on Evolutionary Computation, 1., 1994. **Proceedings**. IEEE, v.2, p.531-535, 1994.

Bala, J.; Wechsler, H. Shape analysis using morphological processing and genetic algorithms. In:International Conference on Tools for Artificial Intelligence, 3., 1991, San Jose. **Proceedings**. San Jose, p.130-137, 1992.

Ball, N. R.; Sargent, P.M.; Ige, D.O. **Genetic algorithm representations for laminate layups**. Artificial Intelligence inEngineering, v. 8, n. 2, p. 99-108, 1993.

Banon, G. J. F.; Barrera, J. Minimal representations for translational-invariant set mappings by mathematical morphology. **SIAM J.Applied Mathematics.**, v. 51, n.6, p. 1782-1798, 1991.

Banon, G. J. F.; Barrera, J. Decomposition of mappings between complete lattices by mathematical morphology - Part I: General lattices. **Signal Processing**, v. 30, p. 299-327, 1993.

Banon, G. J. F.; Barrera, J. Bases da morfologia matemática para a análise de imagens binárias. In: Escola de Computação, 9., Receife, 1994. **Proccedings.** Recife: UFPE, 1994.

Banon, G. J. F. **Set and function operator decomposition**. [Online] Repositório da URLib: <dpi.inpe.br/banon/1996/02.12.16.13> 1996.

Banon, G. J. F.; Barrera, J. **Bases da morfologia matemática para a análise de imagens binárias**. 2. ed.[Online]. Repositório da URLib: <dpi.inpe.br/banon/1998/06.30.17.56>. 1998.

Barrera, J.; Silva, F. S. C.; Banon, G. J. F. **Automatic programming of binary morphological machines**. Relatório Técnico RT-MAC-9405, IME/USP, [Online]. Repositório da URLib: <ime.usp.br/jb/1996/04.03.20.39>. 1994.

Barrera, J.; Tomita, N. S.; Silva, F. S. C.; Terada, R. Automatic programming of binary morphological machines by PAC learning. [Online]. In: , Neural, Morphological and Stochastic Methods in Image and Signal Processing, San Diego, 1995. **Proceedings.** San Diego: SPIE, v. 2568, p. 233-244, 1995. Repositório da URLib: <ime.usp.br/jb/1996/04.03.20.49>.

Barrera, J.; Hashimoto, R. F. Compact representation of *W*-operators. In: Nonlinear Image Processing, 9., San Jose,1998. **Proceedings.** San Jose: SPIE, v. 3304, p. 84-94, 1998.

Baumgartner, J. P.; Cook, D. J. Genetic-based solution to load balancing in parallel computers. In: Annual ACM Computer Science Conference, 22., Phonix, 1994. **Proceedings.** Phonix: ACM, p. 157-164, 1994.

Bayer, S. E.; Wang, L. A genetic algorithm programming environment: Splicer. In: International Conference on Tools for Artificial Intelligence, 3., San Jose, 1991. **Proceedings**. San Jose: p. 138-144, 1992.

Bellgard, M. I.; Tsang, C. P. Some experiments on the use of genetic algorithms in a Boltzmann machine. In: IEEE International Joint Conference on Neural Networks, Piscataway, 1991. **Proceedings.** p. 2645-2652, 1991.

Bezdek, J. C.; Hathaway, R. J. Optimization of fuzzy clustering criteria using genetic algorithms. In: IEEE Conference on Evolutionary Computation, 1., 1994. **Proceedings.** v. 2, p. 589-594, 1994.

Bhandari, D.; Pal, S. K.; Kundu, M. K. Image enhancement incorporating fuzzy fitness function in genetic algorithms. In: IEEE International Conference on Fuzzy Systems, 2., 1993. **Proceedings**. p. 1408-1413, 1993.

Bhandari, D.; Pal, N. R.; Pal, S. K. Directed mutation in genetic algorithms. **Information Sciences**, v. 79, n. 3-4, p. 251-270, 1994.

Bhandarkar, S. M.; Zhang, Y.; Potter, W. D. Edge detection technique using genetic algorithm-based optimization. **Pattern Recognition**, v. 27, n. 9, 1994.

Bickel, A. S.; Bickel, R. W. Determination of near- optimun use of hospital diagnostic resources using the ' genes' genetic algorithm shell. **Computers in Biology and Medicine**, v. 20, n. 1, p. 1-13,1990.

Cartwright, H. M.; Harris, S. P. Analysis of the distribution of airborne pollution using genetic algorithms. **Atmospheric Environment, Part A**: General Topics, v. 27A, n.12, p. 1783-1791, 1993.

Chakraborty, U. K.; Dastidar, D. G. Using reliability analysis to estimate the number of generations to convergence in genetic algorithms. **Information Processing Letters,** v.46, n. 4, p. 199-209, 1993.

Clark, D. E.; Jones, G.; Willett, P.; Kenny, P. W.; Glen, R. C. Pharmacophoric pattern matching in files of three-dimensional chemical structures: Comparison of conformational-searching algorithms for flexible searching. **Journal of Chemical Information and Computer Sciences,** v. 34, n. 1, p. 197-206, 1994.

Chu, C. H. H.; Kottapalli, M. S. Genetic algorithm approach to visual model-based halftone pattern design. In: Visual Comunications and Image Processings' 91 – **The International Society for Optical Engeneering,** 1991, p. 470-481. (SPIE Proceedings, 1606).

Cohoon, J. P.; Hegde, S. U.; Martin, W. N.; Richards, D. Floorplan design using distributed genetic algorithms. In: IEEE International Conference on Computer-Aided Design – **Digest of Technical Papers.** p. 452-455. (ICCAD-1988).

Corcoran, A. L. III; Wainwright, R. L. Genetic algorithm for packing in three dimensions. In: ACM/SIGAPP Symposium on Applied Computing SAC'92, 1992. **Proceedings.** ACM/SIGAPP, p.1021-1030, 1992. (Applied Computing: Technological Challenges of the 1990's Proceedings).

Cui, J.; Fogarty, T. C.; Terence, C.; Gammack, J. G. Searching databases using parallel genetic algorithms on a transputer computing surface. **Future Generation Computer Systems**, v.9, n.1, p.33-40, 1993.

Dandy, G. C.; Simpson, A. R.; Murphy, L. J. Review of pipe network optimization techniques. In: Australasian Conference on Computing for the Water Industry Today and Tomorrow, 2., 1993. **Proceedings**. p.373-383, 1993.

Dasgupta, D. Handling deceptive problems using a different genetic search. In: IEEE Conference on Evolutionary Computation, 1., 1994. **Proceedings.** IEEE**,** v.2, p. 807-811, 1994.

Dasgupta, D.; Mc Gregor, D. R. **Digital image registration using structured genetic algorithm** - The International Society for Optical Engineering, 1., 1992, p. 226-234. .(Proceedings of SPIE, v. 1766).

Dasgupta, D.; Mc Gregor, D. R. More biologically motivated genetic algorithm: the model and some results. **Cybernetics and Systems**, v. 25, n. 3, p. 447-469, 1994.

Davidor, Y. Robot programming with a genetic algorithm. In: IEEE International Conference on Computer Systems and Software Engineering - COMPEURO 90, 1990. **Proceedings.** IEEE**,** p.186-191, 1990.

Deb, K. Optimal design of a class of welded structures via genetic algorithms. In: - AIAA/ASME/ASCE/AHS Structures, Structural Dynamics & Materials Conference, 31., Long Beach, 1990.. **Proceedings.** Washington: NASA, v. 1, p. 444-453, 1990. Collection of Technical Papers.(NASA CP, 3064).

Dorigo, M. Using transputers to increase speed and flexibility of genetics-based machine learning systems. **Microprocessing and Microprogramming**, v.34, n.1, p.147-152, 1992.

Dougherty, E. R..; Mathew, A.; Swarnakar, V. **A conditional-expectation-based implementation of the optimal mean-square binary morphological filter**.,127-147, 1991. (Proceedings of the SPIE, 1451).

Dougherty, E. R..; Haralick, R. M**. The hole spectrum-model-based optimization of morphological filters**., 1568, 233-246, 1991. (Proceedings of the SPIE).

Dougherty, E. R..; Loce, R.P. Facilitation of optimal morphological filter design via structuring element libraries and design constraints. SPIE Optical Engineering, v.31, n.5, 1992.

Elketroussi, M.; Fan, D. P. Optimization of simulation models with GADELO: a multi-population genetic algorithm. **International Journal of Biomedical Computing**, v.35, n. 1, p. 61-77, 1994.

Esbensen, H.; Mazumder, P. SAGA: a unification of the genetic algorithm with simulated annealing and its application to macro-cell placement. In: International Conference on VLSI Design, 7., 1994. **Proceedings.** p. 211-214, 1994.

Falkenauer, E.; Delchambre, A. A genetic algorithm for bin packing and line balancing. Proceedings. In: IEEE International Conference on Robotics and Automation. 1992. **Proceedings**.v. 2, p. 1186-1192, 1992.

Fang, J. H.; Karr, C. L.; Stanley, D. A. Genetic algorithm and its application to petrophysics. **Society of Petroleum Engineers of AIME**, p.1-12, 1993.

Forrest, S.; Mitchell, M. What makes a problem hard for a genetic algorithm? Some anomalous results and their explanation. **Machine Learning**, v. 13, n. 2-3, p. 285-319, 1993.

Fukuda, T.; Hasegawa, Y.; Shimojima, K. Hierarchical fuzzy reasoning - adaptive structure and rule by genetic algorithms. In: IEEE Conference on Evolutionary Computation, 1., 1994. **Proceedings.** IEEE, v. 2, p. 601-606, 1994.

de Garis, H. Genetic programming: Building nanobrains with genetically programmed neural network modules. In; International Joint Conference on Neural Networks - IJCNN'90, 1990. **Proceedings.** p. 511-516, 1990.

de Garis, H. GenNets: Genetically programmed neural nets - using the genetic algorithm to train neural nets whose inputs and/or outputs vary in time. In: IEEE International Joint Conference on Neural Networks - IJCNN'91, 1991. **Proceedings.** IEEE, p.1391-1396, 1991.

Goldberg, D. E. **Genetic algorithms in search, optimization, and machine learning**. New York: Addison-Wesley, 1989. 412p.

Hadi, M. A.; Wallace, C.E. Hybrid genetic algorithm to optimize signal phasing and timing. **Transportation Research Record**,n. 1421, p.104-112, 1993.

Harp, S. A.; Samad, T. Optimizing neural networks with genetic algorithms. In: American Power Conference, 1992. **Proceedings.** v.54, p.1138-1143, 1992.

Hartley, S. J.; Konstam, A. H. Using genetic algorithms to generate Steiner triple systems. In: Annual ACM Computer Science Conference, 21., 1993. **Proceedings.** ACM, p. 366-371, 1993.

Harvey, N. R.; Marshall, S. Using genetic algorithm in the design of morphological filters. **Mathematical Morphology and Its Applications to Image Processing**, p. 53-59, 1994.

Hintz, K. J.; Spofford, J. J. Evolving a neural network. In: IEEE International Symposium on Intelligent Control, 5., 1990. **Proceedings.** IEEE, p. 479-484, 1990.

Holland, J. H. **Adaptation in natural and artificial systems**. Ann Arbor: The University of Michigan Press, 1975.

Homaifar, A.; Qi, C. X.; Lai, S. H. Constrained optimization via genetic algorithms. **Simulation**, v.62, n. 4, p. 242-254, 1994.

Hoptroff, R. G.; Hall, T. J.; Burge, R. E. Experiments with a neural controller. In: International Joint Conference on Neural Networks 90 - IJCNN 90, **Proceedings.** p. 735-740, 1990.

Horng, J. T.; Kao, C. Y.; Liu, B. J. Genetic algorithm for database query optimization. In: IEEE Conference on Evolutionary Computation, 1., 1994. **Proceedings.** IEEE, v.1, p.350-355, 1994.

Huang, W.; Bian, Z. GA approach to invariant matching: Under noise and geometric transformations. In: IEEE Region 10 Conference on Computer, Communication,

Control and Power Engineering (TECON'93), 1993. **Proceedings.** IEEE, v.2, p.1021-1024, 1993.

Hull, R. A.; Johnson, R. W**.** Performance enhancement of a missile autopilot via genetic algorithm optimization techniques. In: American Control Conference, 1994. **Proceedings.** v. 2, p.1680-1684, 1994.

Huntley, C. L.; Brown, D. E. Parallel heuristic for quadratic assignment problems. **Computers & Operations Research**, v.18, n.3, p. 275-289, 1991.

Husbands, P. Distributed coevolutionary genetic algorithms for multi-criteria and multi-constraint optimisation. **Evolutionary Computing, Lecture Note in Computer Science,** v. 865, p.150-165, 1994.

Idlebi, N.; Mignot, B**.** GAME parallel support strategies for the parallelization of genetic algorithms. In: Computing and Control Division Colloquium on Applications of Genetic Algorithms, 1994. **IEE Colloquium (Digest).** IEE, n. 67, p. 4/1-4/4, 1994.

Ikuno, Y.; Kawabata, H.; Shirao, Y.; Hirata, M.; Nagahara, T.; Iagaki, Y. Application of an improved genetic algorithm to the learning of neural networks. **Solid State Communication,** v. 91, n. 3, p. 731-735, 1994.

Jacq, J. J.; Roux, C. Automatic registratrion of 3D images using a simple genetic algorithm with a stochastic performance function. In; Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 15., 1993. **Proceedings.** IEEE, .p 126-127, 1993a.

Jacq, J. J.; Roux, C. Registration of successive DSA images using a simple genetic algorithm with a stochastic performance function. In: IEEE Annual Northeast Bioengineering Conference, 19., 1993. **Proceedings.** IEEE, p. 223-224, 1993b.

Joines, J. A.; Houck, C. R. On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with GA's. In: IEEE Conference on Evolutionary Computation, 1., 1994. **Proceedings.** v.2, p. 579-584, 1994.

Juric, M. Anti-adaptationist approach to genetic algorithms. of the 1st IEEE Conference on Evolutionary Computation, **Proceedings.** IEEE, v. 2, p. 619-623, 1994.

Kanet, J. J.; Sridharan, V. PROGENITOR. A genetic algorithm for production scheduling. **Wirtschaftsinformatik**, v. 33, n. 4, p. 332-336, 1991.

Karr, C. L.; Goldberg, D. E. Genetic algorithm based design of an air-injected hydrocyclone. **Control 90 Miner Metall Process Minerals & Metallurgical Processing**, p. 265-272,1990.

Khuri, S. Informatic crossover in genetic algorithms. In: IEEE International Symposium on Information Theory, 1990. **Proceedings**. IEEE, p. 62, 1990.

Kim, H. Y. Construção automática de operadores morfológicos por aprendizagem computacional. São Paulo, 191 p. Tese (Doutorado em Engenharia, Escola Politécnica da USP), USP,1997.

Kim, J.; Moon, Y.; Zeigler, B. P. Designing fuzzy net controllers using GA optimization. In: IEEE/IFAC Joint Symposium on Computer-Aided Control System Design, **Proceedings.** IEEE, p. 83-88, 1994.

Kingdon, J.; Dekker, L. Development needs for diverse genetic algorithm design. In: Computing and Control Division Colloquium on Applications of Genetic Algorithms, 19994. **IEE Colloquium (Digest)**. IEE, n. 67, p. 3/1-3/11, 1994.

Kinzel, J.; Klawonn, F.; Kruse, R. Modifications of genetic algorithms for designing and optimizing fuzzy controllers. of the 1st IEEE Conference on Evolutionary Computation, **Proceedings.** IEEE, v.1, p. 28-33, 1994.

Konstam, A. H.; Hartley, S. J.; Carr, W. L. Optimization in a distributed processing environment using genetic algorithms with multivariate crossover. In: Annual ACM Computer Science Conference - CSC'92, 20., 1992. **Proceedings. ACM,** p. 109-114, 1992.

Koza, J. R.; Rice, J. P. Genetic generation of both the weights and architecture for a neural network. In: International Joint Conference on Neural Networks - IJCNN'91, 1992. **Proceedings.** p. 397-404, 1992.

Kreinovich, V.; Quintana, C.; Fuentes, O. **Genetic** algorithms. What fitness scaling is optimal?. **Cybernetics and Systems,** v. 24, n.1, p. 9-26, 1993.

Kwok, D. P.; Sheng, F**.** Genetic algorithm and simulated annealing for optimal robot arm PID control. In:IEEE Conference on Evolutionary Computation, 1., 1994. **Proceedings.** IEEE, v.2, p. 707-712, 1994.

Le Riche, R.; Haftka, R. T. Optimization of laminate stacking sequence for buckling load maximization by generic algorithm. **AIAA Journal**, v. 31, n.. 5, p. 951-956, 1993.

Lee, M. A.; Takagi, H. Integrating design stages of fuzzy systems using genetic algorithms. In: IEEE International Conference on Fuzzy Systems, 2., 1993. **Proceedings.** IEEE, p. 612-617, 1993.

Lin, J. L.; Foote, B.; Pulat, S.; Chang, C. H.; Cheung, J. Y. Hybrid genetic algorithm for container packing in three dimensions. In: Conference on Artificial Intelligence for Applications, 9., 1993. **Proceedings.** p. 353-359, 1993.

Lin, C. Y.; Hajela, P. Genetic search strategies in large scale optimization. In: AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, 1993. **Proceedings.** AIAA/ASME/ASCE/AHS/ASC, p. 2437-2447, 1993.

Logar, A. M.;Corwin, E. M.; English, T. M. Implementation of massively parallel genetic algorithms on the MasPar MP-1. In: ACM/SIGAPP Symposium on Applied Computing SAC'92, 1992. **Proceedings.** ACM/SIGAPP, p. 1015-1020, 1992.

Lotufo, R. A. MMachLib functions and MMach operators. In: Brazilian Workshop'97 on Mathematical Morphology, 1997. **Mini-curso**. 1997.

Mandava, V. R.; Fitzpatrick, J. M.; Pickens, D. R. Adaptive search space scaling in digital image registration. **IEEE Transactions on Medical Imaging**, v. 8, n. 3, p. 251-262, 1989.

Mansour, N; Fox, G. C. Allocating data to distributed-memory multiprocessors by genetic algorithms. **Concurrency Practice and Experience**, v. 6, n. 6, p. 485-504, 1994.

Maragos, P. A. **A unified theory of translation-invariant systems with applications to morphological analysis and coding of images**. Atlanta, 240p. Tese (Doutorado, School of Elec. Eng.), Georgia Inst. Tech., 1985.

Martinez, E. R.; Moreno, W. J.; Castilho V. J.; Moreno, J. A. Rod pumping expert system. **Proceedings Petroleum Computer Conference**, p. 201-208, 1993.

Martinez, E. R.; Moreno, W. J.; Moreno, J. A.; Maggiolo, R**.** Application of genetic algorithm on the distribution of gas lift injection. In: Latin American and Caribbean Petroleum Engineering Conference, 3., 1994. **Proceedings.** v. 2, p. 811-818, 1994.

Matheron, G. **Éléments pour une théorie des milieux poreux**. Paris: Masson & Cie., 1967.

Mc Aulay, A. D.; Oh, J. C. Image learning classifier system using genetic algorithms. In: IEEE Proceedings of the National Aerospace and Electronics Conference, 1989. **Proceedings.** IEEE, v. 2, p. 705-710,1989.

Mc Intyre, R. A. Bach in a box: The evolution of four part baroque harmony using the genetic algorithm. In: IEEE Conference on Evolutionary Computation, 1., 1994. **Proceedings.** IEEE, v. 2, p. 852-857, 1994.

Miller, J. A.; Potter, W. D.; Gandham, R. V.; Lapena, C. N. Evaluation of local improvement operators for genetic algorithms. **IEEE Transactions on Systems, Man and Cybernetics**, v. 23, n. 5, p. 1340-1351, 1993.

Moed, M. C.; Stewart, C. V.; Kelly, R. B. Reducing the search time of a steady state genetic algorithm using the immigration operator. In: International Conference on Tools for Artificial Intelligence, 3., 1992. **Proceedings**. p. 500-501, 1992.

Mohammadian, M.; Stonier, R. J. Generating fuzzy rules by genetic algorithms. In: IEEE International Workshop on Robot and Human Communication, 3., 1994. **Proceedings.** IEEE, p. 362-267,1994a.

Mohammadian, M.; Stonier, R. J. Tuning and optimization of membership functions of fuzzy logic controllers by genetic algorithms. of the 3rd IEEE International Workshop on Robot and Human Communication, 1994. **Proceedings.** p. 356-361,1994b.

Muehlenbein, H.; Gorges-Schleuter, M.; Kraemer, O. Evolution algorithms in combinatorial optimization. **Parallel Computing**, v. 7, n. 1, p. 65-85, 1988.

Muehlenbein, H.; Schomisch, M.; Born, J. Parallel genetic algorithm as function optimizer. **Parallel Computing**, v. 17, n. 6-7, p. 619-632, 1991.

Murdock, T. M.; Schmitendorf, W. E.; Forrest, S. Use of a genetic algorithm to analyze robust stability problems. of the American Control Conference, 1991. **Proceedings.** v. 1, p. 886-889, 1991.

Mutalik, P. P.; Knight, L. R.; Blanton, J. L.; Wainwirght, R. L. Solving combinatorial optimization problems using parallel simulated annealing and parallel genetic algorithms. Applied Computing: Technological Challenges of the 1990's. In: ACM/SIGAPP Symposium Applied Computing SAC 92, 1992. **Proceedings…** ACM/SIGAPP, 1992, p. 1031-1038.

Oliveira, J. R. F. Using genetic algorithms in image classification. Brazilian. In: Workshop'96 on Mathematical Morphology, 1996. Repositório da URLib: <dpi.inpe.br/joao/1996/02.09.16.24>.

Orvosh, D.; Davis, L. Using a genetic algorithm to optimize problems with feasibility constraints. In: IEEE Conference on Evolutionary Computation, 1., 1994. **Proceedings…**IEEE**,** v. 2, p. 548-553,1994.

Pal, S. K. **Fuzzy sets in image processing and recognition**. In: IEEE International Conference on Fuzzy Systems - FUZZ-IEEE, 1992. **Proceedings…** IEEE, p. 119-126, 1992.

Pal, S. K.; Bhandari, D.; Kundu, M. L. Genetic algorithms for optimal image enhancement. **Pattern Recognition Letters**, v. 15, n. 3, p. 261-271, 1994.

Park, J. M.; Park, J. G.; Lee, C. H.; Han, M. S. Robust and efficient genetic crossover operator: homologous recombination. In: International Joint Conference on Neural Networks, 1993. **Proceedings…** v. 3, p. 2975-2978, 1993.

Patel, M. J.; Maniezzo, V. NN's and GA's: evolving co-operative behaviour in adaptive learning agents. In: IEEE Conference on Evolutionary Computation, 1., **Proceedings…** IEEE, v.1, p. 290-295, 1994.

Patton, R. J.; Liu, G. P. Robust control design via eigenstructure assignment, genetic algorithms and gradient-based optimization. In: IEEE Control Theory and Applications, 1994. **Proceedings…**IEEE**,** v.141, n. 3, p. 202-208, 1994.

Payne, A. W. R.; Glen, R. C. Molecular recognition using a binary genetic search algorithm. **Journal of Molecular Graphics**, v.11, n. 2, p. 74-91, 1993.

Prahlada, R. B. B.; Patnaik, L. M.; Hansdah, R. C. Genetic algorithm for channel routing using inter-cluster mutation. In: IEEE Conference on Evolutionary Computation, 1994. **Proceedings…** IEEE**,** v.1, p. 97-103, 1994.

Potter, W. D.; Miller, J. A.; Gandham, R.; Lapena, C. N. Improving the reliability of heuristic diagnosis. In: IEEE International Conference on Systems, Man and Cybernetics, 1991. **Proceedings…** IEEE, v. 2, p. 725-730, 1991.

Potts, J. C.; Giddens, T. D.; Yadav, S. B. Development and evaluation of an improved genetic algorithm based on migration and artificial selection. **IEEE Transactions on Systems, Man and Cybernetics,** v.24, n. 1, p. 73-86, 1994.

Rajeev, S.; Krishnamoorthy, C. S. Discrete optimization of structures using genetic algorithms. **Journal of Structural Engineering**, v. 118, n.5, p.1233-1250, 1992.

Renders, J. M.; Bersini, H. Hybridizing genetic algorithms with Hill-climbing methods for global optimization: Two possible ways. In: IEEE Conference on Evolutionary Computation, 1., 1994. . **Proceedings…**IEEE, v.1, p. 312-317, 1994.

Roth, G.; Levine, M. D. Geometric primitive extraction using a genetic algorithm. IEEE **Transactions on Pattern Analysis and Machine Intelligence**, v.16, n. 9, p. 901-905, 1994.

Sakane, S.; Kuruma, T.; Omata, T.; Sato, T. Planning focus of attention for multi-fingered hand with consideration of time-varying aspects. In: CAD-Based Vision Workshop, 2., 1994. **Proceedings…**IEEE, p.151-160, 1994.

Schultz, A. Multiple population Boltzmann machine. In: IEEE Conference on Evolutionary Computation, 1., 1994. **Proceedings…**IEEE, v.1, p. 368-373, 1994.

Seetharaman, G. S.; Prabhu, O. S.; Narasimhan, A. Two modified crossover and mutation operators for image segmentation by genetic algorithms. In: SPIE - The International Society for Optical Engineering, 1992. **Proceedings…**SPIE,v.1766, pp 66-76, 1992.

Sen, M. K.; Gupta, A. D.; Stoffa, P. L.; Lake, L. W.; Pope, G. A. Stochastic reservoir modeling using simulated annealing and genetic algorithm. In: Formation Evaluation and Reservoir Geology Proceedings - SPE Annual Technical Conference and Exhibition v Omega, Washington, DC,1992. **Proceedings…** Washington, DC: SPE, p. 939-950, 1992.

Serra, J. P. F. **Image analysis and mathematical morphology .** London: Academic Press, v.1, 1982.

Serra, J. P. F. **Image analysis and mathematical morphology.:** theoretical advances. Londos: Academic Press, v. 2, 1988.

Shahookar, K.; Mazumder, P. Genetic approach to standard cell placement using meta-genetic parameter optimization. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems**, v. 9, n. 5, p. 500-511, 1990.

Shang, Y.; Li, G. J. New crossover operators in genetic algorithms. In: International Conference on Tools for Artificial Intelligence, 3., 1992. **Proceedings…**p. 150-153, 1992.

Sharman, K. C.; Mc Clurkin, G. D. **Genetic algorithms for maximum likelihood parameter estimation**. In: ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing, 1989. **Proceedings…**IEEE,v. 4, p. 2716-2719, 1989.

Sheung, J.; Fan, A.; Tang, A. **Time tabling using genetic algorithm and simulated annealing**. In: IEEE Region Conference on Computer, Communication, Control

and Power Engineering (TENCON'93), 10., 1993. **Proceedings …** IEEE, p. 448-451, 1993.

Srinivas, M.; Patnaik, L. M. Adaptive probabilities of crossover and mutation in genetic algorithms. **IEEE Transactions on Systems, Man and Cybernetics**, v. 24, n. 4, p. 656-667, 1994.

Stoffa, P. L.; Sen, M. K. Nonlinear multiparameter optimization using genetic algorithms: inversion of plane-wave seismograms. **Geophysics,** v. 56, n. 11, p. 1794-1810, 1991.

Stuckman, B.; Evans, G.; Mollaghasemi, M. Comparison of global search methods for design optimization using simulation. In: Winter Simulation Conference Proceedings, 23., 1991, Phoenix, Arizona. **Proceedings…** Washington: IEEE, p. 937-944, 1991.

Sun,Y.; Wang, Z. Genetic algorithm for 0-1 programming with linear constraints. In: IEEE Conference on Evolutionary Computation, 1., 1994. **Proceedings …**vol. 2, pp 559-564, 1994.

Tanaka, Y.; Ishiguro, A.; Uchikawa, Y. Method of estimation of current distribution using genetic algorithms with variable-length chromosomes. **International Journal of Applied Electromagnetics in Materials**, v. 4, n. 4, p. 351-356, 1994.

Thangiah, S. R.; Nygard, K. E.; Juell, P. L. GIDEON: A genetic algorithm system for vehicle routing with time windows. In: IEEE Conference on Artificial Intelligence Applications, 7., Miami, FL, 1991,. **Proceedings…** Miami: IEEE, p.322-328, 1991.

Thangiah, S. R.; Nygard, K. E. School bus routing using genetic algorithms. In: SPIE - The International Society for Optical Engineering, 10., Orlando. P**roceedings…**Orlando**:** SPIE, p. 387-398, 1992.

Trenkle, J. M.; Schlossser, S.; Vogt, Robert C. Morphological feature-set optimization using the genetic algorithm. In: Gader, P. D.; Dougherty, E. R. (eds). **Image algebra and morphological image processing II.** The International Society for Optical Engineering, v.1568, p. 212-223, 1991. (1991SPIE.1568..212t)

Vemuri, R.; Vemuri, R. Genetic algorithms for MCM partitioning. **Electronics Letters,** v. 30, n. 16, p. 1270- 1271, 1994.

Vose, M. D. Generalizing the notion of schema in genetic algorithms. **Artificial Intelligence**, v 50, n. 3, p. 385-396, 1991.

Wen, F.; Han, Z. Fault section estimation in power systems using genetic algorithm and simulated annealing. In: Zhongguo Dianji Gongcheng Xuebao.(eds). **Proceedings of the Chinese Society of Electrical Engineering**, v.14, n.3, p.. 29-35,1994.

Wieland, A. P. Evolving neural network controllers for unstable systems. In: International Joint Conference on Neural Networks - IJCNN 91, 1992, Seattle, WA. **Proceedings…** Piscataway, NJ: IEEE Press, v. 2, p. 667-673, 1992.

Wilke, P. Visualization of neural networks using neuro graph. **IFIP Transactions A: Computer Science and Technology**, n. A-48, p. 105-117, 1994.

Williams, B. V.; Bostock, R. T. J.; Bounds, D.; Harget, A. Improving classification performance in the bumptree network by optimizing topology with a genetic algorithm. In: IEEE Conference on Evolutionary Computation, 1., Orlando **Proceedings…** Orlando: IEEE, v. 1, p. 490-495, 1994.

Yamamura, M.; Satoh, H.; Kobayashi, S. Analysis of crossover's effect in genetic algorithms. In: IEEE Conference on Evolutionary Computation, 1., Orlando. **Proceedings…** Orlando: IEEE, v. 2, p. 613-618, 1994.

Yang, G. Genetic algorithm for the optimal design of diffractive optical elements and the comparison with simulated annealing algorithm. **Guangxue Xuebao/Acta Optica Sinica**, v.13, n.7, p. 577-584, 1993.

Yang, C. H.; Nygard, K. E. Effects of initial population in genetic search for time constrained traveling salesman problem**s**. Annual Computer Science Conference, 21., 19993, Hartford. **Proceedings…** Hartford: p. 378-383, 1993.

Yao, X. Empirical study of genetic operators in genetic algorithms. **Microprocessing and Microprogramming**, v.38, n.1-5, p. 707-714, 1993.