UNIVERSIDADE ESTADUAL PAULISTA
"JULIO DE MESQUITA FILHO" - CAMPUS DE ILHA SOLTEIRA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

"Algoritmo Genético Especializado na Resolução de Problemas com Variáveis Contínuas e Altamente Restritos"

DISSERTAÇÃO SUBMETIDA À UNIVERSIDADE ESTADUAL PAULISTA – UNESP PARA A OBTENÇÃO DO TÍTULO DE MESTRE EM ENGENHARIA ELÉTRICA

ÉRICO DE OLIVEIRA COSTA ZINI

ILHA SOLTEIRA - SP
Fevereiro de 2009

ÉRICO DE OLIVEIRA COSTA ZINI

Algoritmo Genético Especializado na Resolução de Problemas com Variáveis Contínuas e Altamente Restritos

Dissertação submetida ao Programa de Pós-graduação em Engenharia Elétrica da Faculdade de Engenharia de Ilha Solteira – UNESP, para obtenção do título de Mestre em Engenharia Elétrica.

Área de Conhecimento: Automação

Orientador: Prof. Dr. Rubén Augusto Romero Lázaro

Co-orientador: Prof. Dr. José Roberto Sanches Mantovani

ILHA SOLTEIRA – SP

Fevereiro de 2009

FICHA CATALOGRÁFICA

Elaborada pela Seção Técnica de Aquisição e Tratamento da Informação
Serviço Técnico de Biblioteca e Documentação da UNESP - Ilha Solteira.

Zini, Érico de Oliveira Costa.
Z77a Algoritmo genético especializado na resolução de problemas com variáveis contínuas e altamente restritos / Érico de Oliveira Costa Zini. -- Ilha Solteira : [s.n.], 2009.
149 f. : il.

Dissertação (mestrado) - Universidade Estadual Paulista. Faculdade de Engenharia de Ilha Solteira. Área de conhecimento: Automação, 2009

Orientador: Rubén Augusto Romero Lázaro
Co-orientador: José Roberto Sanches Mantovani
Bibliografia: p. 144-149

1. Algoritmo evolucionário. 2. Algoritmo multiobjetivo. 3. Manipulação de restrições. 4. Otimização com restrições.

unesp

UNIVERSIDADE ESTADUAL PAULISTA

CAMPUS DE ILHA SOLTEIRA

FACULDADE DE ENGENHARIA DE ILHA SOLTEIRA

CERTIFICADO DE APROVAÇÃO

TÍTULO: Algoritmo Genético Especializado na Resolução de Problemas com Variáveis Contínuas e Altamente Restritos

AUTOR: ÉRICO DE OLIVEIRA COSTA ZINI
ORIENTADOR: Prof. Dr. RUBEN AUGUSTO ROMERO LAZARO

Aprovado como parte das exigências para obtenção do Título de MESTRE em ENGENHARIA ELÉTRICA , Área: AUTOMAÇÃO, pela Comissão Examinadora:

Prof. Dr. RUBEN AUGUSTO ROMERO LAZARO
Departamento de Engenharia Elétrica / Faculdade de Engenharia de Ilha Solteira

Prof. Dr. ANTONIO PADILHA FELTRIN
Departamento de Engenharia Elétrica / Faculdade de Engenharia de Ilha Solteira

Prof. Dr. MARCOS JULIO RIDER FLORES
Departamento de Sistemas de Energia Elétrica / Universidade Estadual de Campinas

Data da realização: 20 de fevereiro de 2009.

É com amor e carinho que dedico este trabalho à minha esposa, Eliana; aos meus pais, Celso e Lucinda; às minhas irmãs, Ediléia e Elis; em gratidão ao imenso apoio e torcida pela minha vitória e, também, por me incentivarem a conquistar novos horizontes...

AGRADECIMENTOS

A minha sincera e profunda gratidão vão a todos que contribuíram de forma direta e indireta para que eu completasse com êxito mais esta importante etapa da minha vida. Primeiramente, a DEUS por me dar saúde, sabedoria e forças para lutar; à minha esposa, Eliana, pelo carinho e compreensão; aos meus pais, Celso e Lucinda, e às minhas irmãs, Ediléia e Elis, por terem me incentivado e me acompanhado em todas as etapas da minha vida pessoal e profissional.

Ao meu orientador, professor Dr. Rubén Romero pela sua extrema competência, compreensão, paciência em ensinar e corrigir alguns erros primários que às vezes eu cometia.

Ao grande companheiro Maurício Granada Echeverri, doutorando na UNESP de Ilha Solteira/SP e professor da Universidade Tecnológica de Pereira na Colômbia, por suas valiosas contribuições e importantes dicas, que foram cruciais no desenvolvimento deste trabalho.

Aos professores do Departamento de Engenharia Elétrica (DEE), em especial ao meu co-orientador professor Dr. José Roberto S. Mantovani, pela dedicação e os importantes conhecimentos que me proporcionou. Em geral, agradeço a todos os funcionários e colegas do DEE pela prontidão e apoio nestes anos de convivência.

Aos diversos amigos, Marcelo Oliveira Dias, Marcelo Fuly Batista, entre outros, pela grande ajuda e amizade. Enfim, agradeço a todas as pessoas que contribuíram para a conclusão deste trabalho, pois sei que sozinho não seria possível.

“Tantas vezes pensamos ter chegado,

Tantas vezes é preciso ir além...”

Fernando Pessoa.

RESUMO

Este trabalho apresenta uma metodologia composta de duas fases para resolver problemas de otimização com restrições usando uma estratégia multiobjetivo. Na primeira fase, o esforço concentra-se em encontrar, pelo menos, uma solução factível, descartando completamente a função objetivo. Na segunda fase, aborda-se o problema como biobjetivo, onde se busca a otimização da função objetivo original e maximizar o cumprimento das restrições. Na fase um propõe-se uma estratégia baseada na diminuição progressiva da tolerância de aceitação das restrições complexas para encontrar soluções factíveis. O desempenho do algoritmo é validado através de 11 casos testes bastantes conhecidos na literatura especializada.

Palavras-chave ¾ Algoritmo Evolucionário (AE), Algoritmo Multiobjetivo, Manipulação de Restrições, Otimização com Restrições.

ABSTRACT

This work presents a two-phase framework for solving constrained optimization problems using a multi-objective strategy. In the first phase, the objective function is completely disregarded and entire search effort is directed toward finding a single feasible solution. In the second phase, the problem is treated as a bi-objective optimization problem, where the technique converts constrained optimization to a two-objective optimization: one is the original objective function; the other is the degree function violating the constraints. In the first phase a methodology based on progressive decrease of the tolerance of acceptance of complex constrains is proposed in order to find feasible solutions. The approach is tested on 11 well-know benchmark functions.

Keywords: Evolutionary algorithm (EA), Multi-Objective Algorithm, Constraint Handling, Constrained Optimization.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

CAPÍTULO 1

INTRODUÇÃO

Neste capítulo apresentam-se as motivações para a elaboração do algoritmo que será proposto, uma introdução geral relatando os objetivos deste trabalho e, além disso, apresenta-se a organização deste texto. As idéias básicas para a elaboração deste trabalho foram, principalmente, baseadas no artigo de Venkatraman e Yen (VENKATRAMAN ; YEN, 2005).

1.1 Motivações

Nesta seção apresentam-se algumas motivações que influenciaram na elaboração de um Algoritmo Genético voltado, principalmente, à aplicação em problemas altamente restritos e com variáveis contínuas.

Os Algoritmos Genéticos são técnicas de buscas estocásticas que não oferecem garantia de produzir soluções factíveis. Pensando nesta característica, surge a motivação de implementar um Algoritmo Genético que traga confiança em produzir soluções que sejam úteis e obedeçam as restrições impostas ao problema. Assim, do ponto de vista prático, é essencial que um Algoritmo Genético utilizado em problemas de otimização restrita produza soluções factíveis para todas tentativas de busca pela solução ótima.

Uma motivação que também influenciou a elaboração do algoritmo que será proposto se refere aos diversos parâmetros dependentes do problema que são necessários para implementação de determinados Algoritmos Genéticos aplicados a problemas restritos. Sabe-se que quando se trata de otimização restrita, surgem vários tipos de problemas com suas respectivas peculiaridades. No entanto, seria impraticável se um Algoritmo Genético elaborado e ajustado para um problema específico não pudesse ser utilizado para a resolução de outros problemas. Assim, buscou-se elaborar um algoritmo que possa ser aplicado a uma grande variedade de problemas. Enfim, formulou-se uma estrutura bastante genérica, ou seja, não exige nenhuma informação subjetiva do problema e possui parâmetros de controle não variáveis.

1.2 Introdução Geral

A maioria dos problemas do mundo real apresenta, em sua formulação, restrições de igualdade e desigualdade. Na resolução destes problemas os Algoritmos Evolucionários, em especial os Algoritmos Genéticos, têm sido amplamente utilizados. Estas técnicas, ao serem comparadas com os métodos tradicionais de programação, possuem a vantagem de trabalhar com uma menor quantidade de informações não precisando, por exemplo, calcular gradientes, derivadas, hessianas entre outras. Além disso, estas técnicas são de fácil implementação e são ferramentas de busca global.

Durante os últimos anos, vários algoritmos evolucionários foram propostos para resolver problemas de otimização restrita. Considerando a otimização restrita, pode-se dizer que obter uma solução factível, ou seja, uma solução que seja útil na formulação do problema e satisfaça todas as restrições, prevalece, em determinado estágio do algoritmo, sobre otimizar a função objetivo. Existem problemas bastante complexos que são altamente restritos e, neste caso, encontrar uma única solução factível pode ser considerado uma tarefa extremamente difícil. Muitas vezes, impor que a obtenção de soluções factíveis seja sempre satisfeita, pode ser uma exigência um tanto exagerada. No entanto, deve-se pelo menos esperar que um critério de factibilidade seja satisfeito para todas as tentativas de buscas pela solução ótima e que uma otimização adequada seja alcançada (VENKATRAMAN; YEN, 2005). Sendo assim, objetivou-se elaborar um algoritmo que, em seu primeiro estágio, visa exclusivamente obter,

pelo menos, uma única solução factível. Após alcançar com sucesso o primeiro estágio, o algoritmo transfere-se para um segundo estágio que visa realizar a otimização simultânea da função objetivo e da violação das restrições. Neste estágio um problema mono-objetivo de otimização restrita é tratado como um problema biobjetivo e, desta forma, além de buscar a solução ótima global factível do problema, procura-se também obter soluções que violam de forma aceitável as restrições impostas na tentativa de obter um ganho importante nos valores da função objetivo.

Enfim, o objetivo principal deste trabalho foi elaborar um Algoritmo Genético que visa atender a factibilidade e, posteriormente, a factibilidade e otimalidade simultaneamente. Estas características do algoritmo proposto fazem com que sua estrutura seja eficiente e especializada na resolução de problemas que possuem variáveis contínuas e que são, preferencialmente, altamente restritos, visto que, obter soluções factíveis para estes tipos de problemas é bastante difícil e, além disso, estes problemas muitas vezes permitem uma pequena margem de violação das restrições impostas para obter uma melhoria importante no valor da função objetivo.

Para verificar o desempenho do algoritmo proposto, utilizam-se 11 problemas testes que são bastante utilizados como referência na literatura especializada. Desta forma, obtêm-se os resultados provenientes da aplicação do algoritmo proposto a estes problemas e, posteriormente, realiza-se uma comparação dos resultados obtidos aqui com as soluções já reportadas na literatura utilizando outros algoritmos.

1.3 Organização do Texto

Nesta seção apresenta-se como este trabalho foi organizado. Este trabalho está dividido em dez capítulos e a seguir é citado e comentado cada um destes capítulos:

Neste primeiro capítulo apresentaram-se as motivações e uma introdução geral com os objetivos da realização deste trabalho.

No capítulo 2 é introduzido o assunto otimização, apresentando a formulação geral e os conceitos básicos do tema, além disso, apresentam-se também as alternativas disponíveis para o tratamento desse assunto.

No capítulo 3 são apresentadas as diferenças conceituais e de tratamento entre a otimização mono-objetivo e multiobjetivo. São apresentados os conceitos, definições, formas de abordagem e alguns métodos clássicos de otimização multiobjetivo.

No Capítulo 4 apresentam-se alguns algoritmos evolucionários, em especial, apresenta-se a teoria do Algoritmo Genético básico fornecendo uma estrutura geral desta técnica. São apresentadas também as vantagens e desvantagens da aplicação desta técnica e, além disso, mostram-se as implicações da utilização de um algoritmo genético com codificação real.

No capítulo 5 são introduzidos alguns algoritmos evolucionários multiobjetivos existentes na literatura, em especial, apresenta-se a teoria do Algoritmo Genético Elitista Baseado em um Ordenamento Não-Dominado (NSGA-II) que representa uma técnica importante para o desenvolvimento deste trabalho.

No Capítulo 6 apresentam-se algumas formas de trabalhar com algoritmos evolucionários visando a resolução de problemas de otimização restrita. Este capítulo tem a finalidade de introduzir alguns métodos de manipulação de restrições existentes na literatura e apresentar algumas vantagens e desvantagens provenientes de suas aplicações.

No Capítulo 7 apresenta-se o algoritmo proposto. Descrevem-se suas características, particularidades e estratégias acopladas para sua formação. Este algoritmo é especializado na resolução de problemas altamente restritos e com variáveis contínuas e tenta evitar determinados problemas provenientes da aplicação de outros métodos de otimização restrita.

No Capítulo 8 apresentam-se os 11 problemas testes utilizados para verificar o desempenho do algoritmo proposto. Apresentam-se os enunciados dos problemas testes e, também, mostram-se as melhores soluções ótimas já encontradas e as características de cada um destes problemas. Além disso, quando possível, são esboçados os gráficos da função objetivo e do conjunto de restrições para alguns destes problemas.

No Capítulo 9 apresentam-se os resultados obtidos da aplicação do algoritmo proposto aos 11 problemas testes vistos no capítulo anterior. Além de apresentar os resultados obtidos, apresentam-se detalhadamente os parâmetros de controle utilizados e as dificuldades encontradas na obtenção das soluções. Na apresentação dos resultados, é feita uma comparação dos resultados obtidos com as melhores soluções reportadas na literatura.

Finalmente, no capítulo 10, apresentam-se as conclusões gerais obtidas da realização deste trabalho. Ainda neste capítulo são feitas sugestões para a realização de trabalhos futuros e, consequentemente, são apresentadas idéias que podem ser aplicadas ao algoritmo proposto visando sua melhoria.

CAPÍTULO 2

OTIMIZAÇÃO

2.1 Introdução

Neste capítulo são apresentados, de forma sucinta e objetiva, os principais conceitos, fundamentos e divisões do campo denominado otimização.

A palavra otimizar tem o sentido de melhorar o que já existe, ou então, projetar o novo com mais eficiência e menor custo. A otimização visa determinar a melhor configuração do projeto sem ter que testar todas as possibilidades. Otimização pode também ser definida como a necessidade de eficiência, que pode ser dividida nas etapas de reconhecimento das alternativas e a decisão. Primeiramente, é reconhecido numa tarefa ou problema a possibilidade de optar entre várias hipóteses e, em seguida, é decidida por aquela que se considera a melhor opção (ZINI, 2005).

A decisão pode ser qualitativa quando a escolha é feita através do bom senso, ou quantitativa quando a decisão é baseada em métodos matemáticos. No entanto, no processo de tomada de decisões é de grande auxílio o uso de alguns métodos ou técnicas para realizar tarefas ou resolver problemas.

Um problema de otimização possui a seguinte formulação geral (KOZIEL; MICHALEWICZ, 1999):

$$\text{Otimizar } f(\vec{x}) \tag{2.1}$$

Sujeito a

$$g_j(\vec{x}) \le 0, \text{ para } j = 1, \mathbf{K}, q \tag{2.2}$$

$$h_j(\vec{x}) = 0, \text{ para } j = q+1, \mathbf{K}, m \tag{2.3}$$

$$l(i) \le x_i \le u(i), \quad 1 \le i \le n \tag{2.4}$$

sendo que:

$\vec{x} = (x_1, \mathbf{K}, x_n)$ é definido como sendo as variáveis de decisão ou de projeto. Estas variáveis podem ser contínuas (reais), inteiras ou discretas.

Na expressão (2.1) está definida a função $f(\vec{x})$ que se deseja otimizar. Esta função é chamada de função objetivo do problema e pode ser minimizada ou maximizada.

A expressão (2.4) representa as restrições laterais que definem o espaço de busca que será chamado de S. Este espaço representa o domínio das variáveis definidas pelos seus limitantes inferior e superior. A região S representa o conjunto de todos os pontos onde a função objetivo do problema está definida.

As expressões (2.2) e (2.3) representam um conjunto de *m* restrições que definem uma região que será denominada por F. Esta região está contida no espaço de busca S e será chamada de região factível. Uma solução que pertença à região F é bastante desejável, visto que, a referida região satisfaz, simultaneamente, as expressões de (2.2), (2.3) e (2.4).

O ponto $x^* = (x_1, x_2, \mathbf{K}, x_n)$ é denominado ponto ótimo. Este ponto é formado pelas variáveis de decisão que otimizam a função objetivo $f(\vec{x})$ em alguma região e satisfazem as restrições.

O valor $f(x^*)$ representa o valor da função objetivo no ponto ótimo. Este valor será chamado de valor ótimo do problema.

O par formado pelo ponto ótimo e valor ótimo, ou seja, $(x^*, f(x^*))$ representa a solução ótima do problema. As soluções ótimas podem ser classificadas como (CASTRO, 2001):

- local: quando o valor é ótimo considerando uma parte do espaço de busca;

- global: quando o valor é ótimo considerando todo o espaço de busca;

A seguir, mostra-se a divisão de algumas técnicas de otimização existentes.

2.2 Divisão das Técnicas de Otimização

É comum deparar-se com situações em que é necessário decidir determinadas características de um sistema de onde se deve extrair o maior número possível de benefícios. Pensando nisto, foram desenvolvidos métodos e técnicas que permitem descobrir quais os valores devem ser escolhidos para atingir os valores máximos ou mínimos referentes a determinadas situações (LUCAS, 2002). Portanto, para a resolução de diversos problemas, encontram-se disponível na literatura uma variedade de técnicas e métodos. Em (LINDEN, 2006), por exemplo, estas técnicas estão divididas como:

- Técnicas de busca baseadas em cálculo;
- Técnicas de busca aleatórias guiadas;
- Técnicas de busca enumerativas;

O diagrama abaixo, relaciona e subdivide as referidas técnicas de busca:

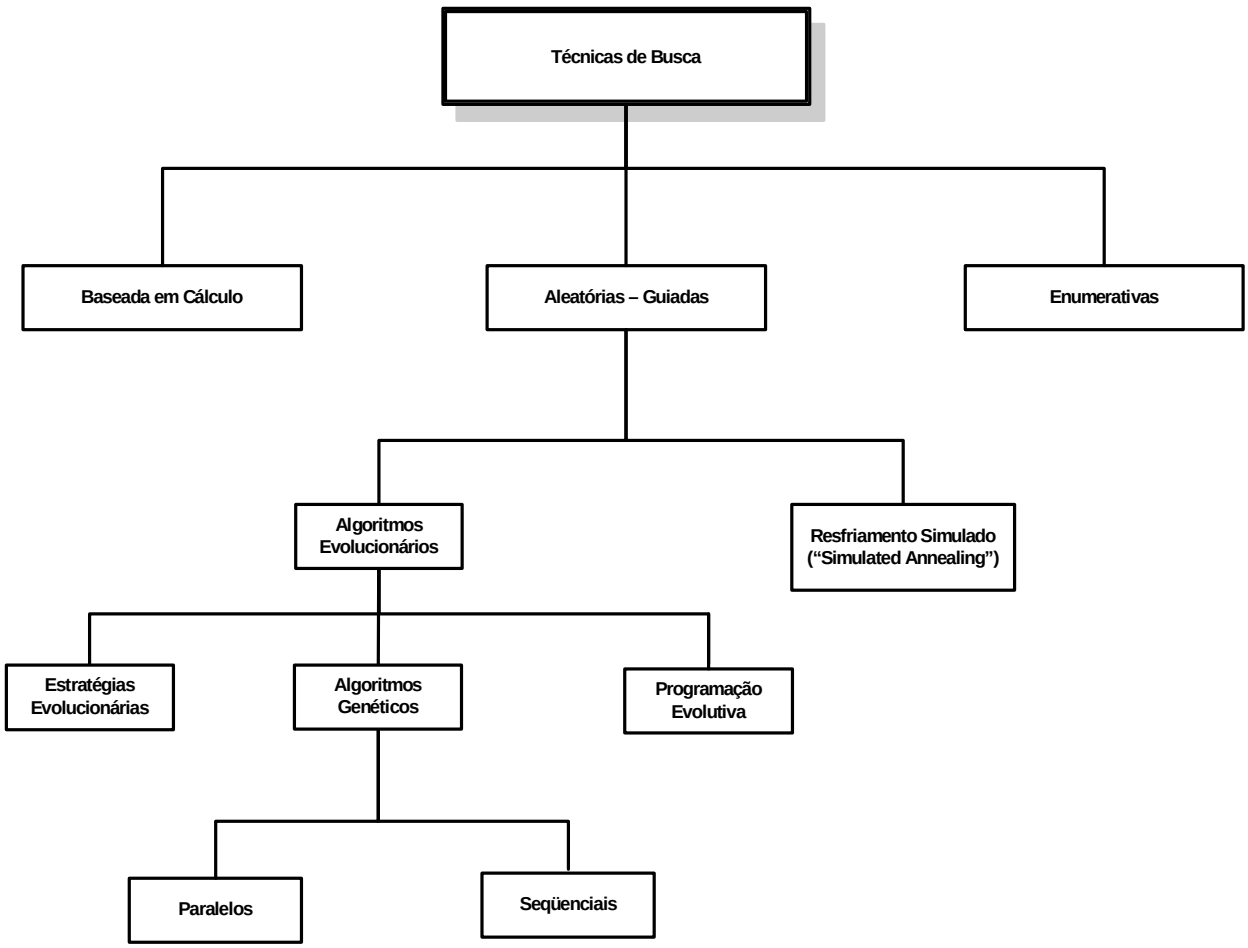

Figura 2.1: Subdivisão das técnicas de busca.

2.2.1 Técnicas Baseadas em Cálculo

Estes técnicas usam um conjunto de condições necessárias e suficientes que devem ser satisfeitas pelas soluções de um problema de otimização. Nesta classe, estão presentes os métodos de Programação Matemática (também conhecido como Métodos Clássicos) que foram uns dos primeiros métodos desenvolvidos para o tratamento dos problemas de otimização. Existe uma grande quantidade de textos voltados a este assunto, como por exemplo, (BAZARAA et al. 1990, 1993).

Na programação matemática, o problema é tratado de forma iterativa e determinística. Para isto, necessita-se trabalhar com uma variedade de informações tais como: gradientes, hessianas, derivadas entre outras e, por esta razão, exigem normalmente muitas informações e condições do problema a ser resolvido como, por exemplo, região factível bem definida, suavidade da função que se deseja otimizar e convexidade do problema.

Uma grande desvantagem da programação matemática é ainda a inexistência de um método que busque soluções ótimas globais, ou seja, as soluções produzidas pelos métodos de programação matemática são suscetíveis a ficarem presas em ótimos locais. Os métodos matemáticos geralmente apresentam teoremas de garantia de convergência para uma solução ótima. No entanto, a solução encontrada não será necessariamente a solução ótima global, o que pode, eventualmente até ocorrer. Isto dependerá basicamente da solução inicial adotada, já que, os métodos de programação matemática necessitam de uma solução inicial para o início do método iterativo (CASTRO, 2001).

Os métodos de Programação Matemática ainda podem ser classificados em diferentes áreas de acordo com o comportamento da função objetivo e das restrições. A seguir, são mostradas estas classificações:

- *Programação Linear:* a função objetivo e as restrições são lineares.

- *Programação Não-Linear:* a função objetivo ou pelo menos uma das restrições são não-lineares.

Em seguida, foram surgindo outras áreas com o intuito de aperfeiçoar a resolução dos problemas. Um exemplo é a Programação Quadrática, onde a função objetivo é quadrática e as restrições são lineares. Outros exemplos que se enquadram nas áreas de Programação Matemática, surgem do fato de os problemas de otimização apresentar ou não restrições. Com isso, surgem os Problemas de Otimização com restrições ou restritos e sem restrições ou irrestritos.

A seguir, tenta-se relacionar e classificar alguns dos inúmeros métodos de programação matemática existentes para o tratamento de problemas de otimização (CASTRO, 2001 apud NEVES, 1997):

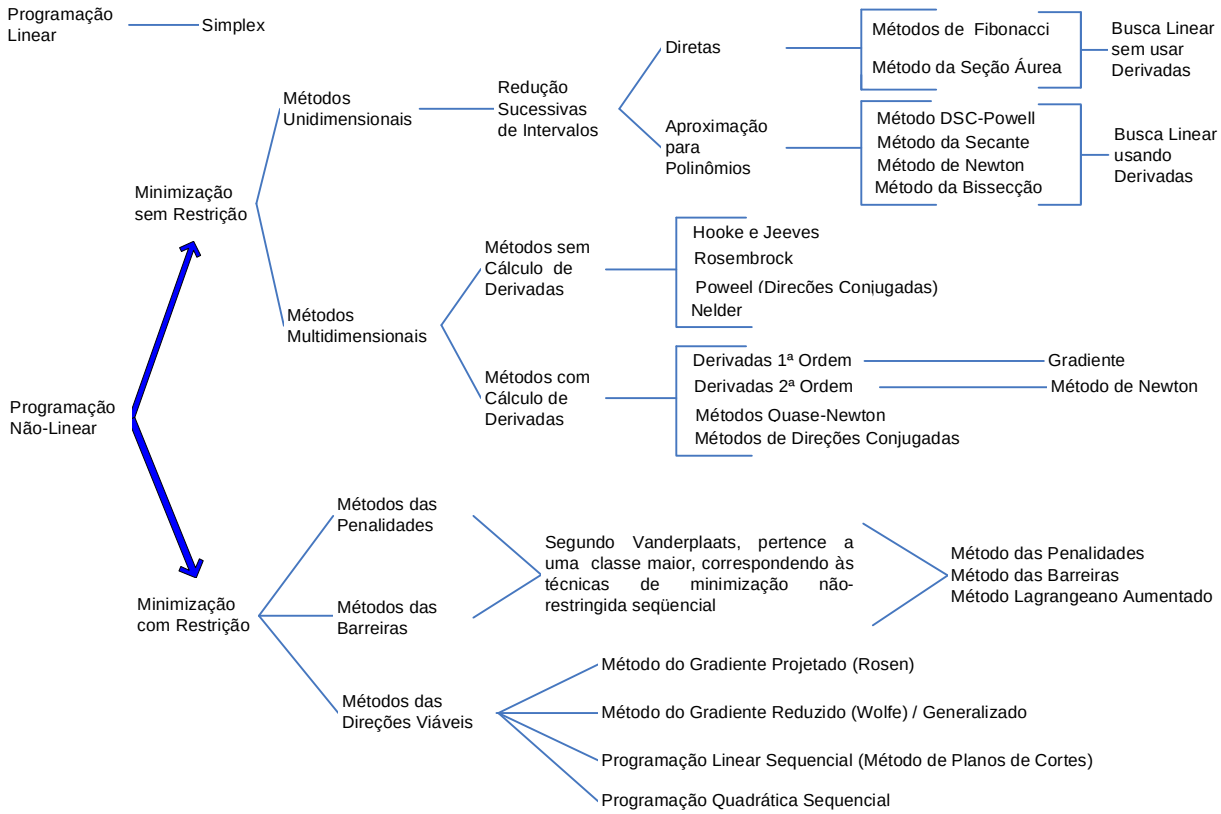

Figura 2.2: Classificação geral dos métodos matemáticos.

2.2.2 Técnicas Enumerativas

As técnicas enumerativas procuram a solução pesquisando, em seqüência, cada ponto do espaço de busca (finito e discreto). Uma destas técnicas é conhecida como Programação Dinâmica (BARCELLOS, 2000).

2.2.3 Técnicas Aleatórias Guiadas

Estas técnicas são baseadas nas técnicas enumerativas. No entanto, utilizam informações adicionais para dirigir a busca às soluções. Entre estas técnicas, estão o Resfriamento Simulado e os Algoritmos Evolucionários.

O Recozimento Simulado ou, talvez mais conhecido pelo termo Simulated Annealing, é uma técnica baseada em um processo que faz uma analogia com a termodinâmica. Na construção de cristais perfeitos, um material é aquecido até uma temperatura elevada e depois esfriado de forma lenta, mantendo durante o processo o chamado quase equilíbrio termodinâmico. Assim, o Recozimento Simulado tenta simular um processo equivalente para encontrar a solução ótima do problema (ROMERO; MANTOVANI, 2004).

Os Algoritmos Evolucionários são técnicas de busca estocásticas, poderosas e amplamente aplicáveis, que fazem uma analogia com os mecanismos de evolução natural e da genética. O princípio básico destas técnicas é a evolução de uma população de indivíduos (conjunto de soluções). Conforme a linha de aplicação e alguns detalhes dos Algoritmos Evolucionários podem-se dividir esta técnica em Estratégias Evolucionárias, Algoritmos Genéticos, Programação Evolutiva. Neste trabalho, será dada ênfase aos Algoritmos Evolucionários, em especial aos Algoritmos Genéticos, visto que, utiliza-se esta técnica para o desenvolvimento principal deste trabalho. Mais adiante, será dedicado um capítulo exclusivo para explicitar os conceitos destas técnicas.

CAPÍTULO 3

OTIMIZAÇÃO MULTIOBJETIVO

3.1 Introdução

Neste capítulo, são apresentados alguns conceitos e definições da otimização multiobjetivo. No capítulo anterior, apresentou-se a formulação geral de um problema de otimização representados pelas expressões de (2.1) a (2.4). Nesta formulação, foi apresentada uma função objetivo e um conjunto de restrições. Na presença de uma única função objetivo, o problema é chamado de otimização mono-objetivo. Na otimização multiobjetivo, em sua formulação, além de um conjunto de restrições, aparece também mais de uma função objetivo e, portanto, uma das diferenças entre as otimizações mono-objetivo e multiobjetivo refere-se à quantidade de funções objetivos associadas ao problema.

Uma grande maioria dos problemas do mundo real apresentam mais de um objetivo a serem otimizados que são, na maioria das vezes, conflitantes entre si, ou seja, a melhoria de um ou mais objetivos causam, consequentemente, a deterioração de outro(s).

Em Deb (2004) é apresentado um exemplo bastante básico que possui objetivos conflitantes. Este exemplo está associado à aquisição de um automóvel.

Na busca da compra de um automóvel, o mercado oferece vários preços e diferentes graus de conforto. A figura a seguir, ilustra uma dessas situações:

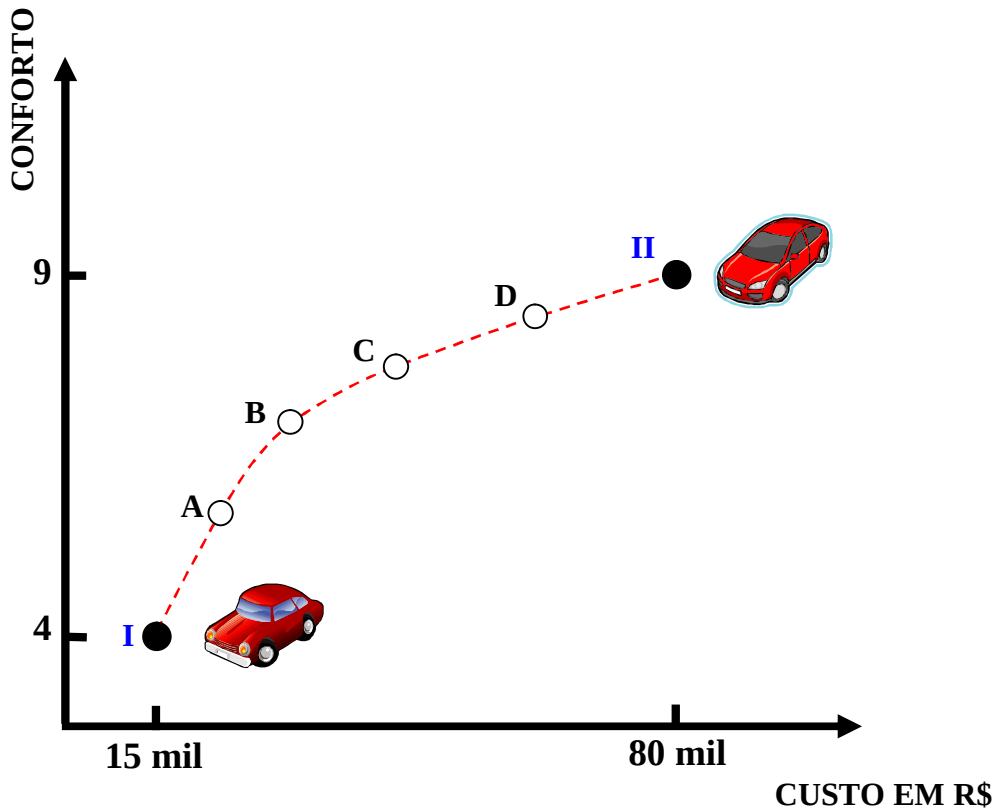

Figura 3.1: Exemplo de aquisição de um automóvel (soluções I, A, B, C, D e II).

Considere na ilustração, um automóvel (solução I) e outro (solução II) com os respectivos valores de mercado: R$ 15 mil e R$ 80 mil. Nesta situação, suponha que o único objetivo é minimizar o custo do automóvel. Desta forma, a solução ótima escolhida será a representada por I e, provavelmente, não haveria carros no mercado que produzam custos altos.

Na vida real, é evidente que esta situação não admite um único objetivo, ou seja, deve-se considerar também o grau de conforto do automóvel. É bastante provável que um automóvel com menor custo seja também menos confortável. A figura anterior indica que o carro mais barato possui um nível de conforto igual a 4. Uma pessoa que almeja o maior conforto escolheria a solução II que possui um grau de conforto igual a 9.

Pode-se verificar que entre as soluções extremas (I e II) existem, na linha tracejada, outras soluções que proporcionam uma variação de custo e conforto. Na figura, as soluções

A, B, C e D representam estas soluções com diferentes custos e confortos. Neste caso, pode-se notar que entre todas as soluções presentes, não é possível dizer que uma solução é melhor que outra, ou seja, nenhuma solução que tenha menor custo e conforto pode ser considerada superior à outra com maior custo e conforto. A escolha do melhor automóvel representado por estas soluções só pode ser feita levando em consideração as necessidades e as possibilidades financeiras da pessoa que deseja adquirir o automóvel. Sendo assim, nos problemas de otimização multiobjetivo existe também uma distinção com os problemas de otimização mono-objetivo quanto à maneira que as soluções são adquiridas. Portanto, isso faz com que esses problemas apresentem um conjunto de soluções ótimas.

Considerando agora a mesma situação do problema e, acrescentando às opções existentes mais 3 tipos de carros (soluções E, F e G), obtêm-se a seguinte situação ilustrada na figura a seguir:

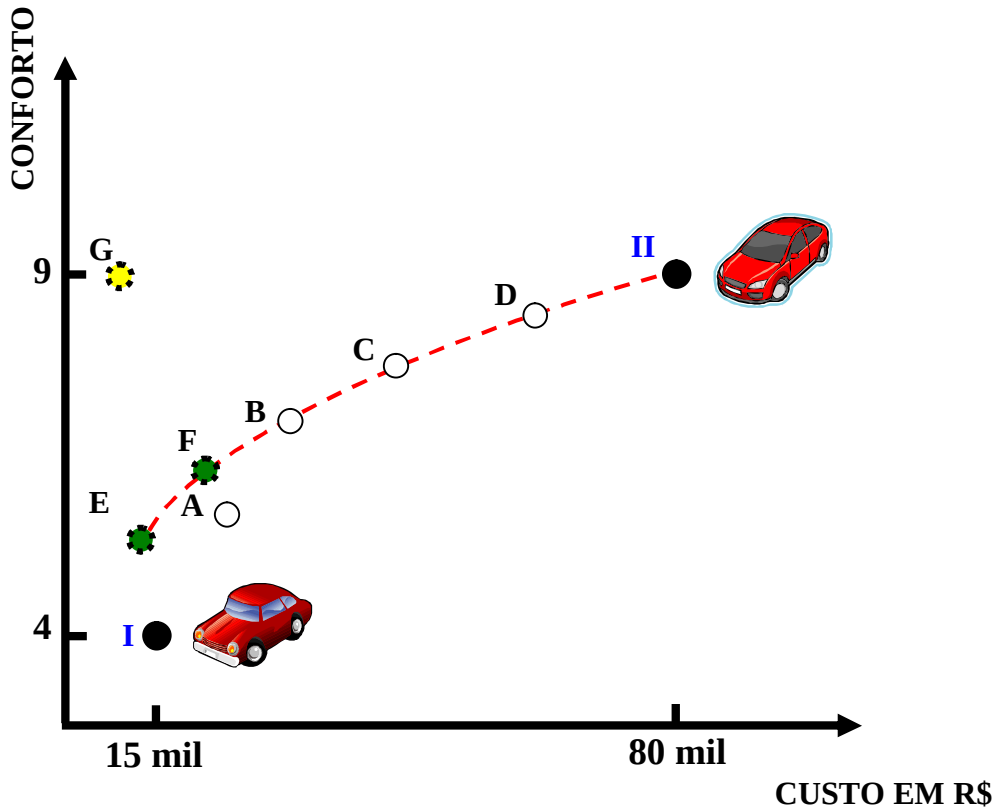

Figura 3.2: Exemplo de aquisição de um automóvel (soluções I, A, B, C, D, II, E, F e G).

Embora comprar um carro com baixo custo e alto conforto seja uma condição ideal, esta condição não é real para o mercado. Portanto, na figura a solução G com baixo custo e

alto conforto é uma opção dita absurda e não deve ser considerada. Porém, existem soluções que são superiores às outras, isto é, apresentam maior conforto e custo menor ou igual. Na figura, as soluções E e F superam, respectivamente, as soluções I e A. Estas soluções que superam as outras são chamadas de *não-dominadas*, enquanto que, as soluções que são superadas por, pelo menos, uma outra solução são denominadas *dominadas*. Assim, quem deseja comprar um automóvel, diante de todas as opções apresentadas na figura anterior (obviamente excluindo a solução G) a melhor opção, desta vez, é escolher entre as soluções (E, F, B, C, D e II).

Deste modo, é muito interessante uma técnica ou método que encontre o conjunto das soluções não-dominadas para que, entre essas, possa ser escolhida aquela que melhor atenda as necessidades de quem busca por estas soluções. Essa é a tarefa da otimização multiobjetivo. Assim, como na otimização mono-objetivo, na otimização multiobjetivo também existem diversos métodos e técnicas para resolver problemas que possuem mais de duas funções objetivos. Mais adiante, serão vistos algumas dessas técnicas.

3.2 Métodos de Abordagem da Otimização Multiobjetivo

Foi visto que uma diferença fundamental entre a otimização mono-objetivo e multiobjetivo está na quantidade de soluções ótimas obtidas. No entanto, do ponto de vista prático, ao resolver um problema de otimização necessita-se, na maioria das vezes, de somente uma solução como resposta independentemente do tipo de otimização adotado. Assim, dentro da otimização multiobjetivo existem duas abordagens no tratamento do problema:

1) Definidas as prioridades e pesos entre os vários objetivos de interesse, encontra-se a solução ótima baseadas nas informações fornecidas anteriormente;

2) Não contendo nenhuma informação adicional, encontra-se o conjunto de soluções não-dominadas para posteriormente escolher uma solução dentre este conjunto.

Em Deb (2004), são esquematizados estas duas abordagens. Nas figuras a seguir mostram-se estas duas abordagens considerando que o problema possui duas funções objetivos.

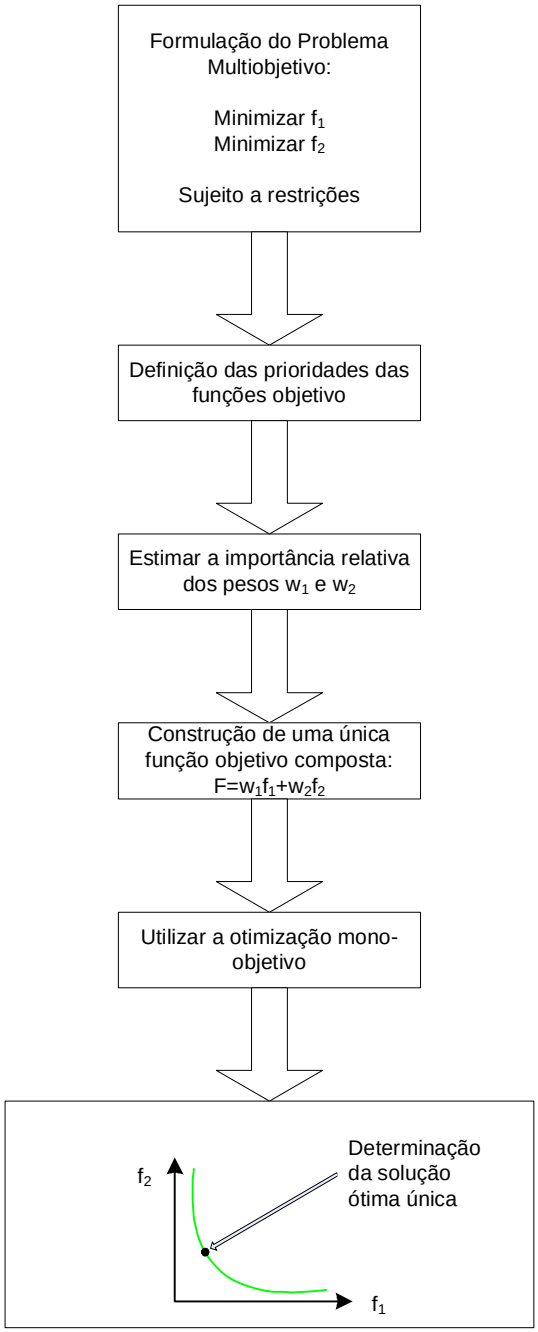

Figura 3.3: Esquematização da abordagem 1.

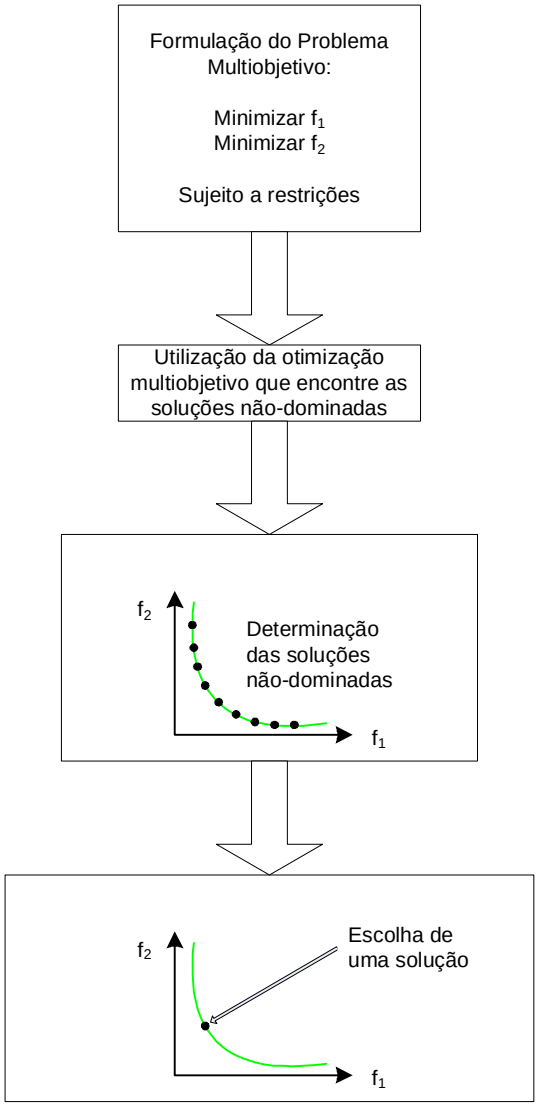

Figura 3.4: Esquematização da abordagem 2.

Na abordagem 1, representada pela figura 3.3, é importante ressaltar que é geralmente bastante difícil estabelecer o nível relativo da importância dos objetivos. Assim, a difícil questão da primeira abordagem é como definir todas as prioridades e pesos dos problemas que, na maioria das vezes, se conhece tão pouco. Nesta abordagem, as diversas funções objetivos do problema são transformadas em uma única e, portanto, basta utilizar uma técnica de otimização mono-objetivo para resolver o problema resultante. Neste caso, encontram-se disponíveis vários algoritmos que podem ser baseados em métodos clássicos, algoritmos de busca aleatória guiada como, por exemplo, o Recozimento Simulado ou até mesmo nos algoritmos evolucionários.

Por outro lado, existem algoritmos evolucionários que trabalham com os problemas baseando-se na abordagem 2. Assim esses algoritmos possibilitam a obtenção de um grande número de soluções não-dominadas para que, posteriormente, seja escolhida entre este conjunto a melhor opção.

Realmente, em problemas multiobjetivos do mundo real existe a dificuldade em encontrar informações qualitativas e quantitativas inerentes ao problema. Deste modo, diante dos fatos apresentados, pode-se concluir que a abordagem 2 é bem mais interessante. Portanto, esta conclusão faz com que os Algoritmos Evolucionários sejam preferidos em relação aos métodos clássicos ou mesmo ao Recozimento Simulado, já que estas duas categorias não foram projetadas para trabalharem com múltiplas soluções como acontece naturalmente com os Algoritmos Genéticos (CASTRO, 2001). Para o desenvolvimento principal deste trabalho, utiliza-se como base a abordagem 2 através da utilização de Algoritmos Genéticos. No entanto, serão apresentados também alguns conceitos de métodos que devem ser aplicados segundo as idéias da abordagem 1.

3.3 Definição do Problema de Otimização Multiobjetivo

A formulação geral de um problema de otimização multiobjetivo, envolve a minimização ou maximização de algumas funções objetivos que estão sujeitas a uma determinada quantidade de restrições. Assim, a formulação geral do problema de otimização multiobjetivo pode ser expressa por (DEB, 2004):

$$\text{Otimizar } Z = f_k(\vec{x}), \text{ para } k = 1,2,\mathbf{K},l \text{ com } l \geq 2 \tag{3.1}$$

Sujeito a

$$g_j(\vec{x}) \leq 0, \text{ para } j = 1,\mathbf{K},q \tag{3.2}$$

$$h_j(\vec{x}) = 0, \text{ para } j = q+1,\mathbf{K},m \tag{3.3}$$

$$l(i) \leq x_i \leq u(i), \quad 1 \leq i \leq n \tag{3.4}$$

A formulação dada pelas expressões (3.1) a (3.4) são semelhantes às dadas no capítulo anterior para um problema de otimização mono-objetivo. Portanto, as definições dadas para as expressões do capítulo anterior, valem também para estas expressões. A diferença, como já foi dito, refere-se à quantidade de funções objetivos associadas ao problema que devem ser otimizadas simultaneamente. Desta forma, o que se busca é encontrar um ponto $\vec{x} = (x_1, \mathbf{K}, x_n)$ que satisfaça o conjunto de restrições e otimize as funções objetivos envolvidas no problema. Assim, com a presença de mais de uma função objetivo, surge um novo espaço importante denominado *espaço objetivo*. Portanto, pode ser feito o mapeamento do espaço das variáveis de decisão com o seu correspondente espaço objetivo. A figura a seguir, representa o mapeamento entre o espaço das variáveis e o objetivo, ambos bidimensionais, ou seja, foram consideradas duas variáveis de decisão e duas funções objetivos.

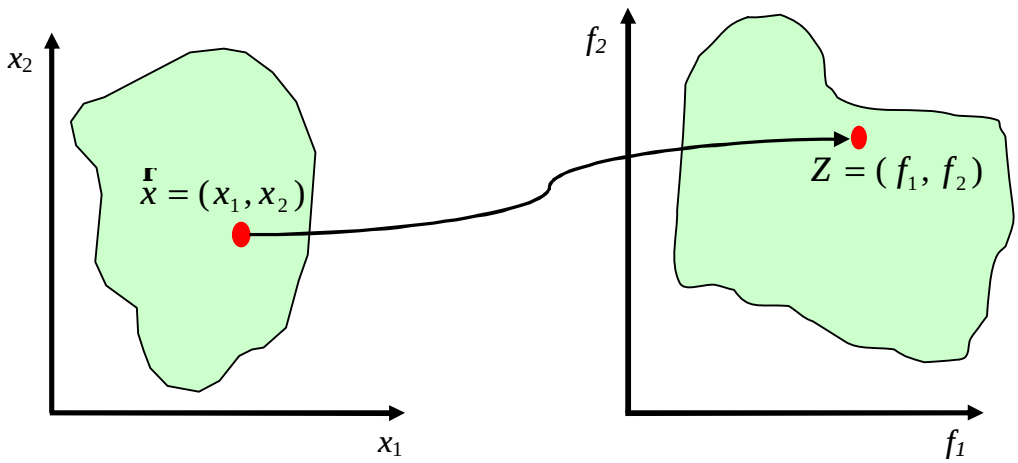

Figura 3.5: Representação do espaço das variáveis de decisão e o correspondente espaço objetivo.

Como já foi comentada, a maneira mais interessante para obter as soluções ótimas de um problema multiobjetivo é encontrar um conjunto de soluções que sejam não-dominadas. Quando se refere às soluções não-dominadas, foi visto que essas superam outras soluções levando em conta os valores das funções objetivos. Sendo assim, o comportamento de uma solução no espaço objetivo é que determina se uma solução é não-dominada ou não. Portanto, a relação entre as soluções mapeadas no espaço objetivo irá guiar a escolha de pontos desejáveis no espaço das variáveis. Essa relação é traduzida pela comparação entre cada duas

soluções mapeadas levando-se em consideração cada função objetivo do problema. Baseando-se nesta idéia, surge o conceito de dominância que será visto a seguir.

3.4 Dominância e Soluções Pareto Ótimas

A maioria dos algoritmos de otimização multiobjetivos usam os conceitos de dominância. Nestes algoritmos, duas soluções são comparadas para verificar se uma domina ou não a outra. Segundo Deb (2004), o conceito de dominância pode ser definido como:

Definição 1: Uma solução $\vec{x}(1)$ domina outra solução $\vec{x}(2)$, se ambas as condições 1 e 2 são verdadeiras:

1- A solução $\vec{x}(1)$ não é pior que $\vec{x}(2)$ em todos os objetivos;
2- A solução $\vec{x}(1)$ é estritamente melhor que $\vec{x}(2)$ em pelo menos um objetivo.

Caso qualquer uma das condições acima seja violada, a solução $\vec{x}(1)$ não domina a solução $\vec{x}(2)$. Se $\vec{x}(1)$ domina a solução $\vec{x}(2)$ (ou matematicamente $\vec{x}(1)$ **p** $\vec{x}(2)$), este fato também pode ser escrito como:

- $\vec{x}(2)$ é dominado por $\vec{x}(1)$;

- $\vec{x}(1)$ é não-dominado por $\vec{x}(2)$;

- $\vec{x}(1)$ é não-inferior a $\vec{x}(2)$.

Considerando um problema de otimização de minimização com *k* funções objetivos, ou seja, todas as funções objetivos deste problema devem ser de minimização. Pode-se descrever o conceito de dominância para este tipo de situação como:

Definição 2: Uma solução $\vec{x}(1)$ domina outra solução $\vec{x}(2)$ se:

- qualquer que seja $k \in \{1,2,\mathbf{K},l\}$, tal que $f_k(\vec{x}(1)) \leq f_k(\vec{x}(2))$, existe $j \in \{1,2,\mathbf{K},l\}$, tal que $f_j(\vec{x}(1)) < f_j(\vec{x}(2))$.

Para facilitar o entendimento das definições anteriores, considere a figura a seguir supondo que se trata de um problema de minimização (BARDANACHVILI, 2006).

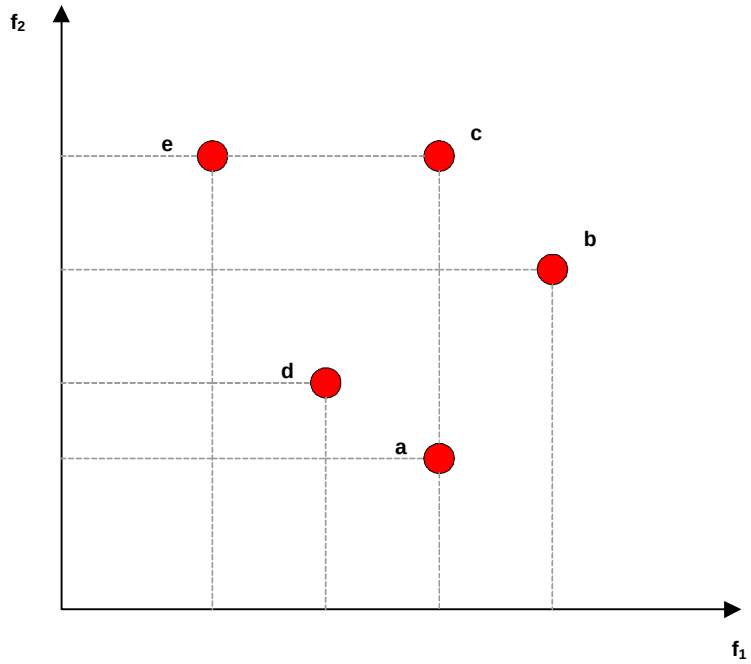

Figura 3.6: Relação de dominância entre as soluções.

Pelas definições anteriores, a solução ***a*** domina as soluções ***b*** e ***c***, ou seja, ***a*** $\underline{\mathbf{p}}$ ***b*** e ***a*** $\underline{\mathbf{p}}$ ***c***. As soluções ***a*** e ***d*** são não-dominadas entre si. Pode-se afirmar que as soluções ***a***, ***d*** e ***e*** formam um conjunto de soluções não-dominadas que domina o conjunto formado pelas soluções ***b*** e ***c***. Logo, isto significa que um elemento do conjunto {***b***, ***c***} encontrará pelo menos um elemento de {***a***, ***d***, ***e***} que o domine.

De modo geral, dado um conjunto de soluções P, o conjunto de soluções não-dominadas P' é aquele contendo elementos não-dominados por qualquer elemento do conjunto P. Então, quaisquer 2 soluções de P' são não-dominadas entre si e qualquer solução das demais do conjunto P são dominadas por pelo menos um elemento de P'. Caso o conjunto P seja o próprio espaço de busca S, então o conjunto P' é chamado de conjunto *Pareto ótimo*.

Sendo assim, existe uma sutil diferença entre um *conjunto de soluções não-dominadas* e um *conjunto Pareto ótimo*. Um conjunto de soluções não-dominadas é definido no contexto de uma amostra do espaço de busca S, enquanto o conjunto Pareto ótimo é definido em

relação a todo espaço de busca. Ao considerar somente funções a serem minimizadas, as soluções Pareto ótimas, assumem o aspecto da figura a seguir:

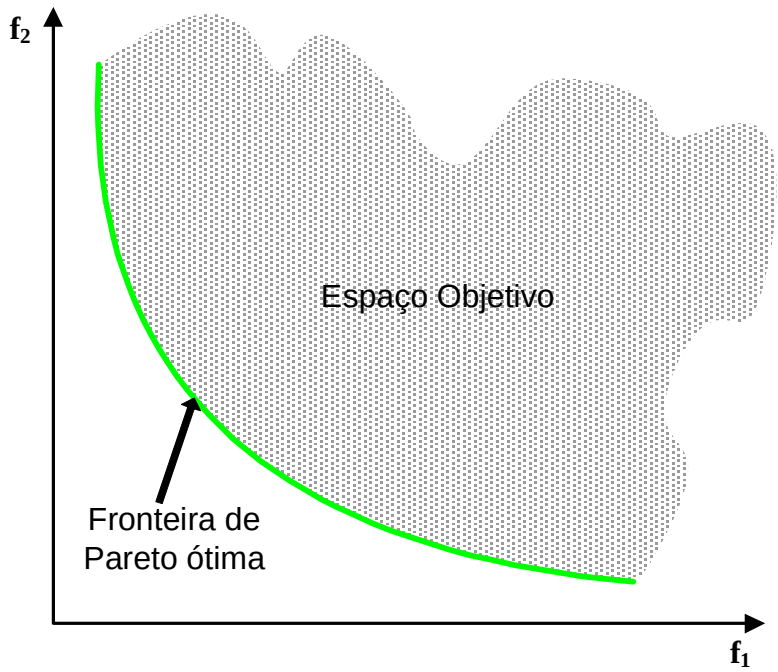

Figura 3.7: Fronteira de Pareto ótima.

Um problema simples de otimização multiobjetivo foi proposto e testado por Schaffer (SCHAFFER, 1984) e tem como objetivo minimizar, simultaneamente, duas funções objetivos ***g*** e ***h*** definidas por:

$$g(x) = x^2 \qquad \text{e} \qquad h(x) = (x-2)^2 \tag{3.5}$$

A representação gráfica de ambas as funções ***g*** e ***h*** são mostradas na figura seguinte. Analisando esta representação, é possível verificar que as soluções ótimas de Pareto devem estar compreendidas no intervalo [0, 2], visto que, fora deste intervalo ambas as funções crescem simultaneamente.

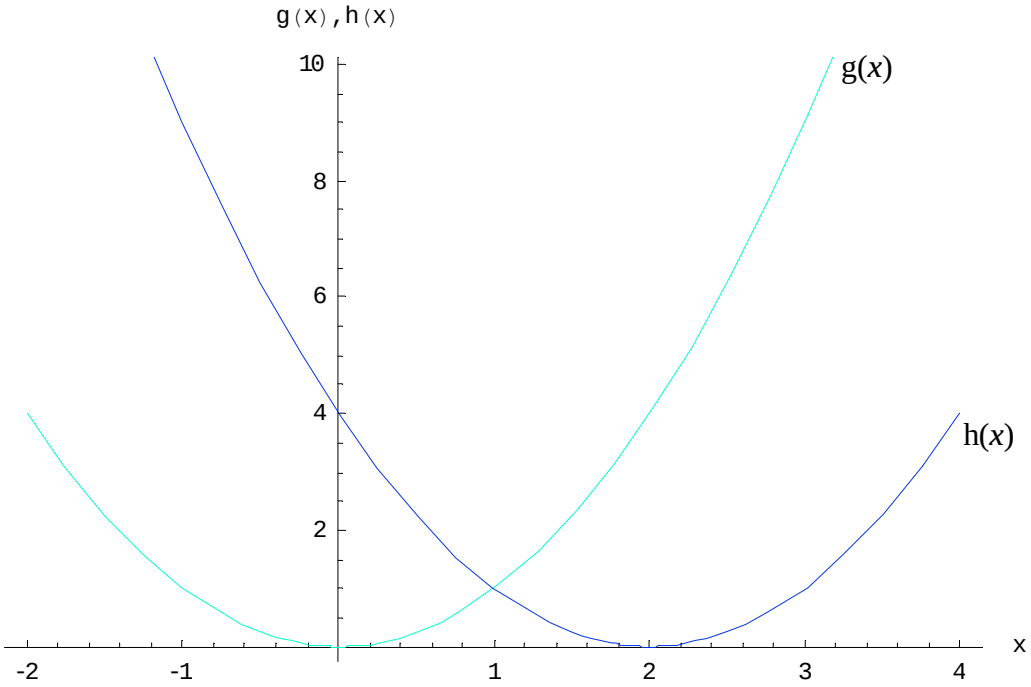

Figura 3.8: Funções g e h de um problema multiobjetivo.

No intervalo [0, 2], enquanto uma função cresce a outra diminui de valor e, desta forma, não é tão fácil identificar a fronteira de Pareto. No entanto, esboçando o gráfico g x h tem-se o espaço objetivo e, portanto, pode-se identificar a fronteira de Pareto. O gráfico do espaço objetivo é mostrado a seguir.

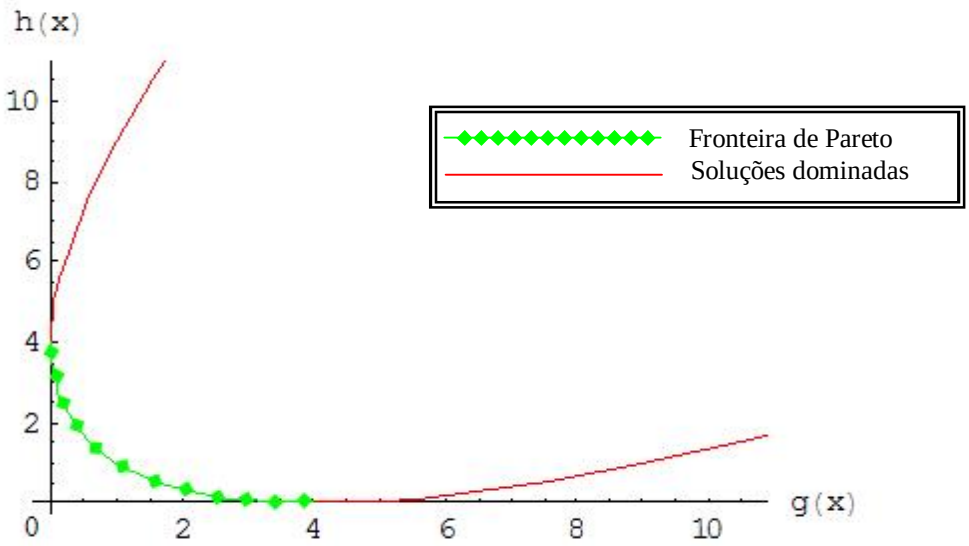

Figura 3.9: Identificação da fronteira de Pareto

A Fronteira de Pareto, conhecida também como Frente de Pareto é uma curva composta de soluções não-dominadas, como já foi dito, considerando todo o espaço de busca. Desta forma, deduz-se que o objetivo do processo de otimização multiobjetivo será obter os elementos da fronteira de Pareto, já que, qualquer solução fora dela encontraria solução melhor sobre ela. Assim, o espaço de soluções se divide em soluções ótimas e não-ótimas. O exemplo anterior apresenta apenas dois objetivos e, portanto, foi fácil identificar a fronteira de Pareto. No entanto, para problemas mais complexos não é possível fazer uma análise gráfica e, assim, é necessário o uso de técnicas ou métodos computacionais para a resolução destes problemas.

Na fronteira de Pareto, em princípio, não há preferência por nenhuma das soluções, sendo necessário acrescentarem relações entre as funções objetivos para escolher uma determinada solução da fronteira de Pareto.

Os pontos no espaço objetivo são obtidos geralmente por métodos ou técnicas computacionais e, desta forma, deve-se procurar encontrar um número adequado de pontos o mais próximo possível da fronteira de Pareto. Além de encontrar os referidos pontos, é desejável que estes estejam relativamente bem espaçados. Portanto, na otimização multiobjetivo, busca-se alcançar duas importantes metas na busca de soluções (DEB, 2004):

1- Encontrar um conjunto de soluções o mais próximo possível da fronteira de Pareto;

2- Encontrar um conjunto de soluções que estejam relativamente bem espaçados, ou seja, com a maior diversidade possível.

A meta 1 é comum a qualquer processo de otimização, já a meta 2 é específica para problemas de otimização multiobjetivo. A figura a seguir, representa a solução de um problema que é ideal para a otimização multiobjetivo, ou seja, as duas metas impostas foram atendidas.

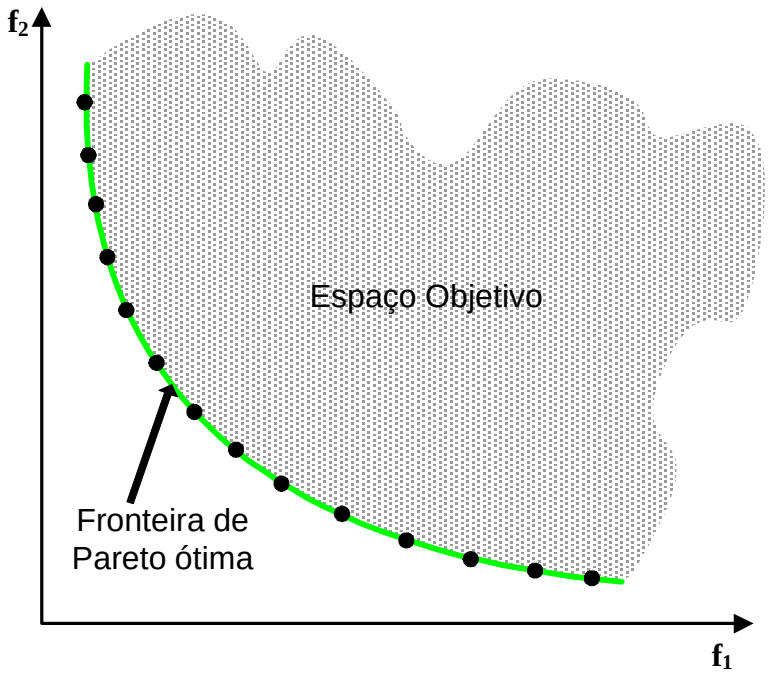

Figura 3.10: Soluções bem espaçadas na fronteira de Pareto.

No entanto, a figura a seguir atende somente a meta 1, ou seja, encontram-se soluções ótimas, mas com baixa diversidade.

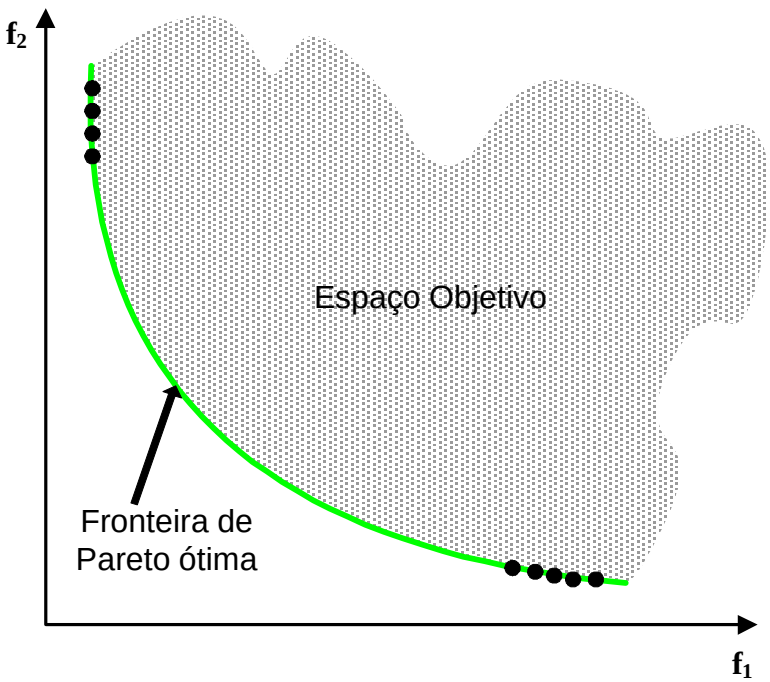

Figura 3.11: Soluções com baixa diversidade na fronteira de Pareto.

3.5 Procedimentos para Encontrar um Conjunto Não-Dominado

Encontrar as soluções não-dominadas de um dado conjunto é similar ao princípio de encontrar o valor mínimo de um conjunto de número reais. No caso de encontrar o valor mínimo em um conjunto dos números reais, dois números são comparados para verificar qual é o menor e, desta forma, o operador '<' (menor) é usado para fazer esta verificação. No entanto, para encontrar um conjunto não-dominado a relação de dominância '**p**', vista anteriormente, pode ser usada para identificar a melhor de duas soluções. Assim, como existem diferentes procedimentos para encontrar o valor mínimo de um conjunto finito de números reais, para encontrar um conjunto de soluções não-dominadas de um dado conjunto não é diferente (DEB, 2004). Os procedimentos para determinar um conjunto não-dominado geralmente possuem diferentes complexidades computacionais e, desta forma, será apresentado apenas o método considerado pela literatura especializada como o mais eficiente. Este procedimento é chamado de Método de Kung e será visto em seguida. Em Deb (2004), além do Método de Kung apresentam-se mais dois métodos para a obtenção de um conjunto de solução não-dominadas.

3.5.1 Método de Kung para Obtenção do Conjunto Não-Dominado

Entre os métodos de abordagem dos problemas de otimização multiobjetivo visto neste capítulo, os métodos que obtêm um conjunto não-dominado como solução é o mais interessante. E não é para menos, pois a maioria dos algoritmos de otimização multiobjetivo usa o conceito de dominância em seus processos de busca pelas soluções ótimas. Nestes algoritmos, duas soluções são comparadas a fim de estabelecer qual solução domina a outra.

O método de Kung, apresentado em Deb (2004), Peñuela e Granada (2007), propõe uma divisão recursiva do conjunto de soluções. O primeiro passo consiste em ordenar descendentemente o conjunto de soluções que será representado por P de acordo com a importância do valor da primeira função objetivo. Posteriormente, o conjunto é dividido recursivamente em dois subconjuntos E (esquerda) e D (direita). Portanto, isto implica que o subconjunto E é de melhor qualidade que o subconjunto D do ponto de vista da primeira

função objetivo. Assim, é possível verificar o critério de dominância, em relação à segunda função objetivo, entre os subconjuntos D e E. As soluções de D que não são dominadas por qualquer membro de E são combinadas com os membros de E para formar um conjunto não-dominado M. A conformação do conjunto M e a verificação de dominância têm lugar no momento em que o tamanho de E e D são iguais a 1, ou seja, até que as divisões recursivas dos subconjuntos permitam comparar só uma solução do conjunto E com uma do conjunto D.

Dos três métodos apresentados por Deb (2004), o próprio autor afirma que o método de Kung não é fácil de ser visualizado como os outros dois métodos restantes. No entanto, como já discutido, o método de Kung é o mais eficiente computacionalmente.

Em Peñuela e Granada (2007), são apresentados um diagrama de fluxo e um pseudocódigo que juntos sintetizam os procedimentos do Método de Kung que também é chamado de Bissecção Recursiva. Para apresentar este diagrama e o pseudocódigo, os autores consideram a definição dos seguintes parâmetros:

V: Matriz que contêm os valores de todas as funções objetivos para cada uma das soluções do conjunto.

N: Número de soluções do conjunto a serem avaliadas.

P: Vetor com elementos de 1 a N usado para identificar cada solução a ser avaliada.

M: Número de funções objetivos do problema.

TipoOt: Vetor que define o tipo de otimização (minimização=0 ou maximização=1) de cada uma das funções objetivos. Por exemplo, TipoOt=[0, 1] significa que F_{obj1} é de minimização e F_{obj2} é de maximização.

A figura 3.12 e a tabela 3.1 mostram, respectivamente, o diagrama de fluxo e o pseudocódigo do método da bissecção recursiva:

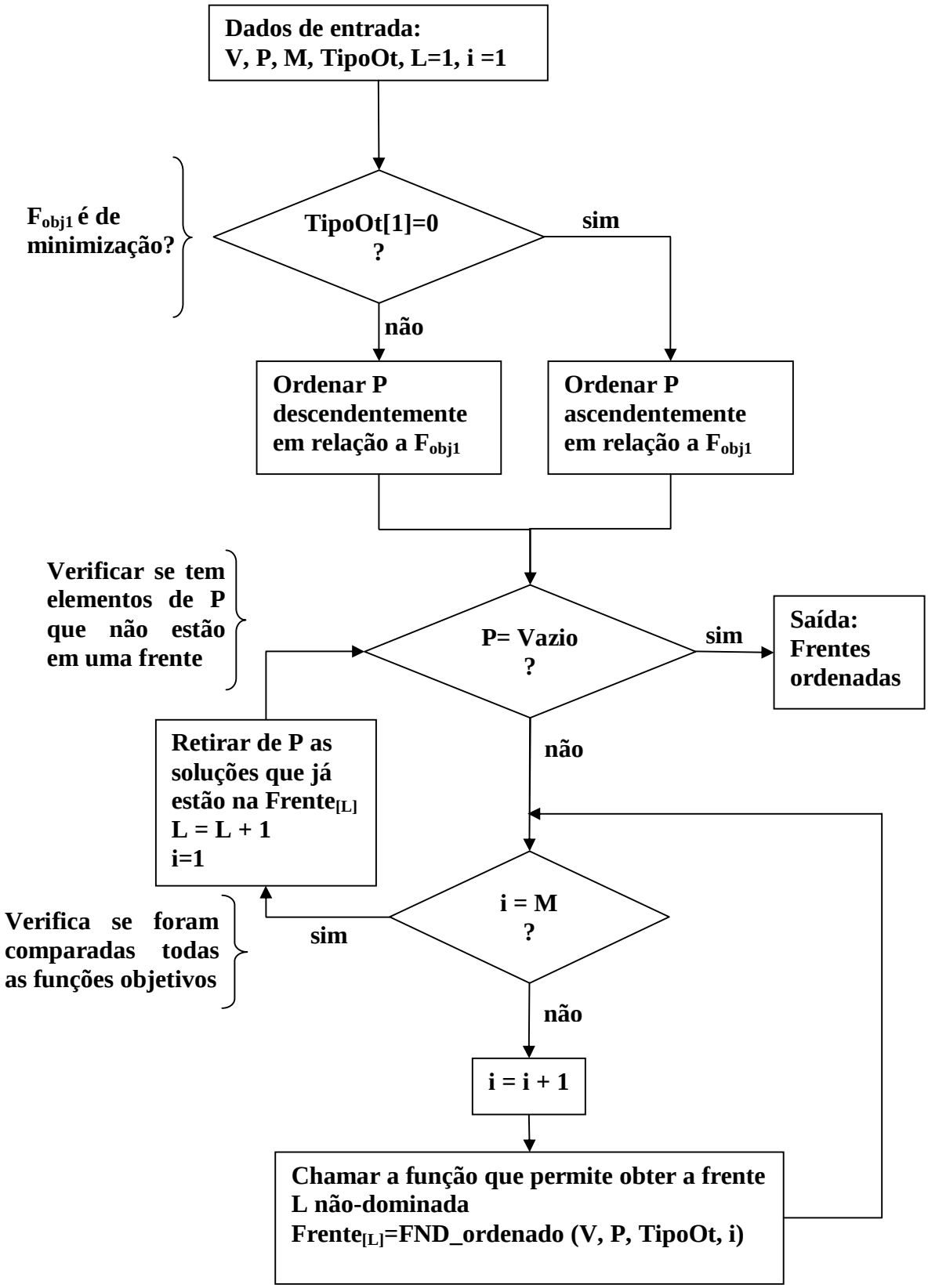

Figura 3.12: Diagrama de fluxo do método da bissecção recursiva.

Tabela 3.1: Pseudocódigo do método da bissecção recursiva.

Método de Kung (bissecção recursiva)

```
1.  function [Frente] = FND_ordenado(V, P,TipoOt, i)
    % A entrada i corresponde a i-ésima função objetivo com
    % a que se comparará o conjunto atual.
2.  N=size(P,1);
3.  PosDividir=round((N)/2);
4.  if N> 1
     % Divide-se a frente e chama a funcão recursivamente.
5.   [E]= FND_ordenado (V,P ([1:PosDividir],:), TipoOt, i);
6.   [D]= FND_ordenado (V,P([PosDividir+1:N],:), TipoOt, i);
7.   if TipoOt(i)==0      % Se o problema é de minimização
8.    if V(D(size(D,1)),i)<=V(E(size(I,1)),i)   % Se D domina E
9.     M=[E;D];
10.       else
11.         M=[E];
12.       end
13.     end
14.     if TipoOt(i)==1    % Se o problema é de maximização
15.      if V(D(size(D,1)),i)>=V(E(size(I,1)),i) % Se D domina E
16.       M=[E;D];
17.      else
18.       M=[E];
19.      end
20.     end
21.     Frente=M;
22.   else
23.     Frente=P;
24.   end
```

Como exemplo, considere a tabela seguinte contendo um conjunto de soluções (vetor P) e os seus respectivos valores das funções objetivo (Matriz V).

Tabela 3.2: Conjunto de soluções e seus respectivos valores de função objetivo.

Vetor P	**Matriz V**	
Soluções	**Função Objetivo 1 (F_{obj1})**	**Função Objetivo 2 (F_{obj2})**
1	**0,2285**	**11,5128**
2	**0,3902**	**7,4299**
3	**0,7559**	**7,1822**
4	**0,1688**	**28,4664**
5	**0,4570**	**11,8462**
6	**0,4078**	**8,0556**
7	**0,3727**	**7,7673**

8	**0,1195**	**26,9118**
9	**0,7348**	**6,5577**
10	**0,2813**	**8,2604**
11	**0,6785**	**3,8630**
12	**0,8121**	**3,5582**

Assumindo que ambas as funções objetivos são de minimização e, utilizando o Método de Kung apresentado, são obtidos os seguintes resultados mostrados na figura 3.13 que segue.

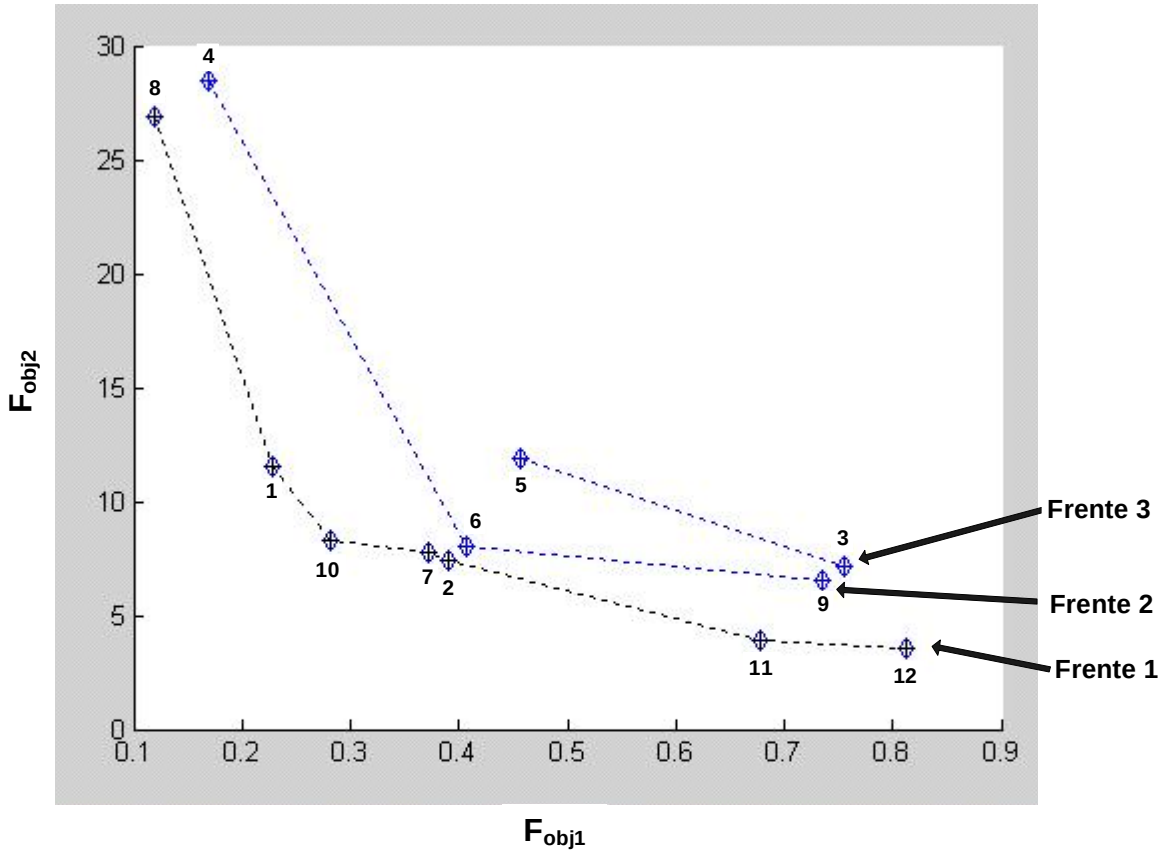

Figura 3.13: Organização das alternativas de solução em frentes de dominância.

Observando a figura 3.13, percebe-se que as soluções foram divididas em frentes de dominância. Sendo assim, este algoritmo apresenta uma característica importante, pois além de encontrar o conjunto de soluções não-dominadas (Frente 1 = {8, 1, 10, 7, 2, 11, 12}) de todo o conjunto P, ele também permite organizar as soluções em frentes que indicam o nível ou classe de dominância que uma solução possui em relação às outras. A idéia para se obter as

demais frentes das soluções que não fazem parte do conjunto não-dominado (Frente 1), consiste em retirar do conjunto P as soluções que já estão alocadas em uma frente e repetir o processo da bissecção recursiva (PEÑUELA; GRANADA, 2007).

Obter as soluções organizadas em frentes de dominância é muito importante, pois existem algoritmos que exigem a classificação das soluções de todo o conjunto em vários níveis de dominância. Desta forma, é possível avaliar a qualidade da solução dependendo da frente a qual pertencem. No caso dos Algoritmos Genéticos, um conjunto de soluções equivale a uma população e cada solução, representa um indivíduo da população. Sendo assim, a organização dos indivíduos em frente de dominância se faz da mesma forma bastando, simplesmente, manter a analogia entre os termos apresentados. No desenvolvimento principal deste trabalho, utiliza-se um Algoritmo Genético que se baseia nesta mesma idéia, ou seja, é feita a classificação da população em frentes de dominância para avaliar a qualidade dos indivíduos.

3.6 Métodos Clássicos de Otimização Multiobjetivo

Nesta seção, descrevem-se alguns métodos de otimização multiobjetivo. A utilização do termo métodos clássicos, utiliza-se principalmente para distingui-los das técnicas evolucionárias. A literatura dispõe de vários métodos clássicos para tratar um problema de otimização multiobjetivo. Em Deb (2004) e Coello (1996), por exemplo, são apresentados alguns destes métodos.

Os métodos clássicos transformam o problema de otimização multiobjetivo em um problema mono-objetivo, incorporando informações subjetivas adicionais. Desta forma, o problema equivalente possui algumas informações adicionais e para sua formulação e resolução, é necessária a definição de alguns parâmetros. A seguir são descritos alguns destes métodos.

3.6.1 Método da Soma Ponderada

Um dos métodos clássicos mais simples é o Método da Soma Ponderada. Este método consiste em formar uma única função objetivo composta da soma ponderada das funções objetivo do problema de otimização multiobjetivo original. Para isso, é necessária a atribuição de pesos (parâmetros), que são valores proporcionais ao nível de preferência atribuída à respectiva função objetivo. Sendo assim, a função objetivo unificada é expressa por:

$$F(\vec{x}) = \sum_{k=1}^{l} w_k f_k(\vec{x}) \tag{3.6}$$

sendo: $$\sum_{k=1}^{l} w_k = 1 \qquad \text{e} \qquad w_k \in [0,1] \tag{3.7}$$

Considerando, por exemplo, um problema multiobjetivo com duas funções objetivos ($f_1(\vec{x})$ e $f_2(\vec{x})$) e seus respectivos pesos w_1 e w_2 tem-se a minimização da seguinte função:

$$F(\vec{x}) = w_1 f_1(\vec{x}) + w_2 f_2(\vec{x}) \qquad \vec{x} \in S \tag{3.8}$$

A figura 3.14 ilustra, hipoteticamente, o espaço objetivo e as soluções Pareto ótimas para este problema. Na expressão (3.8) a função F é uma combinação linear de ambos os objetivos f_1 e f_2. Desta forma, as linhas de contorno da função F no espaço objetivo é uma reta. De fato, dividindo a expressão (3.8) por w_2 e isolando a função objetivo f_2 temos:

$$f_2(\vec{x}) = -\frac{w_1}{w_2} f_1(\vec{x}) + \frac{F(\vec{x})}{w_2} \tag{3.9}$$

Na expressão (3.9) a parcela $\frac{F(\vec{x})}{w_2}$ representa o coeficiente linear da reta e desta forma, conhecendo-se os pesos w_1, w_2 e atribuindo valores a F são obtidos diferentes retas no espaço objetivo. As retas pontilhadas a, b, c e d na figura 3.14 representam diferentes retas correspondentes a diferentes valores de F. Sendo assim, qualquer solução no espaço objetivo

pertencente à reta terá o mesmo valor de F. Pode-se observar, por exemplo, que para um valor de F temos a reta a, enquanto que, um valor menor de F resulta na reta b.

Neste problema representado pela expressão (3.8), queremos minimizar o valor de F e, portanto, o intuito é obter uma reta com o valor mínimo de F. Isto acontece com a reta que é tangente ao espaço objetivo. Na figura 3.14 esta reta é representada por d. Sendo assim, o ponto A é a solução mínima de F e, conseqüentemente a solução Pareto ótima correspondente aos pesos adotados.

É importante observar que na expressão (3.9) o coeficiente $-\frac{w_1}{w_2}$ representa o coeficiente angular da reta e, portanto, se diferentes pesos forem utilizados as retas que representam as linhas de contorno de F terão diferentes inclinações e produzirão, desta forma, diferentes soluções ótimas na fronteira de Pareto.

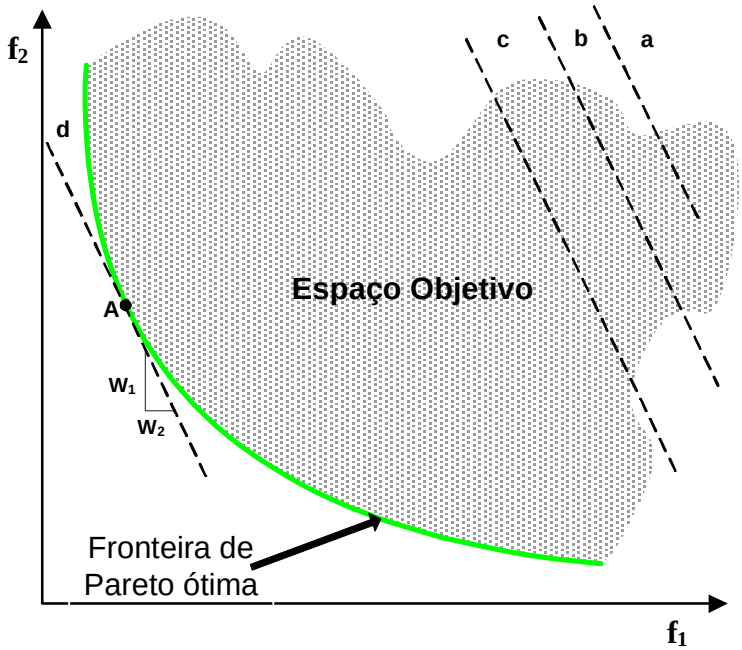

Figura 3.14: Interpretação gráfica do método da soma ponderada.

É interessante notar que esse tipo de abordagem é adequado apenas quando o espaço objetivo é uma região convexa[1] na fronteira de Pareto e, sendo assim, é possível provar que

[1] Dizer sem formalidades que uma região é convexa é o mesmo que afirmar que: dados dois pontos quaisquer pertencente a esta região o segmento de reta que une tais pontos deve estar inteiramente contido nesta região. Caso contrário, a região é não-convexa.

todos os pontos desta fronteira podem ser obtidos variando-se os pesos (DEB, 2004). A figura 3.15 representa um espaço objetivo não-convexo na fronteira de Pareto. Nesta figura, a fronteira de Pareto foi dividida nas regiões AB, BD, DC e CE. Neste caso, a região entre BDC do espaço objetivo não poderá ser encontrada pelo método exposto e, desta forma, algumas soluções que poderiam vir a ser interessantes serão perdidas. Isto ocorre porque nesta região as retas que representam as linhas de contorno de F encontrarão mais de um ponto no espaço objetivo e não serão tangentes e, sendo assim, será possível encontrar uma reta tangente que produzirá um menor valor para a função F. Na figura pode-se verificar que a reta "c" passa pelos pontos D e C na região BDC e não é tangente, no entanto, a reta "b" que é paralela a "c" encontrará o ponto B tangente ao espaço objetivo que produzirá um menor valor para a função F.

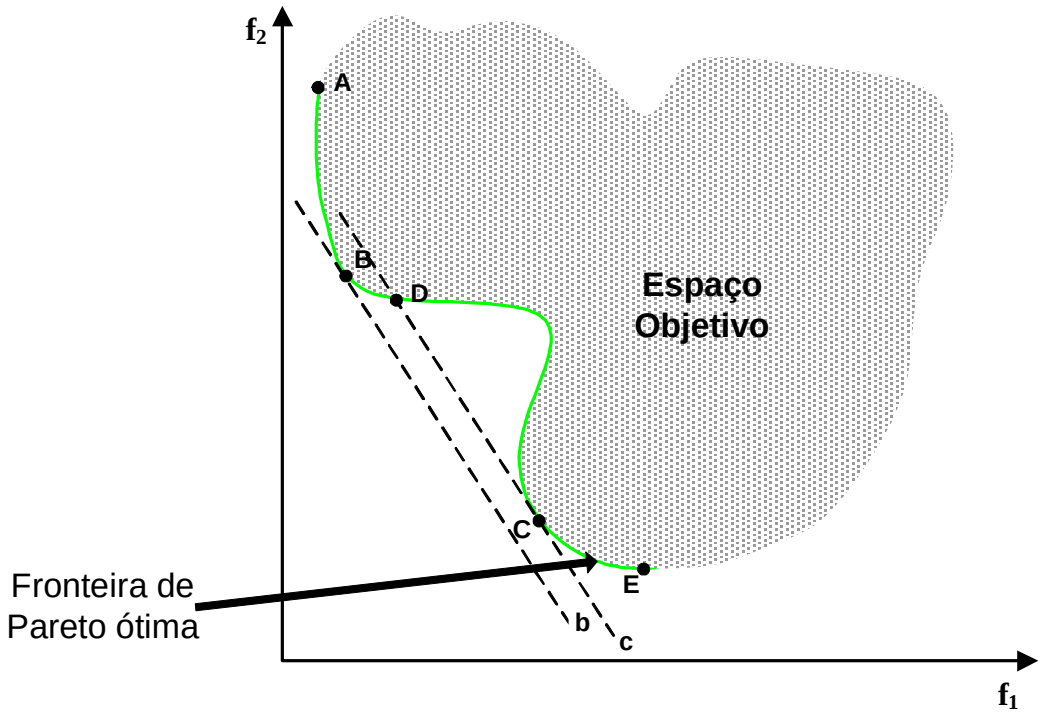

Figura 3.15: Região da fronteira de Pareto não alcançado pelo método da soma ponderada.

3.6.2 Método da Restrição-*e*

Para atenuar as dificuldades enfrentadas pelo método da soma ponderada em resolver problemas que possuem a região do espaço objetivo não-convexa, o método da restrição-*e*

pode ser usado (DEB, 2004). Este método consiste em tratar todas as funções objetivos como restrições em níveis e_k, a menos uma que será minimizada. Sendo assim, o método consiste em minimizar o objetivo de maior importância sujeito a limitação e_k dos outros objetivos. Deste modo, a formulação do problema para o método restrição-e é a seguinte:

$$\begin{aligned} &Min \quad f_m(\vec{x}) \\ &\text{sujeito a} \\ &f_k(\vec{x}) \le e_k \qquad k = 1, 2, \mathbf{K}, l \ \text{ e } \ k \ne m \\ &x \in S \end{aligned} \tag{3.10}$$

Na formulação acima, o parâmetro e_k representa um limitante superior dos valores de f_k e não é, necessariamente, um valor próximo de zero. Para o entendimento do presente método, a figura 3.16 mostra a representação de um problema com dois objetivos f_1 e f_2. Nesta figura é considerado a minimização de f_2 e o objetivo f_1 como uma restrição $f_1(\vec{x}) \le e_1$.

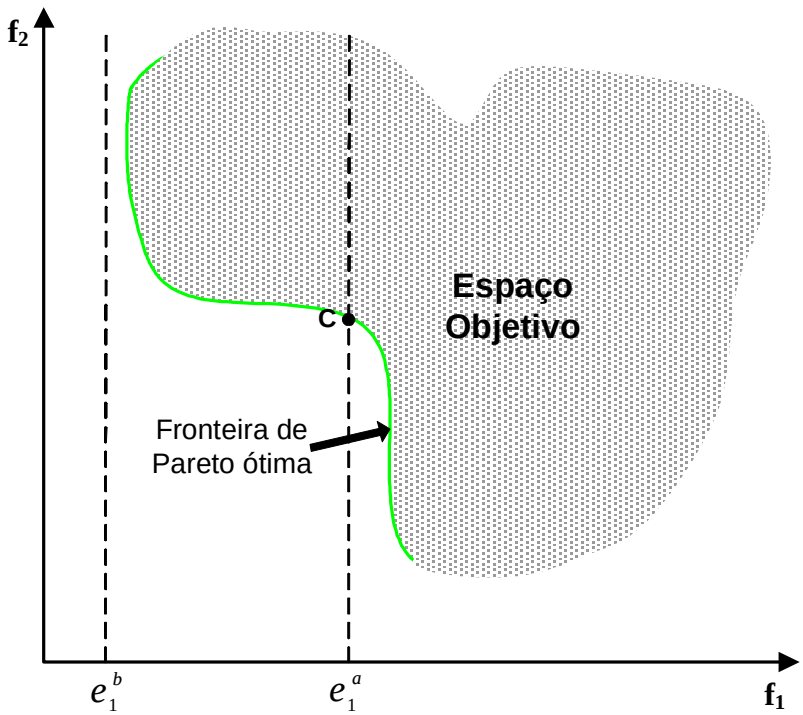

Figura 3.16: Representação do método da restrição-e .

Considerando como limitante superior de f_1 o valor e_1^a o espaço objetivo é dividido em duas partes e a região à esquerda da reta pontilhada correspondente a este limitante torna-se a região factível. Desta forma, a tarefa é encontrar a solução que minimiza f_2 nesta região factível. Fica claro que a solução procurada é a C.

Pode-se perceber na figura 3.16 que a escolha do limitante e_1^b para f_1 não é uma boa opção, visto que, para este valor a região factível para o problema é vazia e, desta forma, o problema não terá solução.

3.6.3 Métodos de Programação por Metas

A idéia principal dos métodos de programação por metas é encontrar soluções que alcancem uma referência (objetivo predefinido) para uma ou mais funções objetivos. Se não existem estas soluções, a tarefa será encontrar soluções onde a diferença com a referência estipulada seja mínima. Por outro lado, se a solução com valor de função objetivo igual ao da referência existe, a tarefa da programação por metas será identificar esta solução.

Em Deb (2004), são apresentados três métodos de programação por metas que são: Método de Programação por Metas Ponderado, Lexicográfico e Min-Max. Neste texto apresenta-se o Método Lexicográfico, visto que, será utilizada a idéia deste método junto com os Algoritmos Genéticos para alcançar os resultados que serão apresentados.

Método Lexicográfico

Neste método, diferentes metas são categorizadas dentro de muitos níveis de prioridade. Uma meta do primeiro nível de prioridade é mais importante que uma meta do segundo nível de prioridade. Assim, é importante cumprir as metas do primeiro nível de prioridade antes de considerar as metas do segundo nível de prioridade.

Este método formula e resolve um número seqüencial de problemas de programação por metas. Sendo assim, primeiro consideram-se somente as metas e as correspondentes restrições do primeiro nível de prioridade na formulação do problema de programação por

metas e, em seguida, este problema é resolvido. Se existem múltiplas soluções para o problema anterior, outro problema de programação por metas é formulado considerando desta vez o segundo nível de prioridade. As metas do primeiro nível de prioridade são usadas como restrições de igualdade para assegurar que as soluções do segundo nível de prioridade não violem as restrições do primeiro nível de prioridade. O procedimento repete-se seqüencialmente para os outros níveis de prioridade (DEB, 2004; GRANADA et al., 2008).

A figura 3.17, ilustra o princípio de funcionamento do método lexicográfico da programação por metas para um problema de minimização com dois objetivos f_1 e f_2.

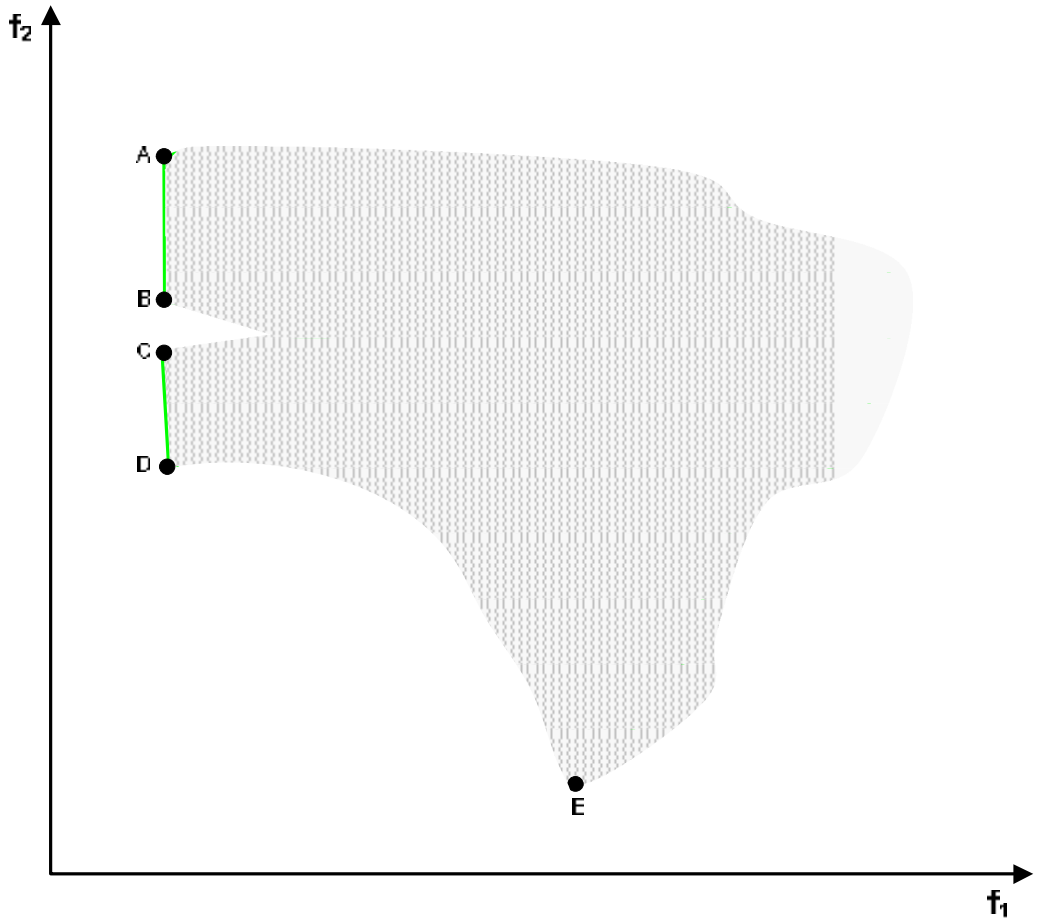

Figura 3.17: Método lexicográfico da programação por metas.

Nesta figura, considerando que o primeiro nível de prioridade é minimizar f_1, ou seja, atingir a meta para f_1 é mais importante que f_2, o procedimento consiste em primeiro minimizar f_1 e ignorar f_2. Desta forma, encontra-se o conjunto de soluções representados pelos segmentos AB e CD para o primeiro nível de prioridade. A solução para o segundo nível de prioridade será aquela que minimiza f_2 e faça parte de f_1. Neste caso, a solução para o segundo nível de prioridade será a representada pelo ponto D. Esta última solução será a solução de todo o problema de programação por metas lexicográfico.

É importante observar que dando maior prioridade para a meta de minimizar f_2 do que f_1, a solução do problema muda para a solução representada na figura pelo ponto E.

3.6.4 Vantagens e Desvantagens dos Métodos Clássicos

Nos métodos clássicos, a vantagem principal é a existência de provas que garantem a convergência para soluções Pareto ótimas. No entanto, os métodos clássicos utilizam a idéia da abordagem 1 apresentada na seção 3.2, ou seja, o problema de otimização multiobjetivo original é transformado em um problema de otimização mono-objetivo. Na modificação de um problema multiobjetivo para um mono-objetivo surge a desvantagem da necessidade de introdução de parâmetros adicionais que afetam diretamente os resultados obtidos.

No capítulo a seguir, é apresentada a teoria dos Algoritmos Evolucionários em especial a dos Algoritmos Genéticos, visto que, esta técnica é bastante poderosa e vem sendo amplamente aplicada em vários problemas de otimização.

CAPÍTULO 4

ALGORITMOS EVOLUCIONÁRIOS

4.1 Introdução

No capítulo 2 deste trabalho foi apresentada uma possível divisão dos Algoritmos Evolucionários em: Estratégias Evolucionárias, Algoritmos Genéticos e Programação Evolutiva. Em Deb (2004), além da teoria dos Algoritmos Evolucionários citados, são apresentados os conceitos de Programação Genética que também se enquadra neste ramo. Sendo assim, existem diferentes tipos de Algoritmos Evolucionários e a idéia envolvida na teoria destes algoritmos está relacionada em utilizar conceitos e princípios da evolução natural das espécies como estratégia de otimização de problemas. Apesar do presente capítulo tratar de Algoritmos Evolucionários, aqui será apresentado somente um desses algoritmos: o Algoritmo Genético. O intuito de dar ênfase somente a este algoritmo se deve ao fato que a base principal do desenvolvimento deste trabalho se apóia nesta técnica. Desta forma, neste capítulo são apresentados os principais fundamentos da técnica da qual se denomina Algoritmos Genéticos. Inicialmente, apresenta-se uma estrutura básica e mais usual da forma com que esta técnica foi ou vem sendo originalmente empregada, visto que, novas estratégias podem ser adicionadas em conjunto com a técnica mais básica dos Algoritmos Genéticos com o intuito de promover novas melhorias e dar origem a Algoritmos Genéticos mais elaborados. No decorrer deste capítulo, são explicitados de forma sucinta as origens, definições,

procedimentos, forma de codificação de uma configuração, vantagens, desvantagens e outros tópicos importantes relacionados a esta técnica.

Apresentam-se também as implicações da mudança do Algoritmo Genético mais básico para uma codificação com parâmetros reais.

4.2 Algoritmos Genéticos

O Algoritmo Genético é um ramo dos algoritmos evolucionários e como tal pode ser definido como uma técnica de busca que se baseia no processo da evolução natural.

Analisando invenções como, por exemplo, o avião; podem-se perceber características semelhantes à de um pássaro nesta invenção. Há um bom tempo o homem vem se baseando nas características e princípios da natureza para a criação de máquinas, métodos e técnicas que na maioria das vezes visam melhorar ou facilitar a vida em nosso cotidiano (CASTRO, 2001).

Dentro destes conceitos, fundamentado no processo de seleção natural proposto por Darwin e nos mecanismos da genética surge a teoria dos Algoritmos Genéticos. Este algoritmo foi inventado por Holland na década de 70 (ROMERO, 2005). Holland estudou formalmente a evolução das espécies e propôs um modelo computacional heurístico[1] que, quando implementado, poderia oferecer soluções de boa qualidade para problemas extremamente difíceis que não podiam ser resolvidos computacionalmente até aquela época (LINDEN, 2006).

Os Algoritmos Genéticos são técnicas inspiradas na teoria de Darwin, ou seja, são técnicas de buscas baseadas na teoria da evolução, combinando a sobrevivência dos mais aptos com a troca de informações de uma forma estruturada, onde um problema do mundo real é modelado através de um conjunto de indivíduos que são soluções potenciais que melhor se ajustam ao ambiente.

De forma análoga com a teoria da evolução, esses indivíduos são selecionados, se reproduzem e sofrem mutação, obtendo deste modo uma nova geração de indivíduos que

[1] Heurísticas são algoritmos que encontram soluções de boa qualidade para problemas complexos. Um algoritmo heurístico é um conjunto de procedimentos simples, muitas vezes baseados no senso comum, que encontram soluções de boa qualidade (não necessariamente a ótima) de maneira simples e rápida.

também atendem as necessidades do ambiente. Após certo número de gerações espera-se convergir para uma geração de elite que corresponda a uma solução ótima ou quase ótima para o problema.

Desde então, os Algoritmos Genéticos começaram a se expandir por toda a comunidade científica, gerando uma série de aplicações que puderam ajudar a resolver problemas extremamente importantes. Uma grande aplicação dos Algoritmos Genéticos está em problemas de busca, onde dado um conjunto de elementos ou indivíduos, deseja-se encontrar aquele ou aqueles que melhor atendam a certas condições especificadas (CASTRO, 2001).

4.2.1 Definições

Como visto anteriormente, os algoritmos genéticos se utilizam dos conceitos da genética para simular a evolução das populações. Para melhor entender o conceito dos algoritmos genéticos, faz-se uma analogia entre os termos usados na biologia e o módulo computacional referente ao estudo dos algoritmos genéticos.

A complexidade utilizada pela natureza para formar novas gerações é realizada através da reprodução que existem em dois tipos distintos (LINDEN, 2006):

Assexuada: realizada por organismos inferiores, como as bactérias;

Sexuada: que exige a presença de dois organismos, na maioria das vezes de sexos opostos, que fazem a troca entre si do material genético;

Na reprodução sexuada os elementos que fazem parte desse processo são: genética, cromossomos, genes e alelos.

No desenvolvimento da técnica usada em algoritmos genéticos os elementos que participam desse processo são: Problemas de otimização, indivíduos, variáveis e valor das variáveis.

Para os referidos processos foi criada a seguinte correspondência biunívoca (ROMERO, 2005):

Reprodução Sexuada	$\Leftrightarrow$	**Algoritmo Genético**
Genética	$\Leftrightarrow$	Problemas de otimização
Cromossomos	$\Leftrightarrow$	Indivíduos
Genes	$\Leftrightarrow$	Variáveis
Alelos	$\Leftrightarrow$	Valor das variáveis

Desta forma, nos algoritmos genéticos, podem-se apresentar as principais definições que estão associadas aos termos utilizados na biologia (CASTRO, 2001):

- *Cromossomos ou Indivíduos:* representa uma cadeia de caracteres representando alguma informação relativa às variáveis do problema. Cada indíviduo representa deste modo uma solução do problema;

- *Genes ou variáveis:* É a unidade básica do indivíduo. Cada indivíduo tem certos números de genes, cada um descrevendo certa variável do problema;

- *População:* Conjunto de indivíduos ou soluções;

- *Geração:* O número de iterações que o algoritmo genético executa;

- *Operações Genéticas:* Operações que o algoritmo genético realiza sobre cada um dos indivíduos;

- *Região Factível:* É o conjunto, espaço ou região que compreende as soluções factíveis do problema a ser otimizado. Deve ser caracterizado pelas funções de restrições, que definem as soluções factíveis do problema a ser resolvido.

- *Função Objetivo:* É a função que se deseja otimizar. Esta função contém a informação do desempenho de cada indivíduo na população. Nesta função estão representadas as características do problema que o algoritmo genético necessita para realizar seu objetivo, sendo usualmente expressa como:

$$F = f(x_1, x_2, \mathbf{K}, x_n)$$

sendo $x_1, x_2, \mathbf{K}, x_n$ as variáveis que o algoritmo procura determinar para otimizar *F*.

4.2.2 Estrutura Básica do Algoritmo Genético

Para implementar um algoritmo genético, devem-se realizar os seguintes passos (ROMERO, 2005, POZO et al.):

- Escolhe-se adequadamente uma representação para os indivíduos da população. Geralmente, esta representação é feita utilizando a codificação binária, pois a aplicação dos operadores genéticos de recombinação e mutação são mais simples de serem empregadas. Posteriormente, escolhe-se uma população inicial, normalmente formada por indivíduos criados aleatoriamente;

- Avalia-se adequadamente toda população segundo algum critério determinado por uma função que mede a qualidade do individuo (função de aptidão ou “fitness”). Desta forma, os melhores indivíduos são aqueles que apresentam função de aptidão de melhor qualidade;

- Estabelece-se uma estratégia de seleção dos indivíduos como base para criação de um novo conjunto de indivíduos (nova população);

- Estabelece-se um mecanismo que permita implementar os operadores genéticos de recombinação e mutação. A nova população é obtida aplicando sobre os indivíduos selecionados os referidos operadores;

- Repetem-se os passos acima até que: um indivíduo de qualidade aceitável seja encontrado, um número preestabelecido de passos seja atingido ou o algoritmo não consiga mais mostrar evolução, ou seja, não se consegue melhorar a incumbente já encontrada;

O fluxograma a seguir representa a estrutura básica de um algoritmo genético:

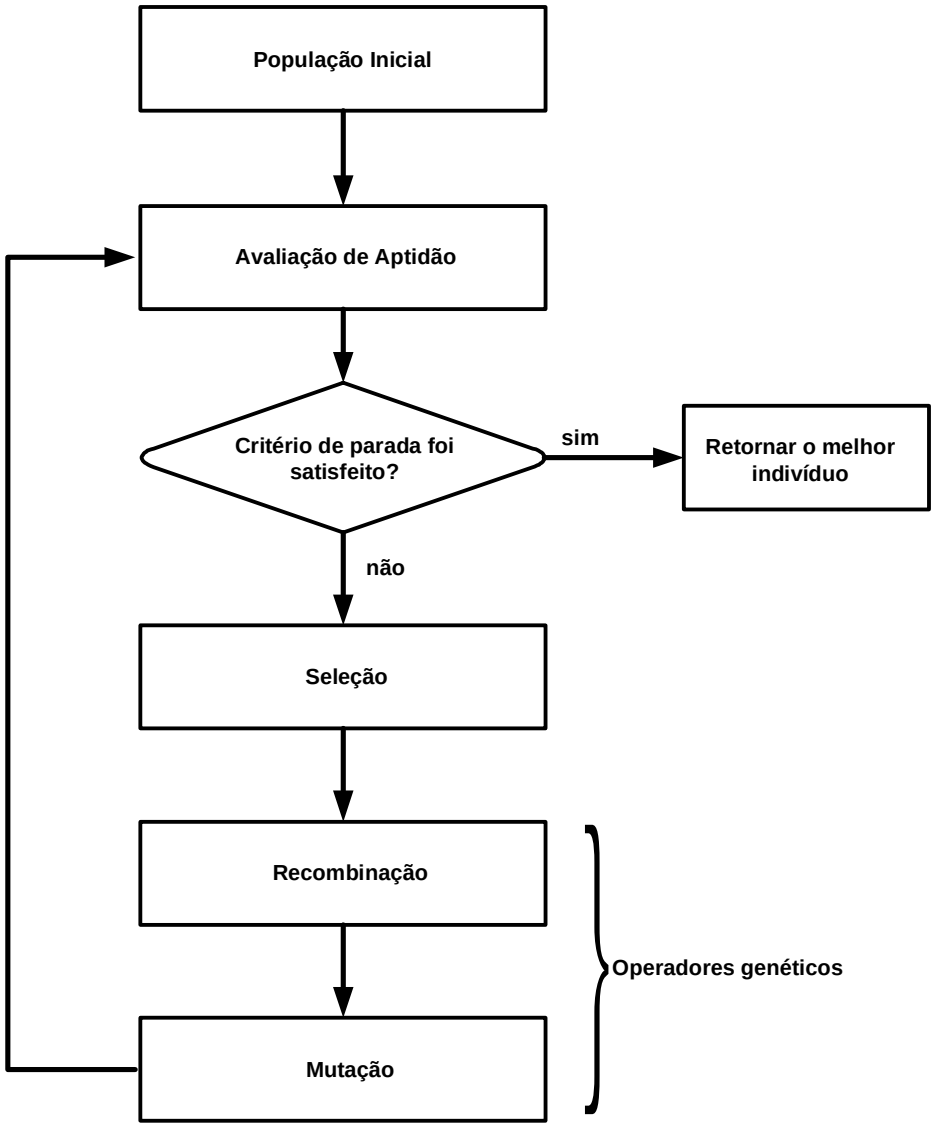

Figura 4.1: Estrutura básica de um algoritmo genético

4.2.3 Representação ou Codificação

A representação ou codificação das variáveis de um problema a ser otimizado proporciona um grande impacto no desempenho de um algoritmo genético, devendo desta forma, ser a mais simples possível sem perder, no entanto, as características de representação do problema tratado. Existem inúmeras formas de representação das variáveis de um problema, tais como: binária, números inteiros ou reais (CASTRO, 2001).

A maioria dos trabalhos desenvolvidos utiliza a codificação binária, onde cada indivíduo é representado por um vetor composto por 0 e 1. Um algoritmo genético básico exige que a representação do indivíduo seja com a codificação binária. Sendo assim, problemas que possuem variáveis inteiras ou reais são transformados, de alguma maneira, em codificação binária. A principal justificativa em se utilizar a codificação binária se deve, principalmente, ao fato da teoria básica de implementação dos operadores genéticos e as características de convergência ter sido baseada em cima da codificação binária. Outra justificativa é que a codificação binária imita os dois tipos de alelos existentes num gene de um cromossomo (ROMERO, 2005).

Infelizmente, a codificação binária traz alguns problemas. Neste trabalho, não se discute estes problemas nem as formas utilizadas para evitar que eles aconteçam. No entanto, em (RENDÓN et al., 2006) pode-se encontrar os tipos e as maneiras de contornar os problemas gerados pela codificação binária.

Por fim, uma vez definidas as variáveis relevantes para a resolução de um problema de otimização, bem como a forma com que estas variáveis serão representadas, devem-se justapor estas variáveis de maneira a formarem os indivíduos.

4.2.4 Geração da População Inicial

A população inicial de indivíduos é na maioria das vezes gerada de forma aleatória. No entanto, existem ocasiões onde é mais apropriada uma geração da população inicial utilizando uma heurística. Assim, logo no início, podem-se introduzir na população inicial indivíduos

com características interessantes, como por exemplo, acrescentar soluções aproximadas conhecidas contendo algum tipo de informação prévia. Existem diversos trabalhos que comprovam que a geração da população inicial não é uma fase crítica em algoritmos genéticos, no entanto, é necessário que a população inicial contenha indivíduos suficientemente diversificados (CASTRO, 2001).

4.2.5 Avaliação da População

A avaliação da população é realizada utilizando uma função de aptidão, que deve indicar a qualidade de cada indivíduo na população. Para problemas de otimização esta função está intimamente ligada à função objetivo.

Após a etapa de geração da população inicial, é necessário avaliar os indivíduos que foram criados e, desta forma, deve-se selecionar alguns indivíduos para dar continuidade à criação de outros que possam evoluir suas características. A avaliação da qualidade dos indivíduos da população é a componente mais importante em qualquer algoritmo genético, pois é através desta que se mede quão próximo um indivíduo está da solução desejada ou quão boa é esta solução.

É essencial que a função que avalia a qualidade dos indivíduos seja muito representativa e diferencie na proporção correta as más soluções das boas. Se houver pouca precisão na avaliação, uma solução ótima pode ser posta de lado durante a execução do algoritmo, além de gastar um maior tempo computacional explorando soluções pouco promissoras (POZO et al.).

4.2.6 Seleção dos Indivíduos

Após a etapa de avaliação dos indivíduos de uma população tem que se criar um mecanismo que transmita a hereditariedade deles às populações seguintes, preservando suas boas características. Nesta etapa os indivíduos são escolhidos e, posteriormente, recombinados.

O processo de seleção está baseado no princípio da sobrevivência dos melhores indivíduos, ou seja, os indivíduos com melhor aptidão recebem uma maior probabilidade de serem copiados para fazer parte da geração da nova população.

Existem vários métodos para selecionar os indivíduos sobre os quais serão aplicados os operadores genéticos. A seguir, apresentam-se resumidamente alguns destes métodos:

Seleção por ranking: os indivíduos da população são ordenados de acordo com seu valor de adequação e então sua probabilidade de escolha é atribuída conforme a posição que ocupam;

Seleção por giro de roleta: cada indivíduo da população é representado na roleta proporcionalmente ao seu índice de aptidão. Assim, para indivíduos com alta aptidão é dada uma porção maior da roleta, enquanto aos indivíduos de aptidão mais baixa, é dada uma porção relativamente menor;

Seleção por torneio: considera-se este método bastante interessante e, na implementação do algoritmo que será apresentado aqui, utiliza-se a idéia deste método. Assim, será dada maior ênfase em sua apresentação. Este método é um dos mais simples de implementar computacionalmente e, além disso, têm encontrado bons resultados.

A idéia deste método é promover um torneio entre um grupo de n ($n \geq 2$) indivíduos aleatoriamente tomados na população. Assim, o indivíduo que vencer este torneio (indivíduo com melhor valor de aptidão entre o grupo), é selecionado para fazer parte da geração da nova população, enquanto os demais indivíduos do grupo são descartados. O processo de seleção termina quando se realiza uma quantidade de torneios igual ao tamanho da população.

Dentre as vantagens do método de seleção por torneio, a seguir citam-se algumas, tais como (CASTRO, 2001):

- não acarreta convergência prematura;
- combate a estagnação;
- nenhum esforço computacional extra é necessário, tal como ranking;

- aptidão explícita é desnecessária;
- inspiração biológica do processo;

4.2.7 Operadores Genéticos

Operadores genéticos são funções que se aplicam às populações, permitindo obter novas populações. Após a seleção dos indivíduos em uma população, realiza-se a recombinação e/ou mutação desses, obtendo assim uma nova população com melhores indivíduos ou não. Através de sucessivas gerações, repetem-se esses procedimentos até chegar a um resultado satisfatório a fim de que a população se diversifique e mantenha características de adaptação adquiridas pelas gerações anteriores.

Os algoritmos genéticos básicos são normalmente constituídos de dois operadores: **recombinação** e **mutação**.

O operador de recombinação permite a troca de material genético entre dois indivíduos denominados pais, combinando informações de maneira que exista uma probabilidade razoável dos novos indivíduos produzidos serem melhores que seus pais. Como passo inicial, toda a população selecionada é agrupada aleatoriamente em pares. A recombinação acontece através de um processo de decisão aleatório, ou seja, escolhe-se uma taxa de recombinação (r_r) e gera-se um número aleatório para cada par. Logo, se o valor aleatório gerado for inferior a r_r a recombinação é permitida; caso contrário, os pares são mantidos inalterados (CASTRO, 2001). Os mais conhecidos operadores de *recombinação* são:

- **Operadores de um ponto:** Cada par de indivíduos a serem recombinados são particionados em um ponto, chamado ponto de corte, escolhido aleatoriamente. Um novo indivíduo é gerado permutando a metade inicial de um indivíduo com a metade final do outro. Normalmente este operador apresenta pior desempenho que o multiponto que será visto a seguir.

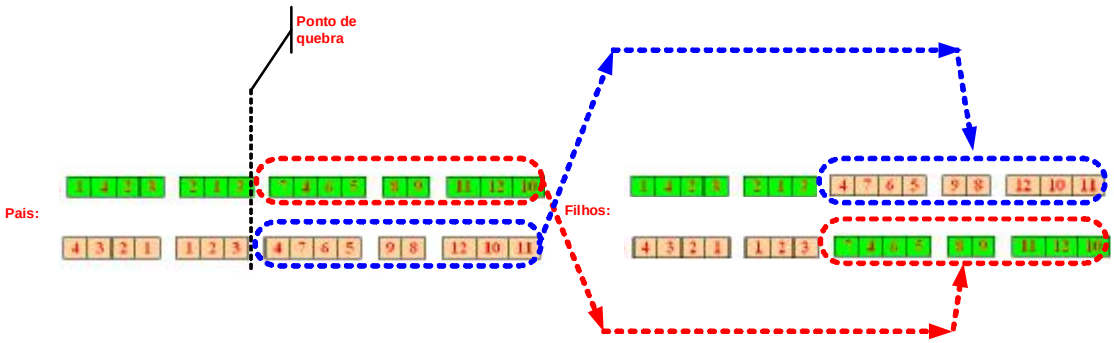

Figura 4.2: Operador de recombinação com um ponto de corte.

- **Operador multiponto:** é uma generalização do operador de um ponto. Nele é escolhido um número *n* de pontos de corte. A figura a seguir mostra um exemplo desta operação:

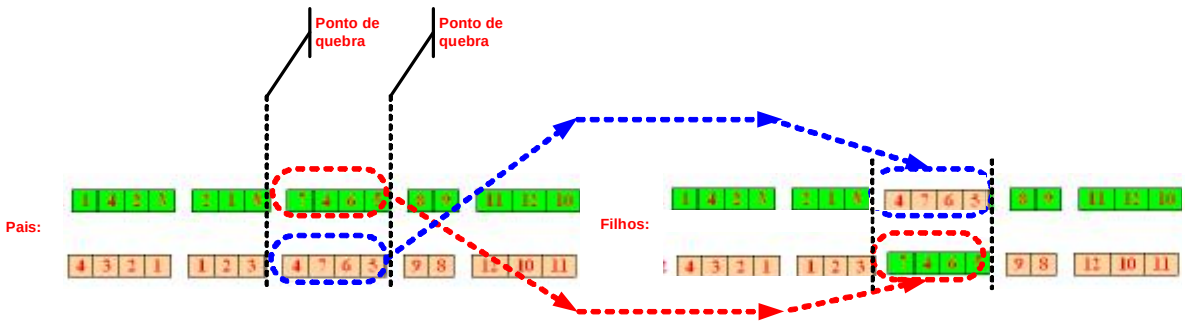

Figura 4.3: Operador de recombinação multiponto com dois pontos de corte.

Após o operador de recombinação, caso haja necessidade de criar uma variabilidade maior entre os descendentes, usa-se o operador genético de mutação, que corresponde a uma pequena alteração aleatória em seu código genético da qual garante que diversas alternativas serão exploradas. Na figura a seguir mostra-se um exemplo de mutação:

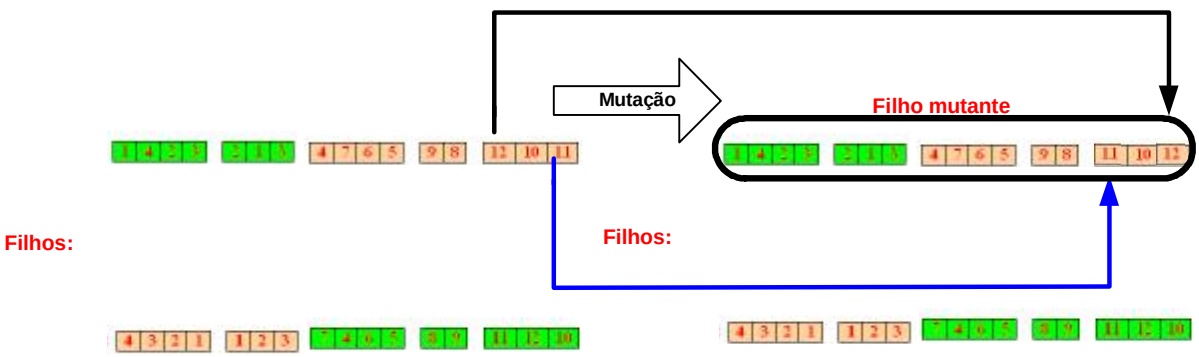

Figura 4.4: Operador de mutação.

A figura 4.4 mostra o caso em que a mutação corresponde à troca entre si de duas posições escolhidas aleatoriamente. Neste exemplo, a codificação não é a binária. No entanto, na mutação para uma representação binária, um bit pode ser invertido de 0 para 1 ou vice-versa, segundo uma probabilidade de mutação r_m.

4.2.8 Elitismo

Com o intuito de aumentar a velocidade de convergência do algoritmo e de preservar e utilizar as melhores soluções encontradas na geração atual para as próximas gerações, surge o conceito de elitismo. Em sua versão mais simples, o processo simplesmente realiza a cópia dos ($N \geq 1$) melhores indivíduos da população corrente para a próxima geração, garantindo que estas soluções não serão destruídas nas etapas de recombinação e mutação. Sendo assim, além dos melhores indivíduos serem passados de uma geração para outra, também participarão da criação de novos indivíduos. A principal vantagem deste método está na garantia de convergência, ou seja, caso o ótimo global seja descoberto durante o processo de busca, o algoritmo genético deve convergir para esta solução. No entanto, devido a presença de uma cópia ou mais dos melhores indivíduos, existe a possibilidade de forçar a busca na direção de algum ótimo local que tenha sido descoberto antes do global (CASTRO, 2001).

4.2.9 Parâmetros de Controle

A utilização correta dos parâmetros de controle é, sem dúvidas, um dos aspectos mais relevantes dentro da estrutura de um algoritmo genético. No entanto, as escolhas destes parâmetros irão depender entre outras coisas da aplicação a ser resolvida. A eficiência e o funcionamento de um algoritmo genético são altamente dependentes dos seus parâmetros de controle e, a seguir, descrevem-se estes parâmetros:

- *Tamanho da População* (n_p)*:* O tamanho da população indica o número de indivíduos em cada população, que é normalmente constante durante a evolução. O tamanho da população pode afetar o desempenho global e a eficiência dos algoritmos genéticos. Populações muito pequenas têm grandes chances de perder a diversidade necessária para convergir a uma boa solução, pois fornecem uma pequena cobertura do espaço de busca do problema. Entretanto, se a população tiver muitos indivíduos, o algoritmo poderá perder grande parte de sua eficiência pela demora em avaliar a

função de aptidão de todo conjunto a cada iteração, além de ser necessário trabalhar com maiores recursos computacionais (POZO et al.). Uma maneira interessante é relacionar o tamanho da população com o tamanho do indivíduo, ou seja, quanto maior for o indivíduo maior deverá ser o tamanho da população. Na literatura especializada, encontram-se valores típicos recomendados de $n_p \in [30; 200]$ (ROMERO, 2005).

- *Taxa de Recombinação* (r_r)*:* Este parâmetro indica com qual taxa ou probabilidade irá ocorrer a recombinação entre os indivíduos selecionados na população. Quanto maior esta taxa, mais rapidamente novas estruturas serão introduzidas na população. No entanto, esta taxa sendo muito alta poderá retirar rapidamente indivíduos de boa qualidade da população. Valores baixos podem tornar a convergência do algoritmo muito lenta (CASTRO, 2001). Na literatura especializada, encontram-se valores típicos recomendados de $r_r \in [0,5; 1,0]$ (ROMERO, 2005).

- *Taxa de Mutação* (r_m)*:* Este parâmetro indica com qual taxa ou probabilidade ocorrerá a mutação nos indivíduos da população ao longo do processo de evolução. A mutação é empregada para fornecer novas informações dentro da população visando, desta forma, aumentar a diversidade populacional. A taxa de mutação possibilita ainda a maior varredura do espaço de busca. No entanto, deve-se tomar o cuidado em utilizar uma taxa de mutação muito alta, pois isto pode tornar a busca essencialmente aleatória (CASTRO, 2001). Na literatura especializada, encontram-se valores típicos recomendados de $r_m \in [0, 001; 0,05]$ (ROMERO, 2005).

4.3 Vantagens e Desvantagens dos Algoritmos Genéticos

Nesta seção, apresentam-se algumas vantagens e desvantagens em se utilizar os Algoritmos Genéticos. Em Linden (2006), são informadas algumas vantagens e desvantagens desta técnica das quais são mostradas a seguir.

4.3.1 Vantagens dos Algoritmos Genéticos

As principais vantagens que pode ser considerada na utilização dos Algoritmos Genéticos são:

- São robustos e aplicáveis a uma grande variedade de problemas;
- Não usam apenas informação local, logo, não ficam presos, necessariamente, a ótimos locais como determinados métodos de busca. Esta característica é uma das mais interessantes dos algoritmos genéticos e fazem com que eles sejam técnicas extremamente adequadas para funções multimodais e de comportamento complexo;
- Seu desempenho não é afetado por descontinuidades na função ou em suas derivadas. Os algoritmos genéticos não usam informações de derivadas na sua evolução nem necessitam de informação dos gradientes da superfície da função objetivo para efetuar a busca. Isto faz com que sejam muito adequados para funções com descontinuidades ou para os quais não temos como calcular a derivada;
- Apresentam um bom desempenho para uma grande escala de problemas;
- São de fácil implementação e proporcionam maior flexibilidade no tratamento do problema a ser resolvido;

4.3.2 Desvantagens dos Algoritmos Genéticos

Apesar dos algoritmos genéticos possuírem uma grande quantidade de vantagens, pode-se também citar algumas desvantagens provenientes da utilização desta técnica:

- Dificuldade de achar o ótimo global exato;
- Requerem um grande número de avaliações de função de aptidão;

§ Grandes possibilidades de configurações que podem complicar a resolução do problema tratado;

As deficiências dos algoritmos genéticos podem ser atenuadas com uma maior consolidação da técnica e com o avanço das capacidades computacionais.

4.4 Algoritmo Genético com Codificação Real

Nas seções anteriores foram apresentados os conceitos básicos de um Algoritmo Genético tradicional. Na lógica deste algoritmo, a codificação das variáveis do problema deve ser feita utilizando uma cadeia binária. Desta forma, para a utilização do Algoritmo Genético tradicional um problema a ser resolvido deve se enquadrar a esta lógica. A vantagem de se utilizar esta lógica, que implica na codificação binária, é a aplicação trivial dos operadores genéticos de recombinação e mutação. No entanto, existem algoritmos genéticos que modificam a lógica do Algoritmo Genético tradicional, como por exemplo, utilizando-se a codificação dos indivíduos com parâmetros reais ao invés de binários.

A dificuldade que surge da utilização da codificação real para um Algoritmo Genético está relacionada com os operadores genéticos de recombinação e mutação. Sendo assim, foi necessário projetar estratégias para simular tais operadores genéticos. Portanto, a utilização da codificação real implica na mudança dos operadores de recombinação e mutação. Em (DEB, 2004) são descritos alguns operadores de recombinação e mutação para a codificação real, tais como:

- Operadores de Recombinação:
  - Recombinação Linear;
  - Recombinação Blend e suas Variantes;
  - Recombinação Binária Simulada;
  - Operador de Recombinação Fuzzy;
  - Recombinação Simplex;
  - Recombinação Baseada em Conectivos Fuzzy; e outras.

- Operadores de Mutação:

- Mutação Aleatória;
- Mutação Não-Uniforme;
- Mutação Polinomial; e outras.

Neste texto, dentre todos estes operadores, apresenta-se apenas um operador de recombinação e outro de mutação, ou seja, serão apresentados os operadores de Recombinação Linear e de Mutação Aleatória. São apresentados este dois operadores, pois na implementação do algoritmo que será proposto neste trabalho utilizou-se a codificação real e esses dois operadores genéticos.

- Recombinação Linear: O operador de Recombinação Linear é bastante simples e consiste em: dados dois indivíduos (pais) $x_i^{(1,t)}$ e $x_i^{(2,t)}$ na geração **t**, os descendentes são obtidos como:

$$0{,}5(x_i^{(1,t)} + x_i^{(2,t)}),\ (1{,}5x_i^{(1,t)} - 0{,}5x_i^{(2,t)}) \text{ e } (-0{,}5x_i^{(1,t)} + 1{,}5x_i^{(2,t)}) \tag{4.1}$$

Desta forma, um dos três descendentes é eliminado por torneio ou aleatoriamente.

- Mutação Aleatória: este é o esquema mais simples de mutação e consiste em criar uma solução $y_i^{(1,t+1)}$ de forma totalmente aleatória considerando todo o espaço de busca. Isso é feito da seguinte maneira:

$$y_i^{(1,t+1)} = r_i(x_i^{(U)} - x_i^{(L)}) \tag{4.2}$$

sendo r_i um número aleatório entre [0, 1] e os subíndices U e L indicam os limites superior e inferior, respectivamente, do espaço de busca.

No capítulo a seguir, será apresentada a forma em que os Algoritmos Evolucionários, em especial os Algoritmos Genéticos, podem ser aplicados em problemas de otimização multiobjetivo.

CAPÍTULO 5

ALGORITMOS EVOLUCIONÁRIOS MULTIOBJETIVOS

5.1 Introdução

No capítulo anterior, foi apresentado o Algoritmo Evolucionário conhecido como Algoritmo Genético. Neste algoritmo pode-se perceber que, na busca da solução ótima do problema, uma população de soluções é processada em cada iteração ou geração. Esta característica do Algoritmo Genético ou de outra técnica evolucionária faz com que estes algoritmos sejam naturalmente adequados para a determinação de várias soluções. Sendo assim, pode-se afirmar que na resolução de problemas de otimização multiobjetivo, os Algoritmos Evolucionários são mais vantajosos em relação, por exemplo, aos métodos clássicos que não foram projetados para trabalharem com múltiplas soluções. Na seção 3.4 do capítulo 3 foram vistos duas metas importantes da otimização multiobjetivo, que são:

1- Encontrar um conjunto de soluções o mais próximo possível da fronteira de Pareto;

2- Encontrar um conjunto de soluções que estejam relativamente bem espaçados.

Em relação à primeira meta, como os Algoritmos Genéticos trabalham com uma população de soluções, é possível realizar algumas mudanças na teoria do Algoritmo Genético básico (visto no capítulo anterior), com o intuito de fazer com que esta técnica obtenha uma população de soluções Pareto ótimas no final de uma única simulação do algoritmo. Aperfeiçoando o Algoritmo Genético para trabalhar neste sentido, evita-se o uso repetitivo da otimização mono-objetivo para obter apenas uma solução Pareto ótima em cada simulação do algoritmo. Outra vantagem deste procedimento é a de eliminar a utilização de, por exemplo, parâmetros (pesos) necessários para transformar um problema de otimização multiobjetivo em mono-objetivo, estratégia que é utilizada pelos métodos clássicos e que fornecem apenas uma única solução Pareto ótima, já que, cada conjunto de pesos está associado a uma solução particular na fronteira de Pareto. Sendo assim, uma maneira de encontrar múltiplas soluções Pareto ótimas utilizando métodos clássicos, seria realizando várias resoluções do problema utilizando em cada resolução diferentes valores de pesos (DEB, 2004).

A segunda meta da otimização multiobjetivo é tão relevante quanto a primeira, pois obtendo-se um conjunto bem distribuído na fronteira de Pareto, obtêm-se uma maior alternativa de escolha da solução ótima, já que, é possível encontrar um conjunto de soluções Pareto ótimas concentradas e com poucas variações entre as soluções. Sendo assim, além de encontrar soluções não-dominadas na população, na otimização multiobjetivo, é necessário preservar a diversidade entre estas soluções. Até o momento, não foi apresentado nenhuma estratégia de preservar a diversidade entre as soluções em uma população, no entanto, na literatura existem diversas maneiras de fazer esta manutenção da diversidade. A estratégia utilizada para manter a diversificação da população é introduzir no algoritmo funções que avaliam a densidade de soluções por regiões do espaço de busca e, desta maneira, é possível dar preferência a soluções que preencham regiões menos densas (BARDANACHVILI, 2006). Em Deb (2004), por exemplo, podem ser encontrados alguns operadores de nicho que realizam esta manutenção da diversidade populacional. Sendo assim, é possível acoplar aos algoritmos evolucionários alguma estratégia de diversidade e, com isso, após algumas gerações do algoritmo, fazer com que as metas da otimização multiobjetivo sejam atendidas, ou seja, a população convirja próxima à fronteira de Pareto ótima e, além disso, possua uma boa distribuição das soluções.

Neste capítulo, apresentam-se alguns algoritmos evolucionários multiobjetivos, em particular, os que utilizam as idéias dos Algoritmos Genéticos. Sendo assim, descrevem-se algumas modificações que podem ser realizadas em um Algoritmo Genético básico a fim de encontrar múltiplas soluções Pareto ótimas. Especificamente, será apresentada em detalhes a técnica denominada NSGA-II (por seu acrônimo em inglês) que requer a incorporação de um Algoritmo Genético que melhore a qualidade das soluções não-dominadas durante o processo iterativo. A ênfase dada a esta técnica está relacionada a sua utilização como base principal para o desenvolvimento deste trabalho. No entanto, podem-se utilizar outros algoritmos evolucionários multiobjetivos para a obtenção das soluções Pareto ótimas.

5.2 Algoritmos Evolucionários Multiobjetivos

Na literatura especializada, encontram-se uma variedade de algoritmos evolucionários multiobjetivos. No entanto, analisar em detalhes todas as técnicas existentes é considerado uma tarefa inviável neste trabalho. Sendo assim, serão citados alguns algoritmos evolucionários multiobjetivos que podem ser encontrados em (DEB, 2004). Entre os principais algoritmos evolucionários multiobjetivos estão:

- **VEGA** (acrônimo do termo inglês Vector Evaluated Genetic Algorithm), implementado por Schaffer (SCHAFFER, 1984) foi o primeiro Algoritmo Genético Multiobjetivo para encontrar um conjunto de soluções não-dominadas;

- **MOGA** (acrônimo do termo inglês Multiple Objective Genetic Algorithm) foi introduzido por Fonseca e Fleming (FONSECA; FLEMING, 1993);

- **NPGA** (acrônimo do termo em inglês Niched-Pareto Genetic Algorithm) foi proposto por Horn e outros (HORN et al., 1994);

- **NSGA** (acrônimo do termo em inglês Non-Dominated Sorting Genetic Algorithm) foi implementado por Srinivas e Deb (SRINIVAS; DEB, 1994);

- **NSGA-II** (do inglês: Elitist Non-Dominated Sorting Genetic Algorithm) esta técnica foi sugerida por Deb e outros (DEB et al., 2000a e 2000b);

- Outros algoritmos;

De modo geral, os algoritmos apresentados se baseiam em adaptações para que se continue a utilizar as três operações básicas de um Algoritmo Genético, ou seja, a seleção, recombinação e mutação. Uma adaptação especial é feita na operação de seleção que depende da função de aptidão. No algoritmo VEGA, por exemplo, foi empregada a idéia de dividir a população de cada geração em M vetores, onde cada vetor se aplica a uma função objetivo nas comparações das aptidões. Outro exemplo de algoritmo que foi citado é o NSGA que trabalha dividindo a população em frentes não-dominadas e atribui um valor maior de aptidão às frentes de dominância de melhor qualidade. Além disso, as comparações são complementadas por funções que avaliam a densidade de soluções por regiões com o intuito de dar prioridade a preencher regiões menos densas. Entre os algoritmos citados, está o NSGA-II e, como foi comentado, este será apresentado em detalhes por ser o algoritmo base da implementação realizada neste trabalho. O NSGA-II é citado como uma técnica que obtêm um dos melhores desempenhos e, este operador, é considerado elitista e contém várias diferenças em relação ao NSGA comentado anteriormente. Dentre essas diferenças, o NSGA-II passa a avaliar a densidade das soluções no espaço objetivo através da Distância de Aglomeração (Crowding Distance), diferente do NSGA que avalia a densidade das soluções no espaço das variáveis (BARDANACHVILI, 2006). A seguir, dedica-se uma seção completa para a apresentação do NSGA-II.

5.3 Algoritmo Genético Elitista Baseado em um Ordenamento Não-Dominado (NSGA-II)

Como o próprio nome sugere, o NSGA-II é classificado como elitista e incorpora um mecanismo de preservação das soluções dominantes através de várias gerações do Algoritmo Genético. O processo se inicia a partir de uma população (P) com N indivíduos (Pais) obtidos aleatoriamente, ou também, através de uma heurística construtiva. As seguintes gerações são

obtidas usando mecanismos modificados de seleção, recombinação e mutação definidas pelo Algoritmo Genético básico.

Antes de apresentar todo o processo iterativo do NSGA-II é importante recordar que, no capítulo 3 foi apresentado o Método de Kung que é capaz de classificar uma população de indivíduos em frentes não-dominadas. Desta forma, é possível determinar a qualidade dos indivíduos da população baseando-se na frente a qual pertencem.

Outro conceito necessário para a apresentação do Algoritmo NSGA-II é a definição de um operador que permite estimar a densidade de soluções ao redor de uma solução particular na população. Este conceito é a Distância de Aglomeração (Crowding Distance) e será visto a seguir.

5.3.1 Distância de Aglomeração

Os algoritmos multiobjetivos buscam encontrar o maior número possível de soluções que pertençam à frente de Pareto. Portanto, é necessário que a população se mantenha a mais diversificada possível. A Distância de Aglomeração permite quantificar o espaço ao redor de uma solução i. Para isto, deve-se calcular o perímetro do hipercubo formado pelas soluções vizinhas a i que estão localizadas na mesma frente de dominância. Para fins de exemplificação, considere a figura 5.1 que mostra um problema de minimização com dois objetivos. Na figura, os círculos cheios representam soluções que pertencem a melhor frente e os círculos vazios são soluções que fazem parte de uma frente de qualidade inferior, para este exemplo, a Distância de Aglomeração da solução *i* é dada por:

$$d_i = \frac{f_1^{(i+1)} - f_1^{(i-1)}}{f_1^{\max} - f_1^{\min}} + \frac{f_2^{(i-1)} - f_2^{(i+1)}}{f_2^{\max} - f_2^{\min}} \tag{5.1}$$

Na expressão (5.1), $f_1^{\max}$, $f_1^{\min}$, $f_2^{\max}$ e $f_2^{\min}$ representam os valores máximos e mínimos sobre todo espaço solução para as funções objetivo f_1 e f_2. Esta expressão representa o semiperímetro do retângulo pontilhado da figura. Sendo assim, pode-se afirmar

que para um conjunto de soluções, a solução que introduz o maior nível de diversidade é aquela com maior distância de aglomeração.

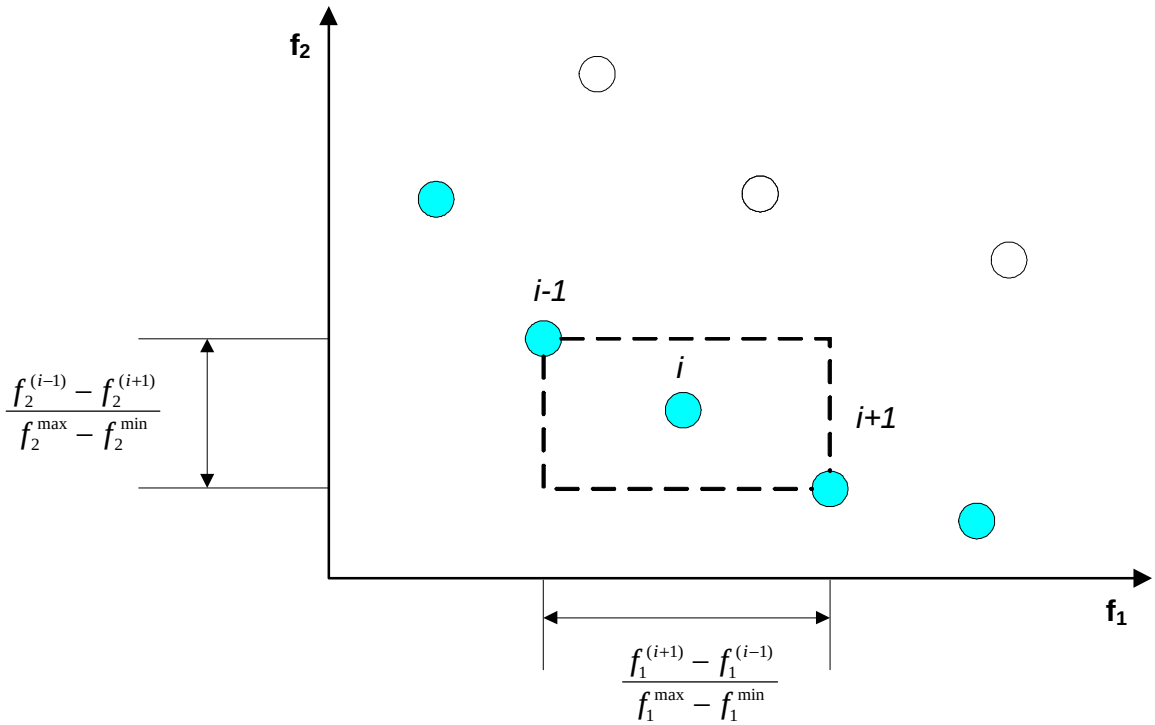

Figura 5.1: Distância de aglomeração para a solução *i*.

A seguir, é apresentado o algoritmo que calcula a Distância de Aglomeração para as soluções de uma determinada frente de dominância (DEB, 2004).

1. Seja n o número de soluções em uma determinada frente de dominância. Para cada solução i pertencente a esta frente de dominância, atribuir inicialmente uma distância $d_i = 0$;

2. Para cada função objetivo f_k com $k = 1,2,\mathbf{K},l$, ordenar o conjunto de soluções em ordem crescente de f_k, ou seja, encontrar o vetor de índices I^k.

3. Para $k = 1,2,\mathbf{K},l$ atribuir uma grande distância para as soluções extremas do vetor I^K, ou seja, $d_{I_1^K} = d_{I_n^K} = \infty$, e para todas as outras soluções $j = 2,3,\mathbf{K},n-1$ calcular:

$$d_{I_j^K} = d_{I_j^K} + \frac{f_K^{(I_{j+1}^K)} - f_K^{(I_{j-1}^K)}}{f_k^{\max} - f_k^{\min}} \tag{5.2}$$

O elemento I_j^K do vetor de índices representa a *j*-ésima solução no conjunto ordenado crescentemente em relação à f_k. Assim, I_1^K e I_n^K representam, respectivamente, o menor e maior valor para a função objetivo f_k. Como já citado, em (5.2) os parâmetros $f_k^{\max}$ e $f_k^{\min}$ representam os valores máximos e mínimos sobre todo o espaço de soluções para a *k*-ésima função objetivo.

Ordenando o conjunto de soluções em frentes de dominância e atribuindo uma Distância de Aglomeração para cada solução das frentes, pode-se definir a seguir o método de seleção usado no NSGA-II.

5.3.2 Seleção por Torneio Segundo a Distância de Aglomeração (< c)

Este procedimento equivale à seleção usada no Algoritmo Genético básico. Consiste em comparar duas soluções, das quais, cada uma, possui duas características:

1 – Um grau de não-dominância r_i (valor atribuído à solução baseado na frente de dominância a qual pertence, soluções da mesma frente possuem o mesmo grau);

2 – Uma distância de Aglomeração d_i (vista anteriormente).

Desta forma, defini-se a seleção (< c) como:

Definição: Uma solução *i* vence um torneio com outra solução *j* se qualquer uma das seguintes condições é verdadeira:

1 – Se a solução *i* tem um melhor grau, ou seja, $r_i < r_j$;

2 – Se têm o mesmo grau, mas a solução *i* tem uma melhor Distância de Aglomeração que a solução *j*, ou seja, $r_i = r_j$ e $d_i > d_j$.

Conhecendo-se a seleção (< c), podem-se determinar os descendentes finais para o NSGA-II. Isto é mostrado a seguir.

5.3.3 Determinação dos Descendentes Finais

Como visto anteriormente, o início do processo para a determinação dos descendentes finais acontece a partir de uma população (P) com N indivíduos (Pais) obtidos aleatoriamente, ou também, através de uma heurística construtiva. A próxima população (F) também com N indivíduos (Filhos) é criada a partir da população atual (P). Para criar a população (F), são selecionados aleatoriamente N pares de indivíduos da população atual (P) e cada par compete entre si através de um torneio e em cada torneio vence o indivíduo que faz parte da frente de dominância de melhor qualidade. Caso as alternativas que competem entre si pertençam à mesma frente, então ganha o indivíduo que introduza o maior grau de diversidade na população em construção (F). Com este procedimento, selecionam-se todos os N indivíduos, e os operadores de recombinação e mutação são realizados de maneiras idênticas aos realizados no Algoritmo Genético básico e, após a aplicação destes operadores, é gerada a população (F) com N indivíduos (Filhos). É importante mencionar que os vencedores de cada torneio são os únicos permitidos a obter descendência. Desta forma, espera-se que as informações genéticas das alternativas dominantes estejam presentes nas seguintes gerações e atraia o resto da população para suas vizinhanças (PEÑUELA; GRANADA, 2007).

Após a criação da população (F) usando a população (P), ao invés de encontrar as frentes não-dominadas de (F), primeiramente as populações (P) e (F) são unidas para formar uma nova população (R) agora com tamanho 2N. Então, a população inteira (R) é ordenada em frentes de dominância. Embora este procedimento exija um esforço maior se comparado com apenas ordenar a população (F), isto permite uma verificação global de não-dominância entre os indivíduos (Pais) e (Filhos). Assim que o ordenamento não-dominado dos 2N

indivíduos de (R) é realizado, esta nova população estará ocupada por soluções com diferentes frentes não-dominadas. A conformação da população final é feita com os indivíduos de (R) e inicia com a melhor frente não-dominada e continua com as soluções da segunda melhor frente não-dominada, em seguida com as soluções da terceira melhor frente não-dominada e assim por diante. Como o tamanho da população (R) é 2N, e a população final deve ser de tamanho N, nem todas as frentes de (R) poderão estar presentes na população final. Sendo assim, quando a última frente de (R) é considerada para conformar a população final, pode existir soluções nesta frente que ultrapassem o tamanho N da população final. Portanto, caso isto aconteça, é necessário eliminar indivíduos da última frente selecionada. Isto é feito eliminando os indivíduos com menor Distância de Aglomeração. Enfim, após todos esses passos os descendentes finais são gerados. A figura a seguir, esquematiza os procedimentos do NSGA-II.

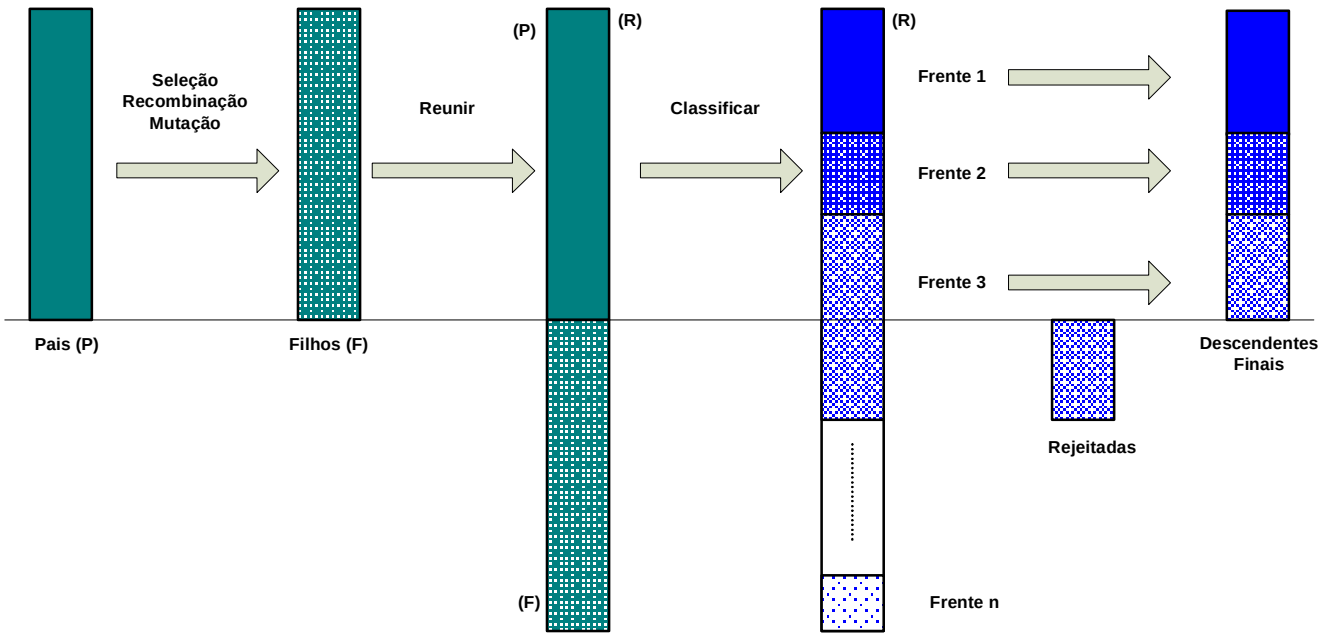

Figura 5.2: Esquemática dos procedimentos do NSGA-II.

A seguir, apresenta-se o pseudocódigo que sintetiza os procedimentos do algoritmo NSGA-II.

5.3.4 Pseudocódigo para o NSGA-II

1 – Gerar uma população P de tamanho N;

2 – Identificar as frentes de dominância e avaliar as Distâncias de Aglomeração em cada frente;

3 – Usando a Seleção (< c), recombinação e mutação, gera-se uma população de descendentes do mesmo tamanho de P;

4 – Reunir Pais e Filhos em um conjunto de tamanho 2N e classificá-los em frentes de dominância;

5 – Determinar os descendentes finais selecionando as frentes com melhor grau de dominância. Se o limite de tamanho N é superado, eliminar as soluções com menor Distância de Aglomeração na última frente selecionada;

6 – Se o critério de convergência é atingido, fim do processo. Caso contrário, retornar ao passo 3.

Até o momento, não foi comentado nada sobre os casos onde aparecem indivíduos infactíveis na população. Para estes casos, devem-se utilizar estratégias que possibilitem tratar de forma adequada os indivíduos com estas características. No capítulo a seguir, apresentam-se alguns métodos de manipulação de restrições usando Algoritmos Evolucionários. Em problemas de otimização restrita o aparecimento de indivíduos infactíveis é bastante comum, sendo assim, serão apresentadas algumas formas de lidar com estes tipos de indivíduos.

CAPÍTULO 6

ALGORITMOS EVOLUCIONÁRIOS PARA PROBLEMAS RESTRITOS

6.1 Introdução

Neste capítulo são apresentadas sucintamente algumas formas de trabalhar com problemas restritos usando Algoritmos Evolucionários, em especial, os Algoritmos Genéticos. Inicialmente, é feita uma introdução em que é mostrado como a presença de restrições dificulta consideravelmente a resolução de determinados problemas. É feita também uma breve análise das maneiras de lidar com problemas restritos, e desta forma, verifica-se algumas vantagens e desvantagens oriundas de cada uma destas estratégias.

Inicialmente, os Algoritmos Genéticos eram aplicados com sucesso em problemas de otimização sem a presença de restrições. No entanto, foi uma questão de tempo para surgirem as primeiras utilizações da técnica na resolução de problemas restritos e, não é para menos, pois a maioria dos problemas de otimização real envolvem restrições (CASTRO, 2001). Um determinado empreendimento, por exemplo, precisa satisfazer um conjunto de restrições relevantes para obtenção de sucesso na sua realização. Em problemas científicos também não é diferente, pois geralmente estes problemas ficam sujeitos a planos limítrofes, dependências

numéricas e outras restrições que devem ser obedecidas, caso contrário, a solução obtida é considerada infactível (LINDEN, 2006).

Apesar de ter sido apresentada no capítulo 2, apresenta-se novamente a formulação geral para um problema de otimização restrito (KOZIEL; MICHALEWICZ, 1999):

$$\text{Otimizar } f(\vec{x}) \qquad \vec{x} = (x_1, \mathbf{K}, x_n) \in R^n \tag{6.1}$$

sendo $\vec{x} \in F \subseteq S$. A função objetivo f está definida no espaço de busca $S \subseteq R^n$ e o conjunto $F \subseteq S$ define a região factível. Usualmente, o espaço de busca S está definido como um retângulo n-dimensional no R^n (domínio das variáveis definidas pelos seus limitantes inferior e superior):

$$l(i) \le x_i \le u(i), \quad 1 \le i \le n \tag{6.2}$$

enquanto que a região factível $F \subseteq S$ está definida pelo conjunto de m restrições adicionais $(m \ge 1)$:

$$g_j(\vec{x}) \le 0, \quad \text{para} \qquad j = 1, \mathbf{K}, q \tag{6.3}$$

e

$$h_j(\vec{x}) = 0, \quad \text{para} \qquad j = q+1, \mathbf{K}, m \tag{6.4}$$

As restrições de desigualdade que assumem valor igual à zero, ou seja, $g_i(\vec{x}) = 0$ no ótimo global do problema são chamadas de *restrições ativas*.

A presença de restrições em um problema de otimização aumenta consideravelmente a dificuldade na resolução deste problema. Estas dificuldades estão relacionadas com a quantidade de limites nas variáveis de decisão, os tipos de restrições envolvidas, as interferências entre as restrições e a inter-relação entre as restrições e a função objetivo (VENKATRAMAN; YEN, 2005).

Para se ter uma idéia das dificuldades envolvidas em problemas restritos, é feita a análise de um exemplo numérico. Considera-se o exemplo abaixo, semelhante ao apresentado em (VENKATRAMAN; YEN, 2005):

$$\text{Minimizar} \quad f(\vec{x}) = x_1 - x_2 \tag{6.5}$$

sendo que as duas variáveis de decisão estão definidas no intervalo [0,1], ou seja, $0 \le x_1 \le 1$ e $0 \le x_2 \le 1$. Sem a presença de mais nenhuma restrição adicional, o valor ótimo acontece quando $x_1 = 0$ e $x_2 = 1$, ou seja, $f(\vec{x}) = -1$. Agora, acrescenta-se ao problema uma restrição adicional $h(\vec{x})$ dada por $x_1 + x_2 = 0.5$. Neste caso, o novo valor ótimo será dado por $f(\vec{x}) = -0.5$ quando as variáveis de decisão assumem os valores $x_1 = 0$ e $x_2 = 0.5$. No entanto, analisando os problemas com e sem a restrição $h(\vec{x})$, pode-se verificar que a quantidade de soluções factíveis no problema que contém a restrição $h(\vec{x})$ diminui drasticamente. Este fato pode ser verificado nas figuras 6.1 e 6.2 que mostram as regiões factíveis dos problemas com e sem a restrição.

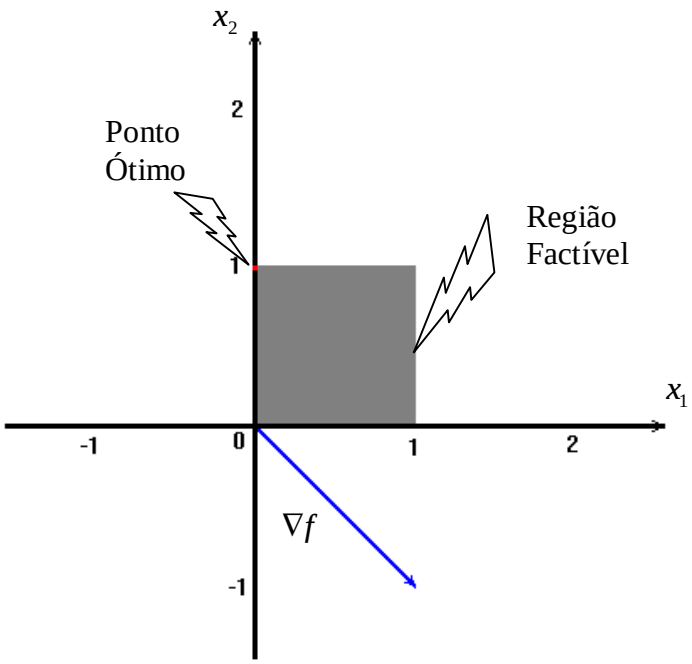

Figura 6.1: Representação gráfica do problema sem a presença da restrição $h(\vec{x})$

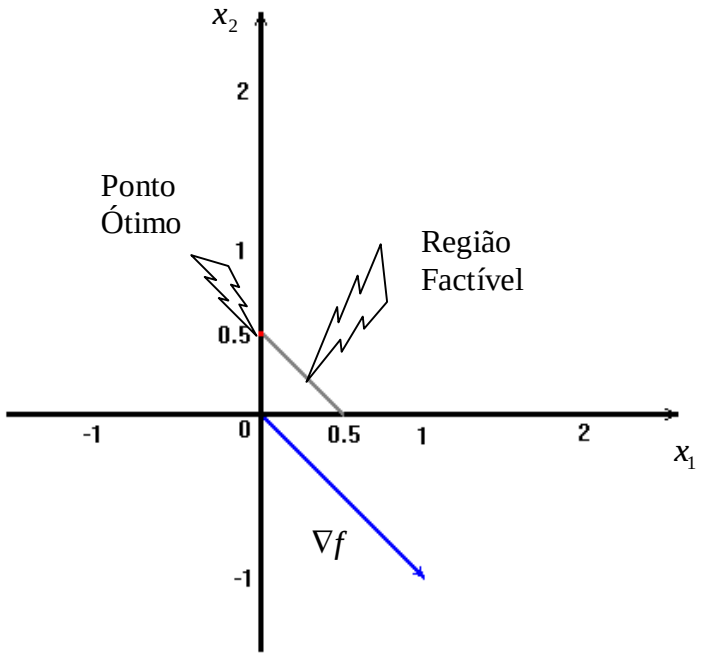

Figura 6.2: Representação gráfica do problema com a presença da restrição $h(\vec{x})$

Portanto, fica evidente que a dificuldade na solução de um problema pode ser significantemente aumentada quando o número de restrições cresce ou quando os tipos de restrições envolvidas são complexas.

Problemas altamente restritos apresentam grandes dificuldades na obtenção de sua solução. Em tais problemas, encontrar uma única solução factível pode ser considerado uma tarefa bastante trabalhosa. Diante disto, pode-se discutivelmente afirmar que a obtenção de uma solução factível prevalece a *priori* sobre otimizar a função objetivo. Problemas com estas características são tratados como problemas de satisfação de restrições e vários Algoritmos Evolucionários têm sido propostos para resolvê-los eficientemente. O desafio principal ao se trabalhar com problemas altamente restritos é realizar a manipulação das restrições e da função objetivo simultaneamente. Os métodos de manipulação de restrição têm direcionado a uma variedade de projetos de formulação da aptidão para cada indivíduo na população dependendo do valor de sua função objetivo e de sua satisfação de restrição (VENKATRAMAN; YEN, 2005). Várias técnicas de manipulação de restrições usando Algoritmos Evolucionários têm sido propostas para resolver problemas de otimização restrita e algumas das mais relevantes serão apresentadas a seguir.

6.2 Algoritmos Evolucionários para Resolver Problemas Restritos

Ao resolver problemas de otimização restritos, um dos primeiros procedimentos proposto é de simplesmente eliminar da população indivíduos que representam soluções infactíveis. Este procedimento pode não ser muito eficiente quando a região factível é muito pequena se comparada ao espaço de busca total. Neste caso, a população inicial pode ser totalmente infactível e, desta forma, inviabilizar a evolução do algoritmo. Outro fato agravante em utilizar tal procedimento acontece principalmente, quando se trabalha em espaços de busca não-convexos e, neste caso, depois de gerada a população inicial a aplicação dos operadores genéticos pode dar origem a uma grande quantidade de indivíduos infactíveis, exigindo desta forma um grande esforço computacional para a composição da nova população (LINDEN, 2006). Este fato é mostrado na figura a seguir.

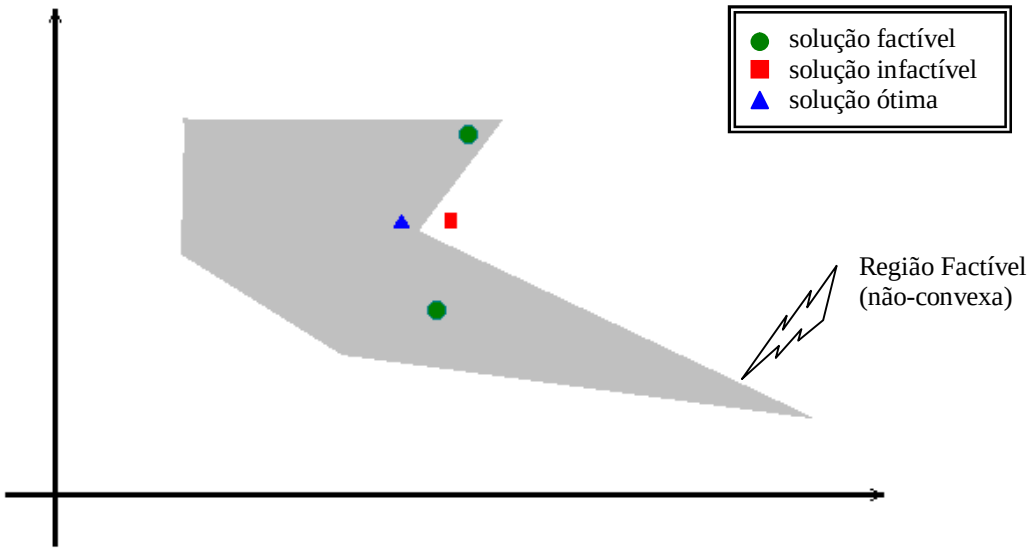

Figura 6.3: Geração de solução infactível após aplicação do operador genético

Na figura, representa-se a geração da solução infactível (quadrado) a partir das duas soluções factíveis (círculos). Pode-se observar que apesar da solução gerada ser infactível, ela está mais próxima da solução ótima (triângulo).

Por todos estes motivos, é importante entender que soluções infactíveis muitas vezes não devem ser excluídas de todo processo de busca. De uma forma geral, é desejável que

soluções factíveis tenham preferência sobre as infactíveis no processo de busca, no entanto, algumas soluções infactíveis estão muito próximas da região factível e possuem características interessantes em termos de avaliação. Caso tais soluções puderem participar da geração da nova população no processo de busca do Algoritmo Genético possivelmente serão encontradas mais rapidamente soluções melhores para o problema (LINDEN, 2006).

Nos últimos anos, os pesquisadores têm desenvolvidos uma grande quantidade de Algoritmos Evolucionários para resolver problemas de otimização restrita. Sendo assim, é possível categorizá-los em diferentes grupos (VENKATRAMAN; YEN, 2005):

- Métodos Baseados em Funções de Penalização;
- Métodos Baseados na Preferência de Soluções Factíveis sobre as Infactíveis;
- Métodos Baseados em Otimização Multiobjetivo.

6.2.1 Métodos Baseados em Funções de Penalização

Em uma grande parte das aplicações são empregados os métodos baseados na penalidade dos indivíduos infactíveis, procedimento também muito utilizado na otimização clássica e está entre os primeiros métodos usados para manipular restrições usando Algoritmos Evolucionários. Nestes métodos, os indivíduos são penalizados baseados em suas violações de restrições. A penalidade imposta nos indivíduos infactíveis pode variar desde a rejeição completa dos indivíduos infactíveis até o decréscimo de sua aptidão baseado no grau de violação das restrições destes indivíduos. Existem diferentes tipos de função penalidade baseados nestes princípios (VENKATRAMAN; YEN, 2005). A seguir, são discutidos sucintamente alguns dos principais tipos de métodos baseados na penalidade.

Primeiramente, apresenta-se o método de penalidade denominado Método da ”Pena de Morte”. Neste método, as soluções infactíveis não são consideradas para a geração da nova população, e esta é a maior penalidade que pode ser imposta a uma solução infactível. Do ponto de vista da complexidade de implementação, pode-se afirmar que o Método da “Pena de Morte” é bastante simples de ser implementado (VENKATRAMAN; YEN, 2005). No entanto, este método possui a grande desvantagem de não aproveitar nenhuma informação de indivíduos infactíveis para guiar a busca à solução ótima e, como já comentado, abolir

completamente a presença de soluções infatíveis no processo de busca pode ser uma opção não muito eficiente. Este método pode ser eficiente quando a região factível for convexa e ocupar uma grande parte do espaço de busca, caso contrário, não é aconselhável utilizá-lo.

Um procedimento comum no método das penalidades é associar um valor numérico referente a cada restrição não satisfeita penalizando o indivíduo infactível com este valor, desta forma, diminui-se proporcionalmente a aptidão destes indivíduos. Basicamente, este procedimento significa modificar a função objetivo da seguinte forma (VENKATRAMAN; YEN, 2005, LINDEN, 2006):

$$obj(\vec{x}) = f(\vec{x}) + \sum_{j=1}^{m} r_j c_j(\vec{x}) \tag{6.6}$$

sendo:

$f(\vec{x})$	valor atual da função objetivo;
r_j	coeficiente de penalidade para a restrição j;
$c_j(\vec{x})$	grau de violação da restrição j correspondente ao indivíduo $\vec{x}$;
$obj(\vec{x})$	função objetivo modificada após adicionar a penalidade;

Sendo assim, penalidade é uma soma dos pesos das violações das restrições. Neste trabalho, sem perda de generalidade, considera-se a minimização da função objetivo, a não ser que o contrário seja especificado. O procedimento de penalização definido pela expressão anterior, dá origem a diferentes métodos de penalidades, que são apenas particularidades do procedimento em questão. Tais métodos podem ser divididos em:

- Método de Penalidade Estática
- Método de Penalidade Dinâmica
- Método de Penalidade Adaptativo

No método de penalidade estática, como o próprio nome já diz, a penalização é definida a *priori* e não se altera com a execução do algoritmo.

O sucesso do método de penalidade estática depende da escolha dos coeficientes de penalidade para as restrições. Estes coeficientes devem ser determinados baseados nas dificuldades das restrições (VENKATRAMAN; YEN, 2005).

No método de penalidade dinâmica, define-se uma penalidade que evolui com o tempo de execução do algoritmo. No início do algoritmo usam-se penalidades menores, pois ainda necessita-se de soluções infactíveis de boas características genéticas para permitir o progresso da população. Conforme o processo evolui, as soluções tendem a melhorar e, consequentemente, aumenta-se gradativamente a penalidade, já que diminui a necessidade de soluções infactíveis para a convergência à solução ótima (LINDEN, 2006). No entanto, a dificuldade envolvida na calibração de muitos parâmetros para o método da penalidade dinâmica tem limitado significantemente sua aplicabilidade.

No método de penalidade adaptativo, em vez de ser usada uma penalização fixa, utiliza-se a população corrente como uma avaliação para a modificação da penalização. Uma maneira de definir esta modificação neste método consiste em diminuir a penalização quando a maioria dos indivíduos da população pertence à região factível e aumentar a penalização caso contrário (LINDEN, 2006).

Uma das desvantagens dos métodos apresentados está na exigência da utilização de parâmetros que devem ser calibrados para cada problema. Ao penalizar um indivíduo infactível diminui-se de alguma forma a aptidão deste indivíduo. Um dos problemas de simplesmente designar uma baixa aptidão arbitrária a indivíduos infactíveis, está no fato da não diferenciação da qualidade destes indivíduos, ou seja, existem indivíduos que estão próximos da região factível e alterando uma ou outra condição desrespeitada, representam soluções de boa qualidade. Portanto, deve-se tratar de forma diferenciada indivíduos infactíveis de acordo com sua qualidade relativa. Tratando relativamente os indivíduos infactíveis, pode-se garantir que ao realizar uma busca dentro da região infactível, o Algoritmo Genético através dos operadores genéticos encontre a região factível.

Podem-se resumir algumas características relacionadas aos métodos de penalidade como (RICHARDSON et al., 1989):

- penalidade que são funções da distância da região factível possuem melhor desempenho do que aquelas penalidades que são simplesmente funções do número de restrições violadas;

- para problemas que possuem poucas restrições e poucas soluções factíveis, as penalidades que são simplesmente funções do número de restrições violadas não fornece garantia de encontrar soluções factíveis;

- quanto mais precisa for a penalidade estimada, melhor pode ser a qualidade das soluções encontradas;

6.2.2 Métodos Baseados na Preferência de Soluções Factíveis sobre as Infactíveis

Pode-se afirmar que é importante a existência de uma ordenação básica para assegurar que o pior indivíduo que respeita todas as restrições tem melhor aptidão do que o melhor indivíduo infactível. Afinal, qualquer solução factível deve ter preferência sobre aquela solução que é infactível. No entanto, isto pode ser violado definindo uma região bem próxima da região factível, ou seja, pode-se definir uma tolerância d de violação de restrição aceitável e, desta forma, se um indivíduo for "levemente" infactível respeitando esta tolerância este indivíduo pode ser considerado como factível. Isto é bastante interessante, pois desta forma utiliza-se informações de soluções que são quase factíveis e que podem apresentar características interessantes para a evolução do algoritmo (LINDEN, 2006).

Baseado no princípio que soluções factíveis devem ter melhor aptidão que um indivíduo infactível, surge os métodos baseados na preferência de soluções factíveis sobre as infactíveis. Nestes métodos, as soluções factíveis dominam as infactíveis e, desta maneira, soluções que respeitam todas as restrições dos problemas terão valores de aptidão superiores aos indivíduos que não respeitam. Uma maneira de realizar tal procedimento foi proposta em (POWELL; SKOLNICK, 1993), onde as soluções factíveis foram avaliadas no intervalo $(-\infty, 1)$ baseadas nos valores da função objetivo, e as soluções infactíveis foram avaliadas no intervalo $(1, \infty)$ baseadas nas violações de restrições e, com isso, as soluções eram ordenadas

considerando estes valores dos intervalos. Consequentemente, neste método, todas as soluções factíveis dominam as infactíveis (obviamente, considera-se um problema de minimização). Soluções infactíveis são comparadas baseadas em suas violações de restrições, enquanto soluções factíveis são comparadas baseadas em seus valores de função objetivo. Analisando este método podem-se observar as seguintes características (VENKATRAMAN; YEN, 2005):

- contanto que nenhuma solução factível é encontrada, a função objetivo não interfere no ordenamento dos indivíduos;

- existindo a combinação de soluções factíveis e infactíveis na população, as soluções factíveis serão ordenadas à frente das soluções infactíveis;

- soluções factíveis serão ordenadas baseadas em seus valores de função objetivo;

Em relação ao método apresentado, pode-se afirmar que a sua maior desvantagem está relacionada com a perda da diversidade que fica definido explicitamente como parte da seleção dos indivíduos. Esta desvantagem fica evidente em problemas multimodais e, neste caso, o Algoritmo Genético pode ficar restrito a uma região do espaço de busca, ou seja, a perda da diversidade faz com que o algoritmo esteja suscetível a ficar preso a ótimos locais (VENKATRAMAN; YEN, 2005).

6.2.3 Métodos Baseados em Otimização Multiobjetivo

Os métodos baseados em otimização multiobjetivo, atualmente são de grande interesse científico e formam o estado de arte dos algoritmos de otimização restrita. Uma maneira de abordar um problema mono-objetivo restrito através de uma filosofia multiobjetivo é transformar o problema mono-objetivo em biobjetivo, ou seja, considerar como objetivos, simultaneamente, a função objetivo e o grau de violação de restrições dos indivíduos na população. A figura a seguir representa esta idéia considerando a minimização simultânea da função objetivo e das violações de restrições:

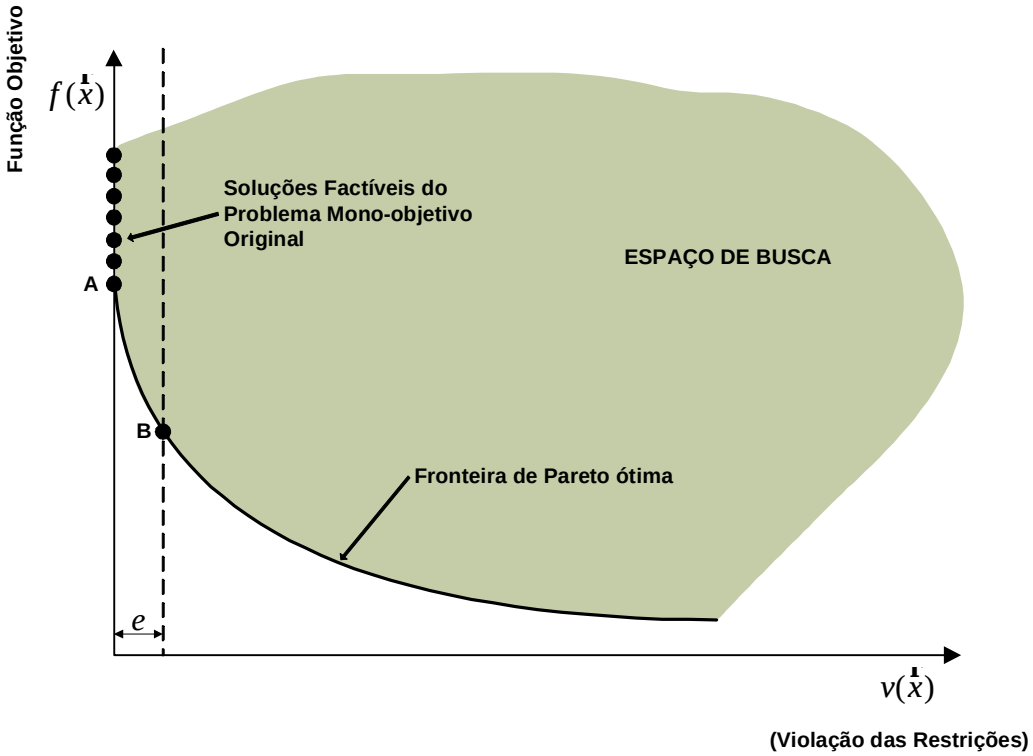

Figura 6.4: Problema mono-objetivo restrito abordado como multiobjetivo.

Na figura 6.4, como já dito, mostra-se um problema mono-objetivo restrito abordado através de uma filosofia multiobjetivo onde $v(\vec{x})$ representa o grau de violação de restrições e $f(\vec{x})$ é a função objetivo original. Também é mostrada a frente de Pareto ótima do problema multiobjetivo, o ponto A que representa a melhor solução factível do problema restrito (mínimo restrito) e o ponto B que representa o valor mínimo considerando uma pequena violação das restrições e aceitando uma margem de violação e .

É importante frisar que nos problemas do mundo real é pouco comum encontrar restrições rígidas (que devem ser obedecidas rigorosamente), ou seja, permite-se uma margem de violação suave nas restrições e, consequentemente, obtêm-se uma melhoria importante no valor da função objetivo. Portanto, o uso da otimização multiobjetivo através de algoritmos evolucionários permite obter um conjunto com os *melhores compromissos* entre as funções objetivos (pareto ótimo) envolvidas no problema. Sendo assim, todas as soluções pertencentes à fronteira de Pareto ótima que estão entre os pontos A e B são de grande interesse.

Abordar um problema mono-objetivo restrito com a filosofia multiobjetivo apresentada, faz com que seja analisado um aspecto interessante. Na figura 6.4 está representada a fronteira de Pareto completa, no entanto, apenas uma parte desta região é de maior interesse (região entre os pontos A e B). Portanto, no processo de busca seria interessante intensificar a busca nesta região. Direcionar a busca para uma região particular do espaço de busca, possui vantagens, tais como: possibilidade de diminuição do esforço de busca e obtenção de uma melhor precisão nas soluções não-dominadas. Como a busca é direcionada para uma região particular, fica clara a redução no esforço de busca. Além disso, como uma menor região é buscada, espera-se que a densidade das soluções obtidas seja maior e, desta forma, aumenta-se a precisão das soluções obtidas (DEB, 2004). Existem estratégias como o *princípio de dominância por pesos*, *método de dominância guiada*, entre outras que podem ser usadas durante o processo de otimização para obter um conjunto de soluções na região Pareto ótima de interesse. O método de dominância guiada, por exemplo, introduz uma transformação dos objetivos através de uma soma ponderada e, desta maneira, o conceito de dominância é modificado possibilitando que diferentes processos enfoquem diferentes regiões de busca. Neste método, a função ponderada de objetivos é definida como (DEB, 2004) apud (BRANKE et al., 2000):

$$\Omega_i(f(\mathbf{r}x)) = f_i(\mathbf{r}x) + \sum_{j=1, j\neq i}^{l} a_{ij} f_j(\mathbf{r}x), \quad \text{para} \quad i = 1,2,\mathbf{K},l \tag{6.7}$$

sendo a_{ij} o valor do ganho na *j*-ésima função objetivo para a perda de uma unidade na *i*-ésima função objetivo. A expressão (6.7) exige fixar uma matriz **a** contendo elementos 1 em sua diagonal principal. Desta forma, o conceito de dominância é definido considerando as funções ponderadas Ω_i, ou seja, para um problema de minimização, uma solução $\mathbf{r}x(1)$ domina outra solução $\mathbf{r}x(2)$ se $\Omega_i(\mathbf{f}(\mathbf{r}x(1)) \pounds \Omega_i(\mathbf{f}(\mathbf{r}x(2))$ e esta desigualdade é estritamente satisfeita para pelo menos um objetivo.

A seguir, ilustra-se o conceito do método através de um exemplo que pode ser encontrado em (DEB, 2004; DEB et al., 2002a). Neste exemplo, considera-se o método para duas funções objetivos $(l = 2)$ e, deste modo, obtêm-se as seguintes funções ponderadas:

$$\Omega_1(f_1, f_2) = f_1 + a_{12} f_2 \tag{6.8}$$

$$\Omega_2(f_1, f_2) = f_2 + a_{21} f_1 \tag{6.9}$$

As equações acima poderiam ser expressas por:

$$\boldsymbol{\Omega} = \begin{bmatrix} 1 & a_{12} \\ a_{21} & 1 \end{bmatrix} \mathbf{f} \qquad \text{ou} \qquad \boldsymbol{\Omega} = \mathbf{af} \tag{6.10}$$

É interessante notar que as funções ponderadas (6.8) e (6.9) podem ser reescritas, respectivamente, como:

$$f_2 = -\frac{1}{a_{12}} f_1 + \frac{\Omega_1(f_1, f_2)}{a_{12}} \tag{6.11}$$

$$f_2 = -a_{21} f_1 + \Omega_2(f_1, f_2) \tag{6.12}$$

Na figura a seguir, são representadas as linhas de contorno correspondente às duas funções lineares acima passando pela solução A no espaço objetivo, sendo $-\frac{1}{a_{12}}$ e $-a_{21}$ os coeficientes angulares dessas retas. Todas as soluções pertencentes à região em destaque são dominadas por A de acordo com a definição de dominância dada anteriormente.

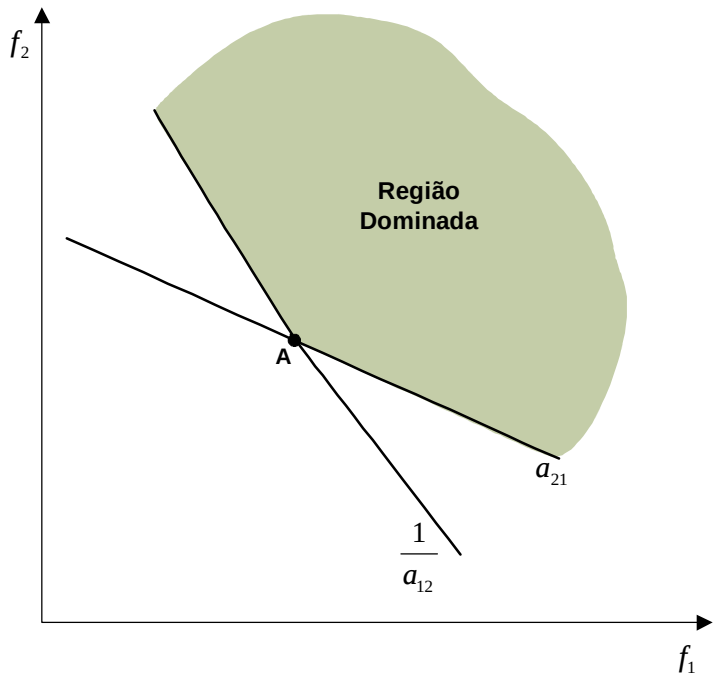

Figura 6.5: Região dominada pela solução A usando a definição de dominância modificada.

Por outro lado, considerando a definição de dominância usual, ou seja, uma solução $\vec{x}(1)$ domina outra solução $\vec{x}(2)$ se $f(\vec{x}(1)) \leq f(\vec{x}(2))$ e esta desigualdade é estritamente satisfeita para pelo menos um objetivo; obtém-se a nova região dominada pela solução A representada na figura a seguir. É interessante observar que a definição de dominância usual, é a mesma dada para a dominância modificada, no entanto, para a dominância usual, considera-se na expressão (6.10) a matriz **a** como sendo a identidade.

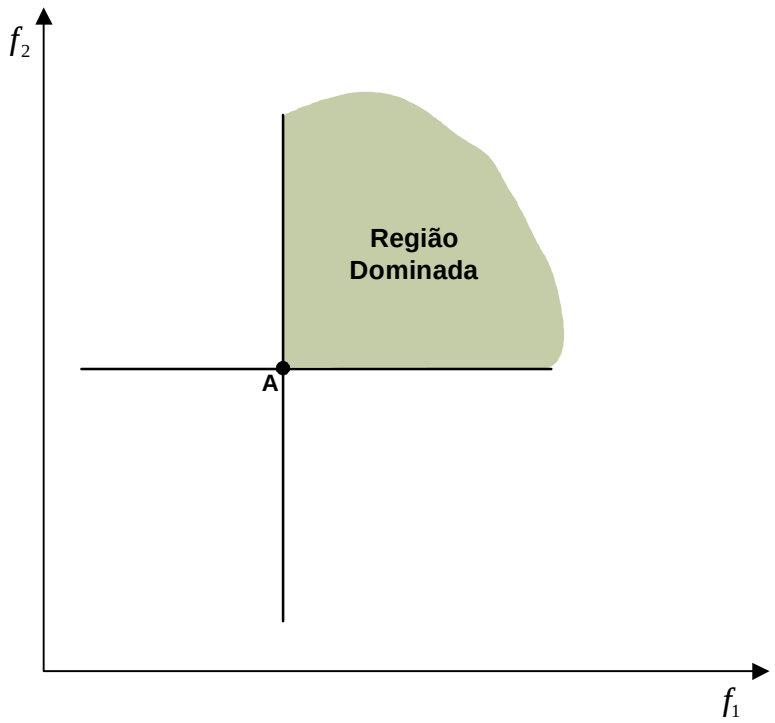

Figura 6.6: Região dominada pela solução A usando a definição de dominância usual.

Portanto, fazendo-se um comparativo entre as regiões mostradas nas figuras 6.5 e 6.6, está claro que a definição modificada de dominância permite tornar dominada uma região maior do que qualquer região representada pela definição usual de dominância. Assim, utilizando uma região maior proporcionada pela definição de dominância modificada, a fronteira de Pareto ótima completa (dada pela definição de dominância usual), pode se tornar dominada ao invés de não-dominada. Para mostrar esse fato, considera-se a figura a seguir que mostra a fronteira de Pareto ótima juntamente com os mesmos valores de a_{12} e a_{21} utilizados no exemplo da figura 6.5.

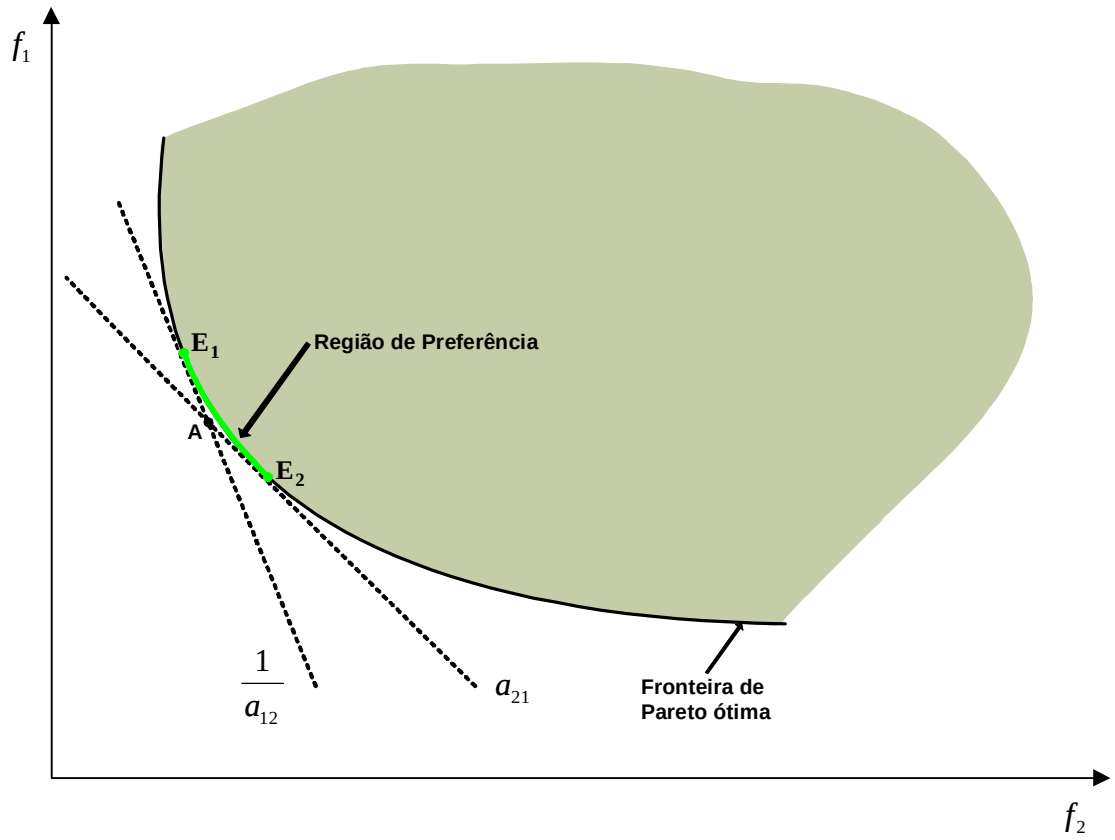

Figura 6.7: Parte não-dominada da fronteira de Pareto ótima.

Nesta figura, percebe-se que apenas as soluções próximas ao ponto A da fronteira de Pareto ótima original, agora são não-dominadas. Assim, fica claro que parcelas da fronteira de Pareto ótima original é dominada (região fora da curva $E_1 - E_2$) pelas soluções pertencentes à curva $E_1 - E_2$ em destaque. Portanto, espera-se que um algoritmo evolucionário multiobjetivo encontre somente pontos da região de preferência na fronteira de Pareto (pontos da curva $E_1 - E_2$), ou seja, este método direciona a busca para uma região particular da fronteira de Pareto ótima original. Desta forma, escolhendo valores apropriados para os elementos da matriz **a**, a região de interesse entre os pontos A e B da figura 6.4 pode ser priorizada.

Como já se afirmou, abordar um problema mono-objetivo restrito através de uma filosofia multiobjetivo é bastante interessante e possui uma série de vantagens. Entre outras características que fazem com que a utilização dos métodos de otimização multiobjetivo usando algoritmos evolucionários sejam mais vantajosos em relação a outros métodos usados para resolver problemas de otimização restrita são:

- As restrições que dominam o problema podem ser resolvidas de uma maneira natural, ou seja, não é necessário formular funções objetivos penalizadas artificialmente.

- Pelo fato anterior, não necessita de parâmetros de penalização. Estes parâmetros introduzem uma componente subjetiva na solução do problema.

Neste trabalho, utiliza-se esta mesma abordagem multiobjetivo para desenvolver o algoritmo proposto que será apresentado no capítulo a seguir.

CAPÍTULO 7

ALGORITMO PROPOSTO

7.1 Introdução

Neste capítulo é apresentado o algoritmo proposto, ou seja, um Algoritmo Genético que é especializado em resolver problemas altamente restritos e com variáveis contínuas. No capítulo anterior, foram apresentadas algumas estratégias utilizadas para resolver problemas de otimização restrita usando algoritmos evolucionários e, desta forma, foram mostradas as vantagens e desvantagens da utilização de cada uma destas estratégias. De modo geral, as aplicações destas estratégias apresentam algumas particularidades como:

- falta de elitismo;
- necessidade de escolha de parâmetros que exigem conhecimentos prévios sobre as características do problema;
- a não garantia de obtenção de soluções factíveis.

A análise das particularidades apresentadas serviu de motivação para a elaboração do algoritmo proposto. O algoritmo proposto, tentar evitar algumas das desvantagens decorrentes da aplicação de alguns métodos para resolver problemas restritos.

No algoritmo proposto é incorporado o elitismo e, além disso, não é necessário utilizar parâmetros dependentes do problema ou qualquer outra componente subjetiva para resolver os problemas. Sendo assim, buscou-se elaborar um algoritmo bastante genérico, mesmo o Teorema da Inexistência do Almoço Grátis (No Free-Lunch Theorem) afirmando que nenhum Algoritmo Genético será eficiente na resolução de todos os tipos de problemas (WOLPERT; MACREADY, 1997).

Finalmente, buscou-se elaborar um algoritmo que garanta o fornecimento de soluções factíveis através de uma busca exclusiva por factibilidade no início do processo. Tal procedimento é bastante interessante para resolução de problemas altamente restritos. Na seção a seguir, apresenta-se detalhadamente este algoritmo.

7.2 Algoritmo Proposto

Entre os métodos apresentados para resolver problemas com restrições, estão os métodos baseados na otimização multiobjetivo e, como já comentado, estes métodos são de grande interesse. Sendo assim, no seu desenvolvimento, o algoritmo proposto utiliza a estratégia multiobjetivo apresentada no capítulo anterior, ou seja, um problema mono-objetivo de otimização restrita é transformado em um problema biobjetivo. Portanto, o problema modificado terá dois objetivos, onde um é a função objetivo original $f(\vec{x})$ e o outro é o grau de violação das restrições $v(\vec{x})$. Assim, um objetivo considera a otimalidade e o outro a factibilidade.

O algoritmo proposto é composto de duas fases. Na primeira fase, a função objetivo original é completamente desconsiderada e o problema de otimização concentra-se somente em minimizar o grau de violação das restrições $v(\vec{x})$. Assim, o algoritmo encontra, eventualmente, uma solução factível, pois a busca é dirigida baseada somente na violação das restrições. Além disso, nesta fase utiliza-se a idéia da filosofia da programação por metas lexicográfico apresentada na seção 3.6.3 do capítulo 3 como estratégia para minimização do

grau de violação das restrições. A segunda fase consiste em otimizar, simultaneamente, a função objetivo original e o grau de violação das restrições usando um algoritmo multiobjetivo.

7.2.1 Fase I: Algoritmo de Satisfação das Restrições

Nesta fase, a função objetivo original é completamente desconsiderada e todo esforço do algoritmo está direcionado em encontrar uma única solução factível. A cada indivíduo da população é atribuído um valor de aptidão de acordo com o seu grau de violação das restrições. Posteriormente, utiliza-se uma estratégia elitista para assegurar que a solução com menor violação de restrições seja copiada para a geração seguinte. Esta fase permite obter uma solução que satisfaça todas as restrições, e representa uma solução útil no mundo real.

Na execução da primeira fase, utiliza-se um escalar de violação das restrições para representar o grau de desrespeito das restrições para cada indivíduo da população e, com isso, atribuir seu valor de aptidão. A seguir, mostra-se o procedimento adotado para a determinação deste escalar.

- Escalar de Violação de Restrição

Na formulação geral de um problema de otimização restrito, estão envolvidas *m* restrições de desigualdade e igualdade. Sendo assim, afirma-se que a violação de restrição de um indivíduo é um vetor *m* dimensional, onde cada coordenada deste vetor representa a violação de determinado indivíduo da população a cada uma das restrições. Portanto, defini-se a violação do indivíduo $\vec{x}$ na *j*-ésima restrição como (VENKATRAMAN; YEN, 2005):

$$c_j(\vec{x}) = \begin{cases} \max\left(0, g_j(\vec{x})\right) & \text{para} \quad j = 1, \mathbf{K}, q \\ \max\left(0, \left|h_j(\vec{x})\right| - d\right) & \text{para} \quad j = q+1, \mathbf{K}, m \end{cases} \tag{7.1}$$

sendo que $\left|\ \right|$ denota o operador de valor absoluto.

Na expressão (7.1) observa-se que as restrições de igualdade (rígidas) foram convertidas em restrições suaves utilizando uma tolerância d. Desta forma, facilita-se a obtenção de soluções que satisfaça as restrições de igualdade $h_j(\vec{x})$.

Para dar a todas as restrições o mesmo grau de importância, cada violação de restrição individual $c_j(\vec{x})$ calculada em (7.1) são normalizadas. Isto é feito dividindo cada uma delas pela maior valor de violação da restrição correspondente de toda a população. Primeiramente, encontra-se a violação máxima de cada restrição na população usando a seguinte expressão:

$$c_{\max}(j) = \max_{X} c_j(\vec{x}) \tag{7.2}$$

Estes valores de máxima violação de cada restrição em toda a população são utilizados para normalizar cada violação calculada em (7.1). Finalmente, as violações de restrições normalizadas de cada indivíduo da população são somadas e divididas pela quantidade de restrições *m*, produzindo um escalar de violação para cada um destes indivíduos. Obviamente, estes escalares assumirão valores entre 0 e 1. Cada escalar é representado por $v(\vec{x})$ e calculado pela seguinte expressão:

$$v(\vec{x}) = \frac{\sum_{j=1}^{m} \frac{c_j(\vec{x})}{c_{\max}(j)}}{m} \tag{7.3}$$

Através da expressão (7.3) calcula-se um único parâmetro que representa o grau de violação normalizada que abrange todas as restrições para cada indivíduo da população. Este escalar é denominado *escalar de violação de restrição.*

No próximo capítulo, serão apresentados, detalhadamente, 11 problemas testes de otimização restrita que foram utilizados na verificação da qualidade do Algoritmo Proposto. No entanto, a seguir antecipa-se um destes problemas para exemplificar como é feito o cálculo dos escalares de violação de restrição de uma população de indivíduos através dos procedimentos vistos anteriormente. Isto permite obter a função de adaptação da fase um.

Problema G5:

$$\text{Minimizar } G5(\vec{x}) = 3x_1 + 0{,}000001x_1^3 + 2x_2 + \frac{0{,}000002}{3}x_2^3$$

s.a

$r_1 \equiv x_3 - x_4 - 0{,}55 \le 0$

$r_2 \equiv x_4 - x_3 - 0{,}55 \le 0$

$r_3 \equiv 1000\,\text{sen}(-x_3 - 0{,}25) + 1000\,\text{sen}(-x_4 - 0{,}25) + 894{,}8 - x_1 = 0$

$r_4 \equiv 1000\,\text{sen}(x_3 - 0{,}25) + 1000\,\text{sen}(x_3 - x_4 - 0{,}25) + 894{,}8 - x_2 = 0$

$r_5 \equiv 1000\,\text{sen}(x_4 - 0{,}25) + 1000\,\text{sen}(x_4 - x_3 - 0{,}25) + 1294{,}8 = 0$

$r_6 \equiv 0 \le x_i \le 1200 \qquad \text{para } i = 1, 2$

$r_7 \equiv -0{,}55 \le x_i \le 0{,}55 \qquad \text{para } i = 3, 4$

sendo $r_1, r_2, \mathbf{K}, r_7$ as restrições do problema.

Através de uma população gerada aleatoriamente com 5 indivíduos, será calculado seus escalares de violação de restrição para o problema G5. A tabela 7.1 mostrada a seguir representa estes indivíduos e os respectivos valores das variáveis de decisão. No processo de geração da população, deve-se ter um cuidado especial para garantir a satisfação das restrições r_6 e r_7 de forma que o problema fique limitado somente pelas 5 primeiras restrições $(r_1, r_2, \mathbf{K}, r_5)$.

Tabela 7.1: População de indivíduos gerada aleatoriamente.

Indivíduos $\vec{x}$ da População X	Variáveis de Decisão x_n			
	x_1	x_2	x_3	x_4
$\vec{x}(1)$	1156,4	133,69	0,14667	0,3424
$\vec{x}(2)$	867,67	791,77	-0,33318	0,08332
$\vec{x}(3)$	955,76	982,42	0,36467	0,01741
$\vec{x}(4)$	722,69	153,13	0,37178	0,055625
$\vec{x}(5)$	747,24	966,49	0,12927	-0,4855

Avaliando cada indivíduo da população acima para o problema G5 e desconsiderando completamente a função objetivo, obtêm-se as violações exatas por indivíduo para cada uma das 5 restrições. A tabela 7.2 que segue mostra esses dados.

Tabela 7.2: Valores das restrições para cada indivíduo.

Indivíduos $\vec{x}$ da População X	Restrições $r_j(\vec{x})$				
	r_1	r_2	r_3	r_4	r_5
$\vec{x}(1)$	-0,74573	-0,35427	-1206,30	226,85	1332,80
$\vec{x}(2)$	-0,9665	-0,1335	-216,97	-1065,90	1294,60
$\vec{x}(3)$	-0,20274	-0,89726	-901,88	123,90	501,93
$\vec{x}(4)$	-0,23384	-0,86616	-711,27	929,26	565,26
$\vec{x}(5)$	0,064772	-11,648	10,647	164,62	-137,11

Aplicando a expressão (7.1) aos dados anteriores e assumindo uma tolerância $d = 0{,}01$ para as restrições de igualdade, obtêm-se os dados apresentados na tabela 7.3 a seguir. Desta forma, é trivial obter os valores $c_{\max}(j)$ referentes à expressão (7.2).

Tabela 7.3: Violações considerando $d = 0{,}01$ para as restrições de igualdade.

Indivíduos $\vec{x}$ da População X	Violação $c_j(\vec{x})$				
	c_1	c_2	c_3	c_4	c_5
$\vec{x}(1)$	0	0	1206,29	226,84	1332,79
$\vec{x}(2)$	0	0	216,96	1065,89	1294,59
$\vec{x}(3)$	0	0	901,87	123,89	501,92
$\vec{x}(4)$	0	0	711,26	929,25	565,25
$\vec{x}(5)$	0,064772	0	10,646	164,61	137,10
$c_{\max}(j)$	0,064772	-	1206,29	1065,89	1332,79
$i_{\max}(j)$	5	-	1	2	1

É conveniente gerar um vetor adicional $i_{\max}(j)$ que contenha os indivíduos geradores de cada $c_{\max}(j)$ atual. Assim, por exemplo, a máxima violação da restrição 1 é ocasionada pelo indivíduo 5, ou seja, $x(5)$.

Finalmente, aplicando a expressão (7.3) aos dados da tabela anterior, obtêm-se um vetor de escalares de violação de restrição $v(X)$, onde cada coordenada deste vetor quantifica o grau de infactibilidade de cada indivíduo da população *X*. Este vetor é representado por:

$$v(X) = [0{,}44256;\ 0{,}43024;\ 0{,}2481;\ 0{,}37711;\ 0{,}25322]^T$$

O vetor $v(X)$ corresponde à função de adaptação da fase um que será usada no processo de seleção. Para a busca de soluções factíveis, utiliza-se um Algoritmo Genético tradicional com codificação real incorporando a idéia do método lexicográfico da programação por metas. No algoritmo proposto a incorporação do método lexicográfico na fase um é chamada de Diminuição Progressiva da Tolerância de Aceitação das Restrições Complexas.

- Diminuição Progressiva da Tolerância de Aceitação das Restrições Complexas (DPTARC) na Fase I

Na figura 6.4 do capítulo anterior, mostrou-se a manipulação das restrições através da aceitação de uma pequena margem de violação e. A técnica usada para encontrar as soluções factíveis consiste em considerar um intervalo para a margem de violação ($e_{\min} \le e \le e_{\max}$) de maneira que o Algoritmo Genético têm como objetivo inicial minimizar $v(X)$ calculado na expressão (7.3), considerando $e_{\max}$. Assim, inicialmente o algoritmo trabalha com uma alta margem de violação e, desta forma, espera-se que o Algoritmo Genético alcance seu objetivo com um baixo custo computacional. Posteriormente, a margem de violação é reduzida cada vez que o Algoritmo Genético alcança um objetivo parcial até finalmente alcançar uma margem de violação de restrições menor ou igual a $e_{\min}$. Neste ponto, o Algoritmo Genético terá encontrado uma solução factível.

Como estratégia de redução da margem de violação propõe-se utilizar $e = (1-t)e$. O valor de $e_{\min}$ corresponde à tolerância d usada para avaliar o cumprimento das restrições de igualdade como, por exemplo, $e_{\min} = d = 0{,}0001$. O valor de $e_{\max}$ é um valor "atrativo" que permite o Algoritmo Genético encontrar facilmente uma população com uma razoável margem de infactibilidade (região factível inicial). A partir desta população, o processo de otimização guia a busca para a seguinte região factível de melhor qualidade até encontrar uma solução factível com o grau de precisão desejado. A figura a seguir, mostra o processo de busca por soluções factíveis para o problema G5 partindo de uma população aleatória e considerando $e_{\max} = 0{,}4$, $e_{\min} = 0{,}0001$ e $t = 0{,}5$. Os círculos brancos correspondem à população inicial, os asteriscos indicam a evolução da população depois de várias gerações e o momento que alcança cada uma das violações e.

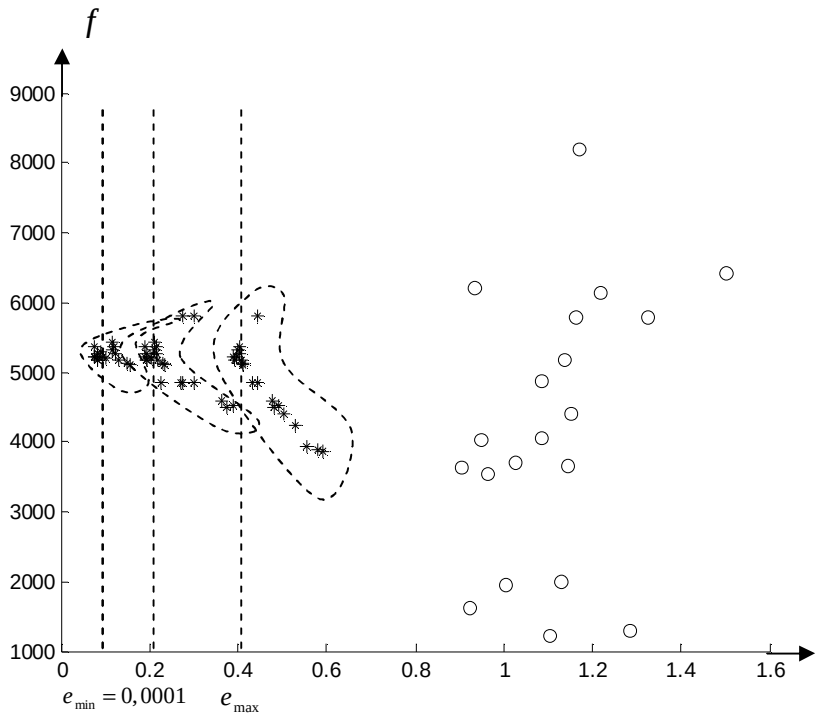

Figura 7.1: Processo de busca por soluções factíveis. Evolução para diferentes valores de e.

Para realizar uma análise desta fase, definem-se alguns termos (VENKATRAMAN; YEN, 2005):

- F região factível, ou seja, o domínio do espaço de busca *S* que, obviamente, é factível;
- C_i *i*-ésima componente factível desconectada no espaço de busca *S,* sendo $i = 1, \mathbf{K}, r$;

- $C_i \cap C_j = f$ para $i \neq j$ e $i, j = 1, \mathbf{K}, r$

- $C_1 \cup C_2 \cup \mathbf{K}\, C_r = F$

Baseando-se nos termos apresentados, discutem-se algumas particularidades da primeira fase deste algoritmo. Como já comentado, o objetivo desta fase é de encontrar pelo menos uma solução factível. O alcance desta solução factível deve ser obtido após uma inicialização randômica de indivíduos que compõem a população inicial.

Como existem *m* restrições e *r* componentes factíveis, o Algoritmo Genético proposto cuja seleção dos indivíduos é baseada exclusivamente na violação de restrições, encontrará uma solução factível com uma probabilidade $t \to \infty$. Esta probabilidade é verdadeira, pois neste caso, o *escalar de violação de restrição* representa somente uma medida da distância da região factível e, como a seleção dos indivíduos favorece a minimização desta distância, pode-se afirmar que, eventualmente, uma solução factível será encontrada. Como existem *r* componentes factíveis, a probabilidade de a primeira solução factível ser encontrada em qualquer uma das componentes factíveis é de aproximadamente $\frac{1}{r}$.

A seguir, apresenta-se a segunda fase do algoritmo proposto. Nesta fase, a verdadeira otimização é realizada, ou seja, busca-se de fato o ótimo global do problema.

7.2.2 Fase II: Algoritmo de Otimização com Restrições

Esta fase entra em execução quando pelo menos uma solução factível é encontrada na fase um. Na fase um, a função de adaptação correspondia ao grau de violação das restrições. Nesta fase, a evolução dos indivíduos acontece com base na qualidade do conjunto não-dominado ao qual pertencem. Na segunda fase a violação das restrições $v(\vec{x})$ e a função objetivo original $f(\vec{x})$ devem ser minimizadas simultaneamente dentro de um espaço de busca modificado (f - v) como foi mostrado na figura 6.4 do capítulo anterior. O indivíduo factível que possui o melhor valor para a função objetivo $f(\vec{x})$ será a incumbente atual do espaço de busca, ou seja, armazena-se a melhor solução factível do problema.

Nesta fase, o algoritmo proposto utiliza o Algoritmo Genético elitista baseado em um ordenamento não-dominado (NSGA-II). Este algoritmo utiliza, para manter a diversidade dos indivíduos pertencentes ao conjunto de soluções não-dominadas um esquema de nicho baseado na utilização da distância Euclidiana normalizada entre os indivíduos no espaço objetivo (f - v) chamado de *distância de aglomeração*. A metodologia completa do NSGA-II foi apresentada no capítulo 5 deste trabalho, no entanto, pode ser encontrada também em Deb (2004), Peñuela e Granada (2007).

Como já apresentado nas seções anteriores, os algoritmos multiobjetivos introduzem um operador de *dominância*, no qual é definido que uma solução $\vec{x}(1)$ domina $\vec{x}(2)$ ($\vec{x}(1)$ **p** $\vec{x}(2)$) se acontecem as seguintes condições:

1 – O indivíduo $\vec{x}(1)$ não é pior que $\vec{x}(2)$ em todos os objetivos;
2 – O indivíduo $\vec{x}(1)$ é estritamente melhor que $\vec{x}(2)$ em pelo menos um dos objetivos.

Desta forma, aplicando iterativamente estas regras sobre uma população qualquer de indivíduos de um problema de otimização multiobjetivo, podem-se estabelecer quais são as alternativas dominantes, conhecida como conjunto *não-dominado*. Os indivíduos restantes representam o conjunto de indivíduos *dominados*. Conseguindo estabelecer qual é o conjunto de indivíduos *não-dominados* ou *dominantes* considerando todo o espaço objetivo (f - v) tem-se a frente ou fronteira Pareto ótima. Assim, a função do Algoritmo Genético é mover o

conjunto não-dominado para regiões de melhor qualidade (Fronteira de Pareto ótima). Para o algoritmo proposto que utiliza o NSGA-II foi obtido, para o problema G5, a Fronteira de Pareto ótima mostrada na figura a seguir.

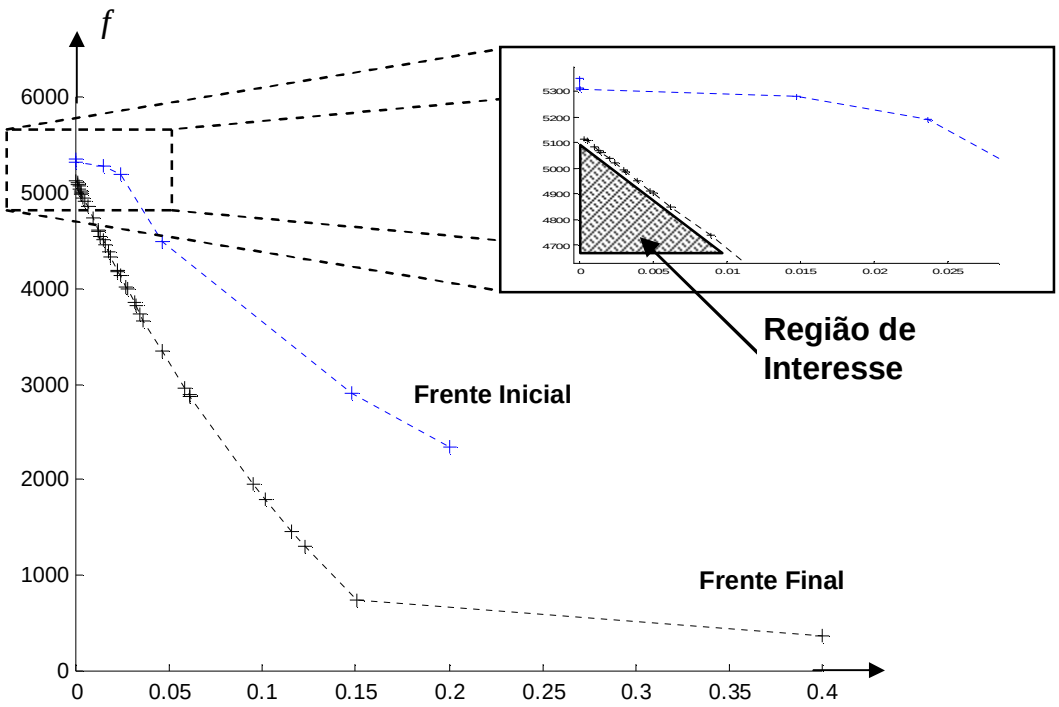

Figura 7.2: Fronteira de Pareto ótima e região de interesse para o problema G5.

Nesta figura, também se destaca a região de interesse do problema G5. Existem diferentes estratégias para intensificar a busca nesta região. No algoritmo proposto utilizou-se o *método de dominância guiada* cuja idéia foi apresentada no capítulo anterior e também pode ser encontrada em Deb (2004). No método de dominância guiada, escolhendo valores apropriados para os elementos a_{12} e a_{21} uma parcela da fronteira Pareto ótima original pode ser priorizada. Neste trabalho, para explorar a região de interesse mostrada na figura 7.2 foram utilizados os seguintes valores: $a_{12} = 0$ e $a_{21} = 1{,}33$.

No capítulo a seguir, apresentam-se com detalhes os 11 problemas testes que foram utilizados para avaliar o desempenho do algoritmo proposto.

CAPÍTULO 8

PROBLEMAS TESTES

8.1 Introdução

Neste capítulo são enunciados 11 problemas testes de otimização restrita. Estes problemas são encontrados em (KOZIEL; MICHALEWICZ, 1999) e foram utilizados para realização de testes que visam constatar a qualidade do algoritmo proposto aqui. Além de enunciá-los, apresentam-se as melhores soluções já encontradas e os gráficos da função objetivo e de suas respectivas restrições para alguns destes problemas. Obviamente, apenas foi possível fazer a ilustração gráfica para os problemas que possuem até duas variáveis de decisão, visto que, além desta dimensão não é mais possível realizar uma análise gráfica.

Este capítulo foi dividido em tópicos, sendo que, cada tópico representa a apresentação de cada um dos 11 problemas testes. Portanto, temos 11 tópicos que varia do problema G1 ao G11.

8.2 Problema G1

Minimizar $G1(\mathbf{r}\vec{x}) = 5x_1 + 5x_2 + 5x_3 + 5x_4 - 5\sum_{i=1}^{4} x_i^{\,2} - \sum_{i=5}^{13} x_i$

sujeito a seguintes restrições:

$$2x_1 + 2x_2 + x_{10} + x_{11} \le 10$$
$$-8x_1 + x_{10} \le 0$$
$$-2x_4 - x_5 + x_{10} \le 0$$
$$2x_1 + 2x_3 + x_{10} + x_{12} \le 10$$
$$-8x_2 + x_{11} \le 0$$
$$-2x_6 - 2x_7 + x_{11} \le 0$$
$$2x_2 + 2x_3 + x_{11} + x_{12} \le 10$$
$$-8x_3 + x_{12} \le 0$$
$$-2x_8 - x_9 + x_{12} \le 0$$

e as variáveis limitadas por:

$$0 \le x_i \le 1, \qquad i = 1, \mathbf{K}, 9$$
$$0 \le x_i \le 100, \quad i = 10,11,12$$
$$0 \le x_{13} \le 1$$

O problema possui 13 variáveis e 9 restrições lineares e, neste caso, não foi possível obter sua representação gráfica. A função G1 é quadrática com seu mínimo global em

$$\vec{x}^{*} = (1,1,1,1,1,1,1,1,1,3,3,3,1)$$

sendo $G1(\vec{x}^{*}) = -15$. Seis das nove restrições estão ativas no ótimo global, ou seja, todas estão ativas exceto as três seguintes:

$-8x_1 + x_{10} \le 0$, $-8x_2 + x_{11} \le 0$ e $-8x_3 + x_{12} \le 0$.

8.3 Problema G2

$$\text{Maximizar } G2(\mathbf{x}) = \left| \frac{\sum_{i=1}^{n} \cos^4(x_i) - 2\prod_{i=1}^{n} \cos^2(x_i)}{\sqrt{\sum_{i=1}^{n} i x_i^{\,2}}} \right|$$

sujeito às seguintes restrições:

$$\prod_{i=1}^{n} x_i \geq 0{,}75$$
$$\sum_{i=1}^{n} x_i \leq 7{,}5n$$

e as variáveis limitadas por:

$$0 \leq x_i \leq 10, \text{ para } 1 \leq i \leq n$$

A função G2 é não-linear e seu máximo global é desconhecido. No entanto, para $n = 20$ soluções $\vec{x}^*$ no qual $G2(\vec{x}^*) = 0{,}803553$ foram encontradas. A seguir ilustram-se alguns gráficos deste problema para $n = 2$. Primeiramente, mostra-se o gráfico da função objetivo $G2(\mathbf{x})$ que segue:

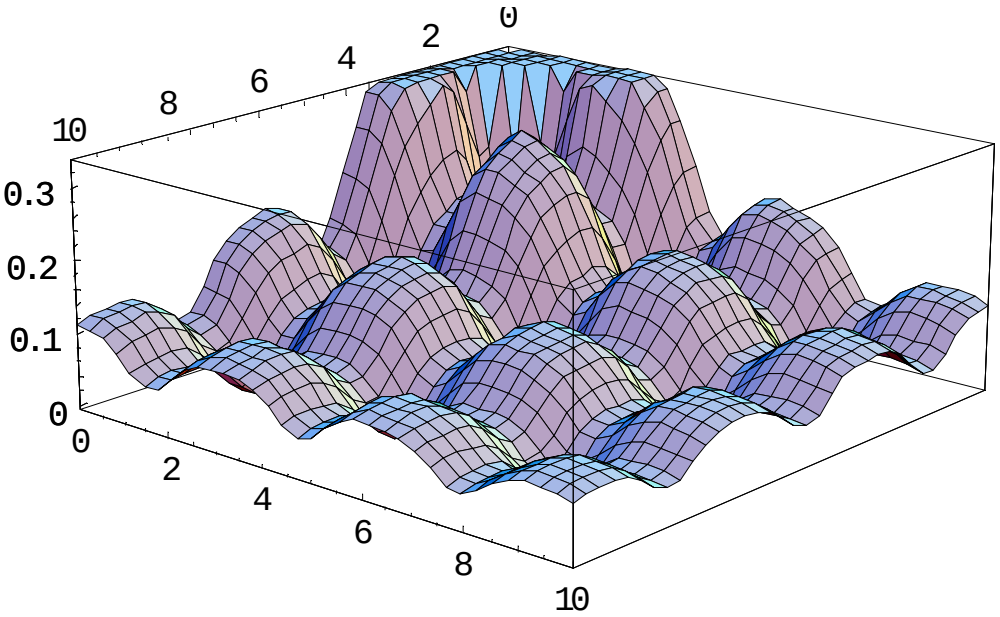

Figura 8.1: Gráfico da função $G2(\mathbf{x})$ para $n = 2$.

Analisando a figura 8.1, percebe-se que a função $G2(\vec{x})$ possui vários picos, sendo assim uma função é multimodal. O gráfico foi representado com as variáveis de decisão x_1, x_2 variando no intervalo de $[0,10]$. Na figura 8.2 que segue representam-se no mesmo eixo as curvas de nível das restrições $g_1(x_1, x_2)$ e $g_2(x_1, x_2)$ do problema G2, ou seja, $g_1(\vec{x}) = 0$ e $g_2(\vec{x}) = 0$.

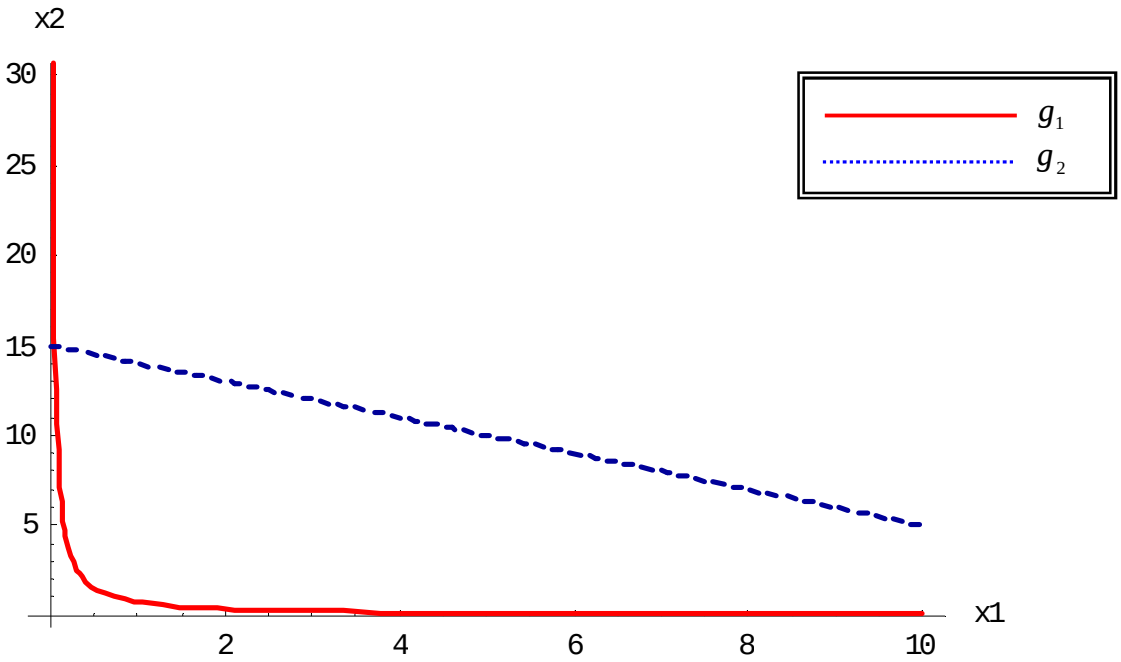

Figura 8.2: Gráfico das curvas de nível de $g_1(x_1, x_2)$ e $g_2(x_1, x_2)$ para o problema G2.

A região factível ocupa uma grande parte do espaço de busca e é dada pela região entre os gráficos de g_1 e g_2. Obviamente, deve ser respeitado o intervalo $[0,10]$ para as variáveis x_1 e x_2.

8.4 Problema G3

Maximizar $G3(\vec{x}) = (\sqrt{n})^n . \prod_{i=1}^{n} x_i$

sujeito a seguinte restrição:

$\sum_{i=1}^{n} x^2{}_i = 1$

e as variáveis limitadas por:

$0 \le x_i \le 1,$ para $1 \le i \le n$

A função G3 tem seu máximo global em $(x_1, \mathbf{K}, x_n) = \left(\frac{1}{\sqrt{n}}, \mathbf{K}, \frac{1}{\sqrt{n}} \right)$ e o valor da função neste ponto é 1. A seguir mostram-se alguns gráficos deste problema para $n = 2$. Primeiramente, ilustra-se o gráfico da função objetivo $G3(\vec{x})$ que segue:

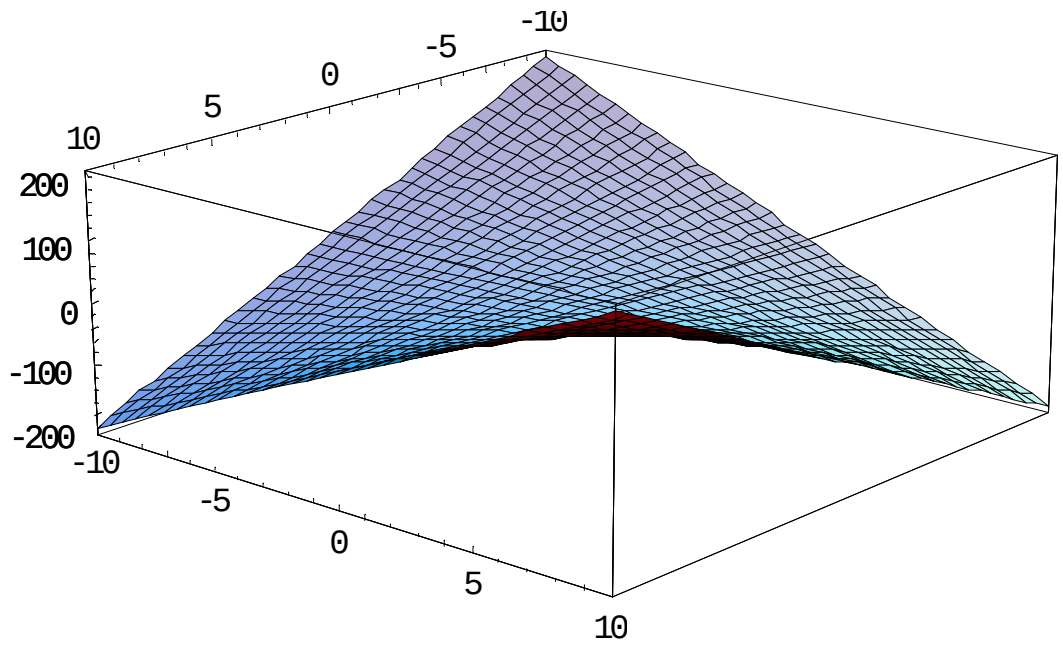

Figura 8.3: Gráfico da função $G3(\vec{x})$ para $n = 2$.

Analisando a figura 8.3, percebe-se que $G3(\vec{x})$ é uma função que, diferentemente de $G2(\vec{x})$, não possui vários picos. O gráfico foi representado, para melhor visualização, com as variáveis de decisão x_1, x_2 variando no intervalo de $[-10, 10]$, mesmo que, para este problema a exigência é que x_1, x_2 pertença ao intervalo $[0, 1]$. Na figura 8.4 que segue é mostrado o gráfico da restrição $h(\vec{x}) = 0$. Esta restrição é a única deste problema e, além disso, representa uma restrição igualdade.

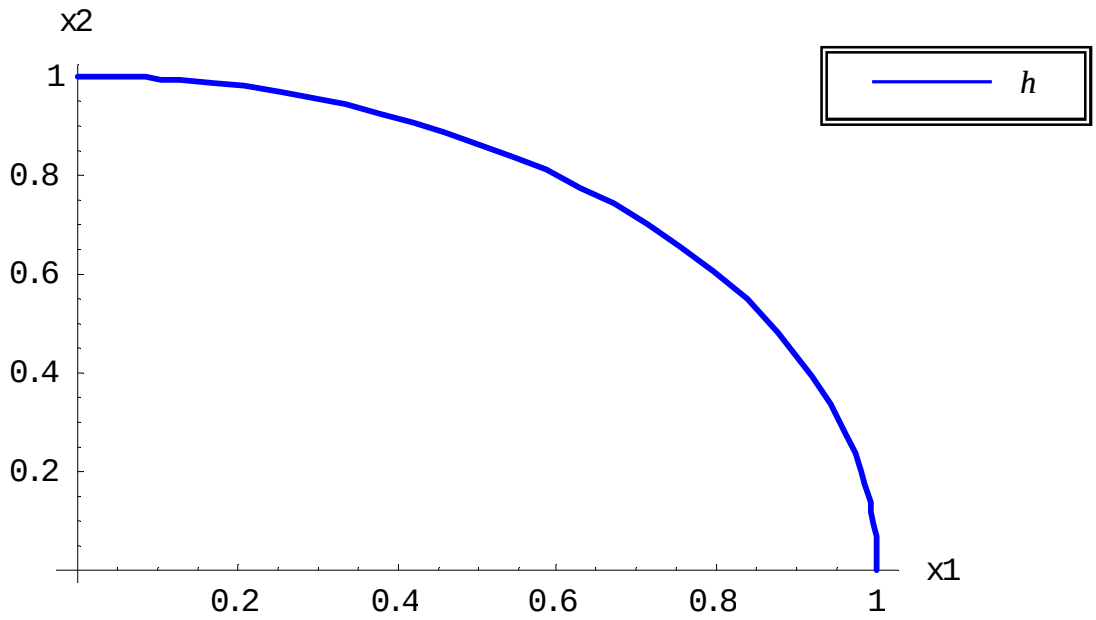

Figura 8.4: Gráfico da curva de nível de $h(x_1, x_2)$ para o problema G3.

A região factível ocupa apenas uma pequena parte do espaço de busca e é dada, exclusivamente, pela linha que representa o gráfico de $h(\mathbf{x}) = 0$. Isto mostra como restrições de igualdade restringem altamente o espaço de busca.

8.5 Problema G4

Minimizar $G4(\mathbf{x}) = 5{,}3578547x_3^2 + 0{,}8356891x_1x_3 + 37{,}293239x_1 - 40792{,}141$

sujeito as três seguintes duplas desigualdades:

$$0 \le 85{,}334407 + 0{,}0056858x_2x_5 + 0{,}0006262x_1x_4 - 0{,}0022053x_3x_5 \le 92$$
$$90 \le 80{,}51249 + 0{,}0071317x_2x_5 + 0{,}0029955x_1x_2 + 0{,}0021813x_3^2 \le 110$$
$$20 \le 9{,}300961 + 0{,}0047026x_3x_5 + 0{,}0012547x_1x_3 + 0{,}0019085x_3x_4 \le 25$$

e as variáveis limitadas por:

$$78 \le x_1 \le 102$$
$$33 \le x_2 \le 45$$
$$27 \le x_i \le 45 \quad \text{para} \quad i = 3,4,5$$

A solução ótima é:

$$\vec{x}^{*} = (78{,}0;\ 33{,}0;\ 29{,}995;\ 45{,}0;\ 36{,}776)$$

com $G4(\vec{x}^{*}) = -30665{,}5$. Duas restrições (a primeira desigualdade limitada superiormente e a terceira desigualdade limitada inferiormente) estão ativas no ponto ótimo. O problema G4 possui 5 variáveis e, neste caso, não é possível fazer sua representação gráfica.

8.6 Problema G5

Minimizar $G5(\mathbf{r}\vec{x}) = 3x_1 + 0{,}000001x_1^3 + 2x_2 + \frac{0{,}000002}{3}x_2^3$

sujeito a seguintes restrições:

$$x_4 - x_3 + 0{,}55 \geq 0$$
$$x_3 - x_4 + 0{,}55 \geq 0$$
$$1000\,\mathrm{sen}(-x_3 - 0{,}25) + 1000\,\mathrm{sen}(-x_4 - 0{,}25) + 894{,}8 - x_1 = 0$$
$$1000\,\mathrm{sen}(x_3 - 0{,}25) + 1000\,\mathrm{sen}(x_3 - x_4 - 0{,}25) + 894{,}8 - x_2 = 0$$
$$1000\,\mathrm{sen}(x_4 - 0{,}25) + 1000\,\mathrm{sen}(x_4 - x_3 - 0{,}25) + 1294{,}8 = 0$$

e as variáveis limitadas por:

$0 \leq x_i \leq 1200$ para $i = 1, 2$
$-0{,}55 \leq x_i \leq 0{,}55$ para $i = 3, 4$

A melhor solução conhecida é:

$$\vec{x}^{*} = (679{,}9453;\ 1026{,}067;\ 0{,}1188764;\ -0{,}3962336)$$

e $G5(\vec{x}^{*}) = 5126{,}4976$. Este problema é bastante complexo, já que o conjunto de restrições é composto por três restrições de igualdade, ocasionando desta forma a diminuição drástica da

região factível. Este problema possui 4 variáveis e, desta forma, não é possível fazer sua representação gráfica.

8.7 Problema G6

Minimizar $G6(\mathbf{r}\vec{x}) = (x_1 - 10)^3 + (x_2 - 20)^3$

sujeito às seguintes restrições não-lineares:

$$(x_1 - 5)^2 + (x_2 - 5)^2 - 100 \geq 0$$
$$-(x_1 - 6)^2 + (x_2 - 5)^2 + 82{,}81 \geq 0$$

e as variáveis limitadas por:

$$13 \leq x_1 \leq 100$$
$$0 \leq x_2 \leq 100$$

A solução ótima global conhecida é:

$$\vec{x}^* = (14{,}095; 0{,}84296)$$

e $G6(\vec{x}^*) = -6961{,}81381$. Ambas as restrições estão ativas no ponto ótimo. O problema possui duas variáveis e, a seguir ilustram-se alguns gráficos deste problema. Primeiramente, mostra-se o gráfico da função objetivo $G6(\mathbf{r}\vec{x})$ que segue:

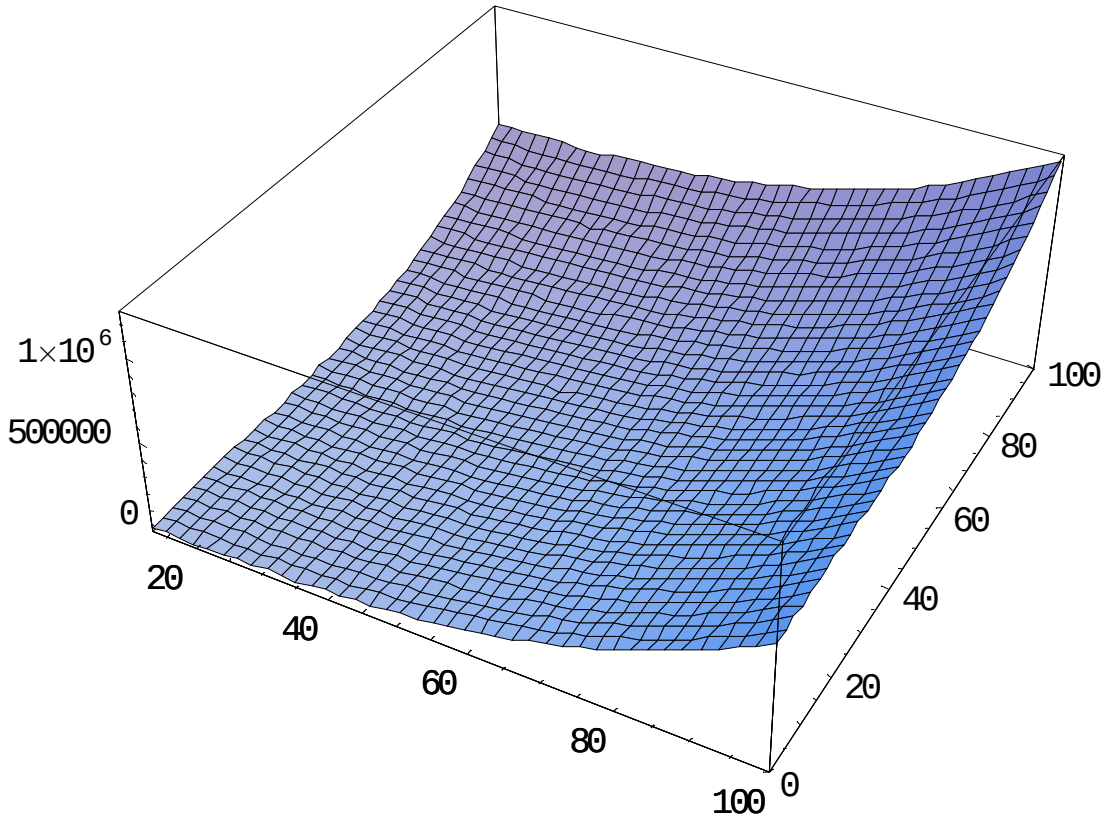

Figura 8.5: Gráfico da função $G6(\vec{x})$.

Analisando a figura 8.5, percebe-se que $G6(\vec{x})$ é uma função que, como $G3(\vec{x})$, não possui grandes elevações. O gráfico foi representado com as variáveis de decisão x_1 variando no intervalo de $[13,100]$ e x_2 variando no intervalo de $[0,100]$. Na figura 8.6 que segue, representam-se no mesmo eixo as curvas de nível das duas restrições $g_1(x_1,x_2)$ e $g_2(x_1,x_2)$ do problema G6, ou seja, $g_1(\vec{x}) = 0$ e $g_2(\vec{x}) = 0$.

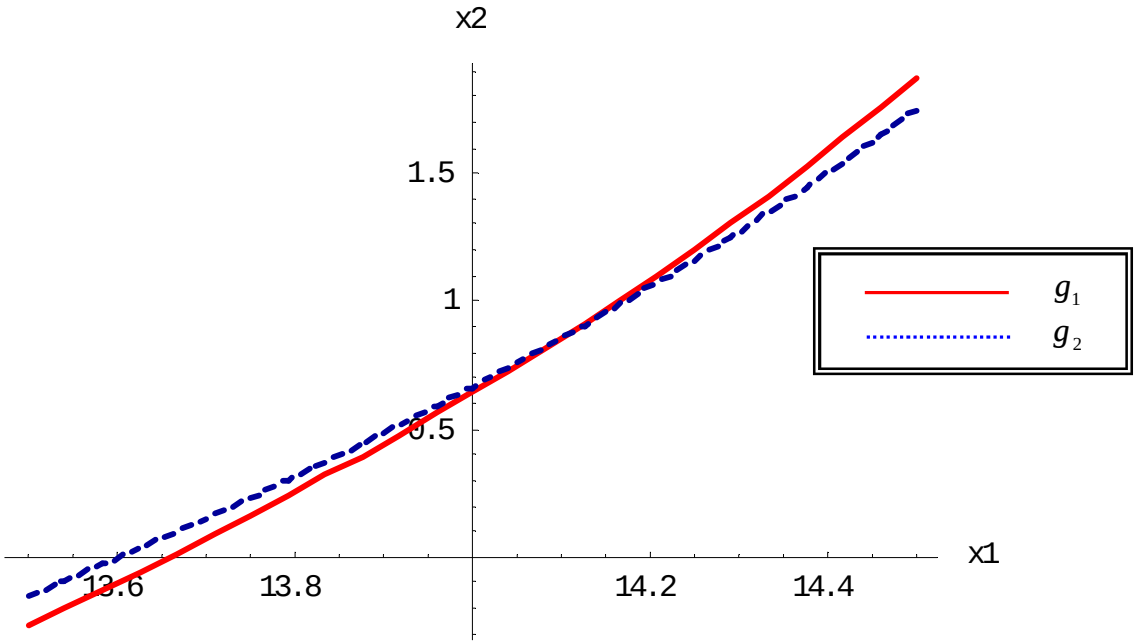

Figura 8.6: Gráfico das curvas de nível de $g_1(x_1, x_2)$ e $g_2(x_1, x_2)$ para o problema G6.

Para se ter uma idéia da dificuldade deste problema a região factível é bastante pequena e é dada por uma estreita fenda delimitada por 2 arcos de círculos que são as curvas de níveis das restrições $g_1(\vec{x}) = 0$ e $g_2(\vec{x}) = 0$. Na figura 8.7, é esboçado, simultaneamente, o gráfico da função objetivo $G6(\vec{x})$ e a estreita fenda que representa a região factível.

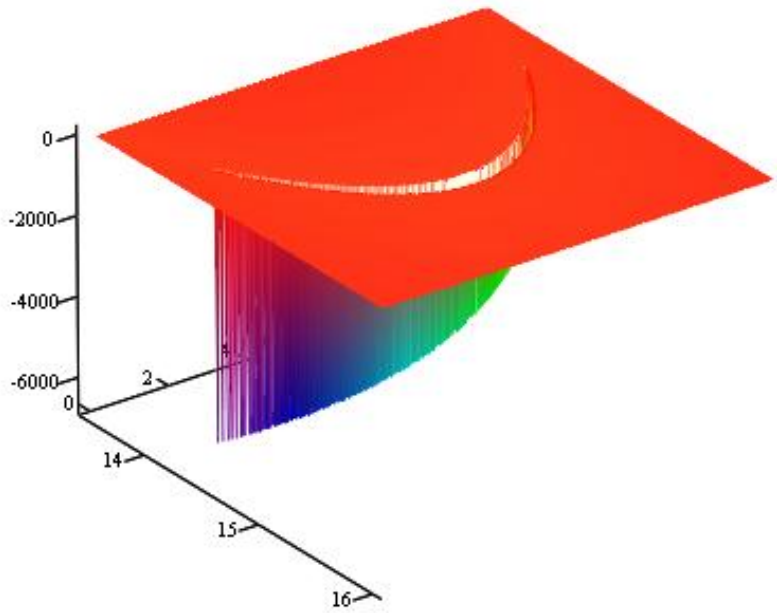

Figura 8.7: Representação de $G6(\vec{x})$ e as restrições $g_1(x_1, x_2)$ e $g_2(x_1, x_2)$.

8.8 Problema G7

Minimizar

$$G7(\mathbf{r}\vec{x}) = x_1^2 + x_2^2 + x_1 x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 + (x_5 - 3)^2 + 2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45$$

sujeito às seguintes restrições:

$$105 - 4x_1 - 5x_2 + 3x_7 - 9x_8 \geq 0$$
$$-10x_1 + 8x_2 + 17x_7 - 2x_8 \geq 0$$
$$8x_1 - 2x_2 - 5x_9 + 2x_{10} + 12 \geq 0$$
$$3x_1 - 6x_2 - 12(x_9 - 8)^2 + 7x_{10} \geq 0$$
$$-3(x_1 - 2)^2 - 4(x_2 - 3)^2 - 2x_3^2 + 7x_4 + 120 \geq 0$$
$$-x_1^2 - 2(x_2 - 2)^2 + 2x_1 x_2 - 14x_5 + 6x_6 \geq 0$$
$$-5x_1^2 - 8x_2 - (x_3 - 6)^2 + 2x_4 + 40 \geq 0$$
$$-0.5(x_1 - 8)^2 - 2(x_2 - 4)^2 - 3x_5^2 + x_6 + 30 \geq 0$$

e as variáveis limitadas por:

$$-10{,}0 \leq x_i \leq 10{,}0; \qquad i = 1, \mathbf{K}, 10$$

O problema tem 3 restrições lineares e 5 não-lineares, a função G7 é quadrática e tem seu mínimo global em:

$$\vec{x}^* = (2{,}171996;\ 2{,}363683;\ 8{,}773926;\ 5{,}095984;\ 0{,}9906548;\ 1{,}430574;\ 1{,}321644;\ 9{,}828726;\ 8{,}280092;\ 8{,}375927)$$

sendo $G7(\vec{x}^*) = 24{,}3062091$. Seis das oito restrições estão ativas no ótimo global, ou seja, todas estão ativas exceto as duas últimas. Novamente este problema possui mais que duas variáveis e, sendo assim, não será realizada sua representação gráfica.

8.9 Problema G8

Maximizar $G8(\mathbf{r}\vec{x}) = \dfrac{\text{sen}^3(2p\ x_1).\text{sen}(2p\ x_2)}{x_1^3.(x_1 + x_2)}$

sujeito às seguintes restrições :

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

e as variáveis limitadas por:

$$0 \leq x_1 \leq 10$$
$$0 \leq x_2 \leq 10$$

A função G8 apresenta vários ótimos locais e os picos mais altos estão localizados ao longo do eixo *x* como, por exemplo, $G8(\vec{x}) = (0{,}00015; 0{,}0225) > 1540$**.** No entanto, na região factível, G8 possui dois valores de aptidão quase igual a 0,1; ou seja, encontrou-se o valor $G8(\vec{x}) = 0{,}095825$. Este problema possui duas variáveis e, sendo assim, é feita sua representação gráfica. Primeiramente, como foi feito em exemplos anteriores, representa-se o gráfico da função objetivo $G8(\vec{x})$ que segue:

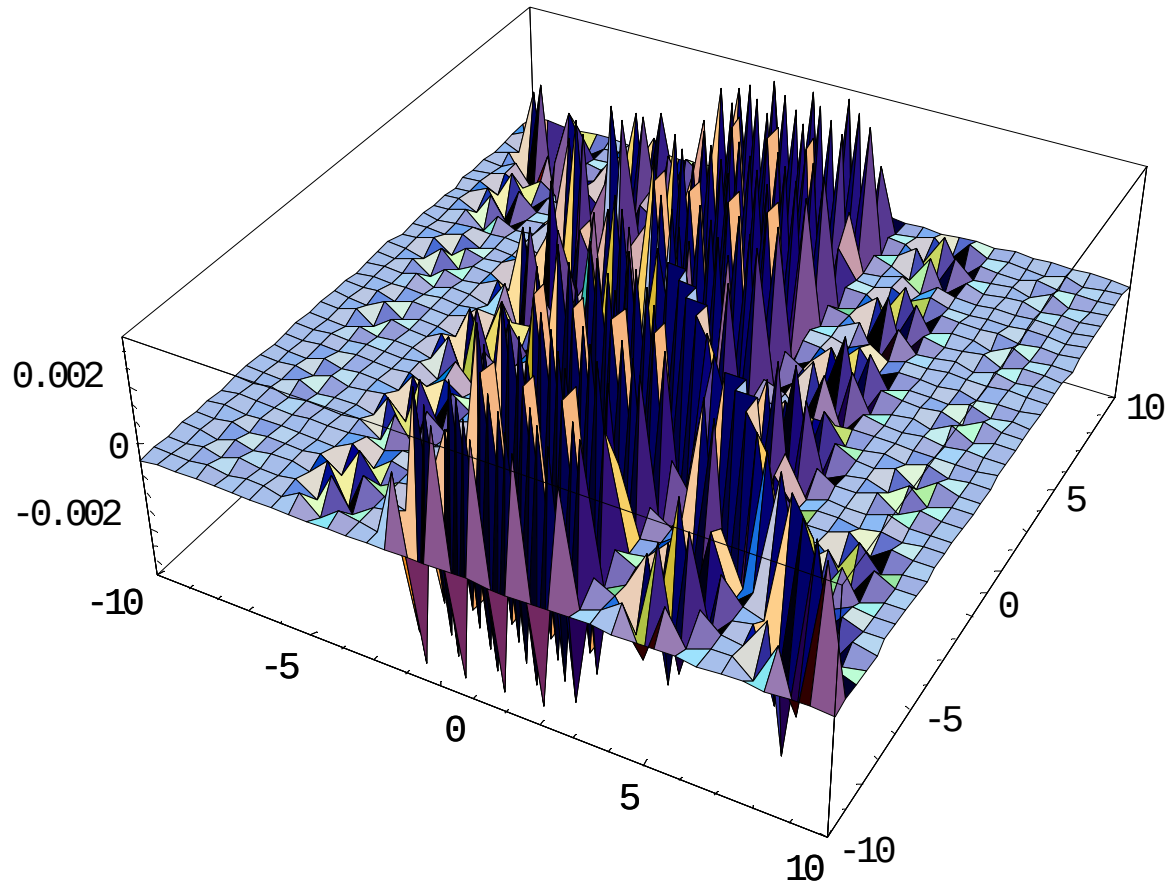

Figura 8.8: Gráfico da função $G8(\vec{x})$.

Analisando a Figura 8.8, percebe-se que $G8(\vec{x})$ possui vários picos, o que comprova que esta função possui vários ótimos locais. O gráfico foi representado, para melhor visualização, com as variáveis de decisão x_1, x_2 variando no intervalo de $[-10,10]$, mesmo que, para este problema a exigência é que x_1, x_2 pertença ao intervalo $[0,10]$. Na figura 8.9 que segue, representam-se no mesmo eixo as curvas de nível das restrições $g_1(x_1, x_2)$ e $g_2(x_1, x_2)$ para o problema G8.

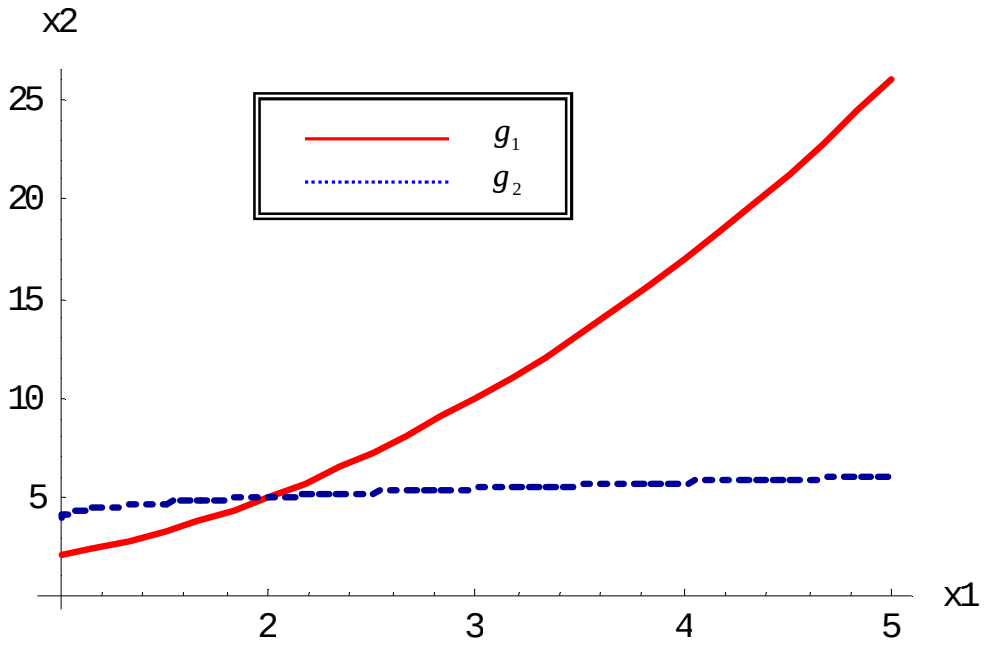

Figura 8.9: Gráfico das Curvas de nível de $g_1(x_1, x_2)$ e $g_2(x_1, x_2)$ para o problema G8.

A região factível ocupa uma pequena porção do espaço de busca e é dada pela região menor entre os gráficos de g_1 e g_2.

8.10 Problema G9

Minimizar

$$G9(\mathbf{r}\vec{x}) = (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2 + 10x_5^6 + 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7$$

sujeito as seguintes restrições :

$$127 - 2x_1^2 - 3x_2^4 - x_3 - 4x_4^2 - 5x_5 \geq 0$$
$$196 - 23x_1 - x_2^2 - 6x_6^2 + 8x_7 \geq 0$$
$$282 - 7x_1 - 3x_2 - 10x_3^2 - x_4 + x_5 \geq 0$$
$$-4x_1^2 - x_2^2 + 3x_1x_2 - 2x_3^2 - 5x_6 + 11x_7 \geq 0$$

e as variáveis limitadas por:

$-10{,}0 \leq x_i \leq 10{,}0$ para $i = 1, \mathbf{K}, 7$.

O problema possui 4 restrições não-lineares, a função G9 é não-linear e tem seu mínimo global em:

$$\vec{x}^{*} = (2{,}330449; 1{,}951372; -0{,}4775414; 4{,}365726; -0{,}6244870; 1{,}038131; 1{,}594227)$$

sendo $G9(\vec{x}^{*}) = 680{,}6300573$. Duas das quatro restrições estão ativas no ótimo global (a primeira e a última). Este problema possui 7 variáveis e, desta forma, não é feita sua representação gráfica.

8.11 Problema G10

Minimizar $G10(\vec{x}) = x_1 + x_2 + x_3$

sujeito às seguintes restrições :

$$1 - 0{,}0025(x_4 + x_6) \geq 0$$
$$1 - 0{,}01(x_8 + x_5) \geq 0$$
$$x_2 x_7 - 1250x_5 - x_2 x_4 + 1250x_4 \geq 0$$
$$1 - 0{,}0025(x_5 + x_7 - x_4) \geq 0$$
$$x_1 x_6 - 833{,}33252x_4 - 100x_1 + 83333{,}333 \geq 0$$
$$x_3 x_8 - 1250000 - x_3 x_5 + 2500x_5 \geq 0$$

e as variáveis limitadas por:

$$100 \leq x_1 \leq 10000$$
$$1000 \leq x_i \leq 10000 \quad \text{para } i = 2, 3.$$
$$10 \leq x_i \leq 1000 \quad \text{para } i = 4, \mathbf{K}, 8.$$

O problema tem 3 restrições lineares e 3 não-lineares; a função G10 é linear e tem seu mínimo global em:

$$\vec{x}^{*} = (579{,}3167; 1359{,}943; 5110{,}071; 182{,}0174; 295{,}5985; 217{,}9799; 286{,}4162; 395{,}5979)$$

sendo $G10(\vec{x}^{*}) = 7049{,}330923$. Todas as seis restrições estão ativas no ótimo global. Novamente, este problema possui mais que duas variáveis e, portanto, não é feita sua representação gráfica.

8.12 Problema G11

Minimizar $G11(\mathbf{r}) = x_1^2 + (x_2 - 1)^2$

sujeito a uma restrição não-linear:

$$x_2 - x_1^2 = 0$$

e as variáveis limitadas por:

$-1 \le x_i \le 1$ para $i = 1, 2.$

A solução global conhecida é $\vec{x}^{*} = (\pm 0{,}70711; 0{,}5)$, $G11(\vec{x}^{*}) = 0{,}75000455$. Este problema possui 2 variáveis e, com isso, é feita sua representação gráfica. Primeiramente, mostrar-se o gráfico da função objetivo $G11(\mathbf{r}\vec{x})$ que segue:

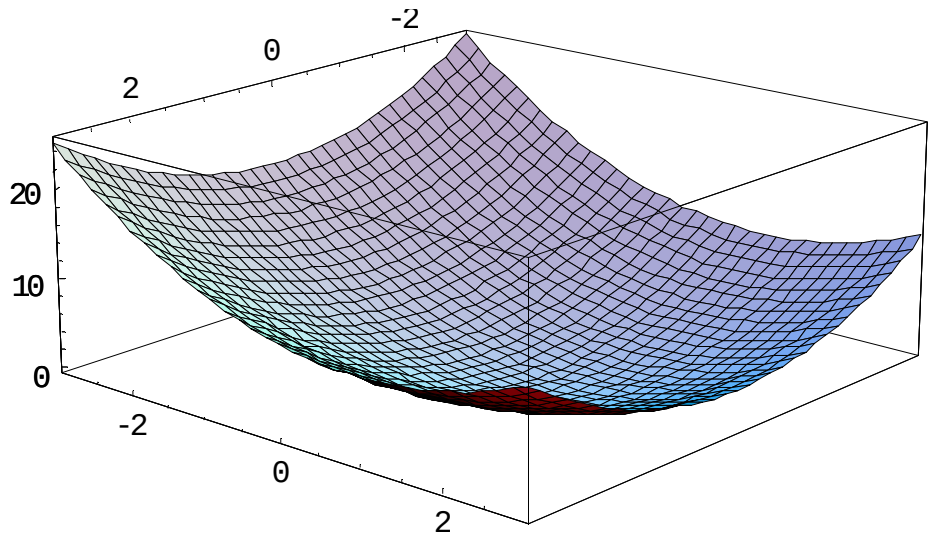

Figura 8.10: Gráfico da função $G11(\vec{x})$.

Analisando a figura 8.10, percebe-se que $G11(\vec{x})$ é uma função que não possui grandes complicações, já que tal função é quadrática. O gráfico foi representado, para melhor visualização, com as variáveis de decisão x_1, x_2 variando no intervalo de $[-3, 3]$, mesmo que, para este problema a exigência é que x_1, x_2 pertença ao intervalo $[-1, 1]$. Na figura 8.11 que segue, representa-se o gráfico tridimensional de $h(\vec{x}) = x_2 - x_1^2$.

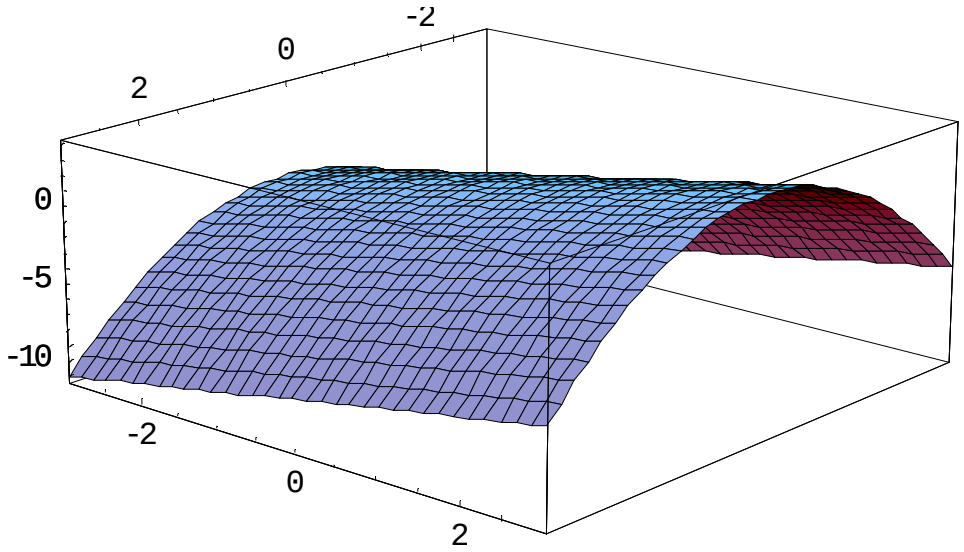

Figura 8.11: Gráfico tridimensional de $h(\vec{x}) = x_2 - x_1^2$.

A restrição do problema G11 é uma restrição não-linear e de igualdade, portanto, isto acentua a dificuldade do problema. Na figura a seguir, representa-se o gráfico da restrição $h(\vec{x}) = 0$.

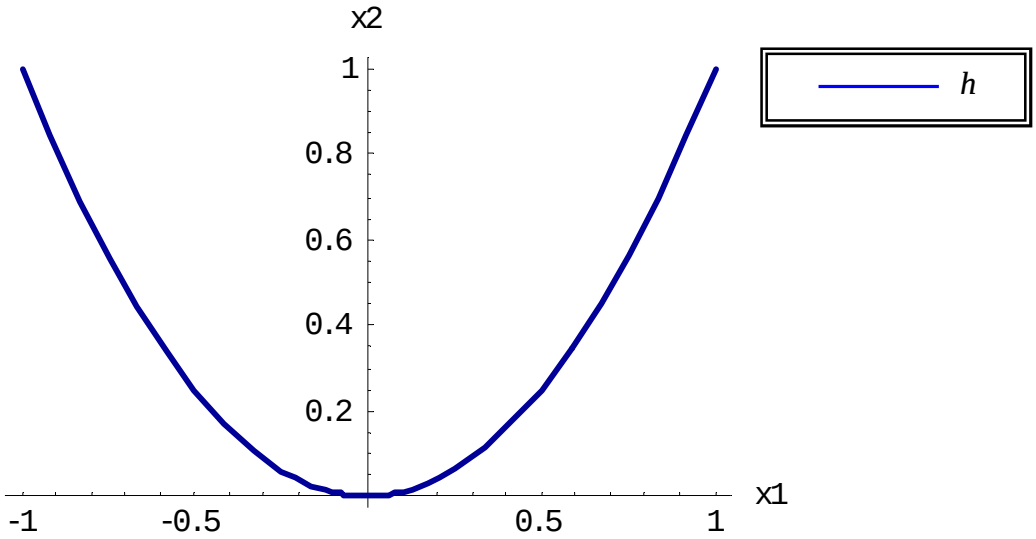

Figura 8.12: Gráfico da curva de nível de $h(x_1, x_2)$ para o problema G11.

A região factível ocupa apenas uma pequena parte do espaço de busca e é dada, exclusivamente, pela linha que representa o gráfico de $h(\vec{x}) = 0$. Esta região pode ser visualizada na figura 8.12. Na figura 8.13 é mostrado, simultaneamente, os gráficos da função objetivo $G11(\vec{x})$, de $h(\vec{x}) = x_2 - x_1^2$ e a restrição $h(\vec{x}) = 0$ quando relaxada por um valor d.

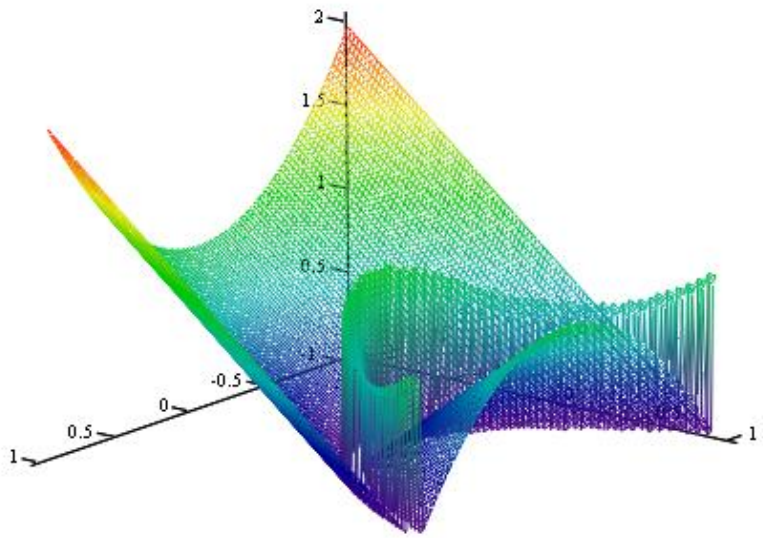

Figura 8.13: Representação de $G11(\vec{x})$, $h(\vec{x}) = x_2 - x_1^2$ e a restrição $h(\vec{x}) = 0$ relaxada.

Para se ter uma idéia da dificuldade deste problema, a região factível está situada ao longo da trilha, representada na figura 8.13, criada quando a restrição $h(\vec{x}) = 0$ é relaxada por um valor d .

No próximo capítulo, apresentam-se os testes e resultados obtidos da aplicação do algoritmo proposto aos 11 problemas testes vistos neste capítulo.

CAPÍTULO 9

TESTES E RESULTADOS

Neste capítulo apresentam-se os resultados obtidos da aplicação do algoritmo proposto aos 11 problemas testes apresentados no capítulo anterior. Através dos resultados obtidos é feita uma comparação com as soluções já obtidas por outros autores utilizando estes mesmos problemas. Além dos resultados, apresentam-se as configurações utilizadas e realiza-se um comentário das dificuldades envolvidas na resolução de alguns destes problemas.

9.1 Parâmetros Utilizados

Nesta seção apresentam-se os parâmetros de controle e as estratégias utilizadas na simulação do algoritmo proposto aos 11 problemas testes.

A taxa de recombinação (r_r) e a taxa de mutação (r_m) utilizadas foram iguais para todos os casos testes, ou seja, $r_r = 90\%$ e $r_m = 1\%$. O tamanho da população (n_p) para todos os casos também foi a mesma, ou seja, $n_p = 15$ indivíduos. O número de iterações (gerações) máximo foram de 5000 e foram executadas 30 simulações para cada caso. Para os problemas testes G2 e G3 utilizou-se $k = 20$ (número de variáveis de decisão). Na primeira fase onde é aplicada a *Diminuição Progressiva da Tolerância de Aceitação das Restrições*

Complexas (DPTARC) utilizou-se $e_{\max} = 0{,}4$ e $e_{\min} = 0{,}0001$, e para a estratégia de redução da margem de violação $e = (1-t)e$ utilizou-se $t = 0{,}5$.

No algoritmo proposto, não foi utilizado nenhuma codificação com cadeias binárias, ou seja, utilizou a codificação dos indivíduos com parâmetros reais. Portanto, isto implicou na mudança dos operadores genéticos de recombinação e mutação. Os operados genéticos utilizados para simulação do algoritmo foram os mais simples, ou seja, a Recombinação Linear e a Mutação Aleatória em que as formas de operar foram apresentadas no capítulo 4 deste trabalho.

9.2 Características Gerais

Nesta seção, apresentam-se algumas características gerais dos 11 problemas teste e, discutem-se como tais características influenciaram na obtenção de algumas soluções. Especificamente, analisam-se como estas características influenciaram na obtenção de soluções factíveis na primeira fase do algoritmo proposto. A seguir, apresenta-se uma tabela reproduzida de (VENKATRAMAN; YEN, 2005, KOZIEL; MICHALEWICZ, 1999). Nesta tabela são citados para cada um dos problemas características tais como: objetivo do problema, número de variáveis (n), tipo da função objetivo (f), porcentagem de factibilidade em relação ao espaço de busca (r), número de desigualdades lineares (DL), número de igualdades não-lineares (EN), número de desigualdades não-lineares (DN) e quantidade de restrições ativas no ponto ótimo (a).

Tabela 9.1: Algumas características dos 11 problemas testes.

Problema	Objetivo	n	Tipo de f	r	DL	EN	DN	a
G1	Minimizar	13	Quadrática	0,0111%	9	0	0	6
G2	Maximizar	k	Não linear	99,8474%	0	0	2	1
G3	Maximizar	k	Polinomial	0,0000%	0	1	0	1
G4	Minimizar	5	Quadrática	52,1230%	0	0	6	2
G5	Minimizar	4	Cúbica	0,0000%	2	3	0	3
G6	Minimizar	2	Cúbica	0,0066%	0	0	2	2
G7	Minimizar	10	Quadrática	0,0003%	3	0	5	6
G8	Maximizar	2	Não linear	0,8560%	0	0	2	0
G9	Minimizar	7	Polinomial	0,5152%	0	0	4	2
G10	Minimizar	8	Linear	0,0010%	3	0	3	6
G11	Minimizar	2	Quadrática	0,0000%	0	1	0	1

A porcentagem de factibilidade em relação ao espaço de busca (r) é dada por $r = \frac{|F \cap S|}{|S|}$, e foi determinada experimentalmente gerando 1000000 de soluções aleatoriamente no espaço de busca S e verificando se estas pertenciam a região factível F. Para os problemas G2 e G3 foi considerado $k = 50$ (número de variáveis de decisão) para a realização deste experimento (KOZIEL; MICHALEWICZ, 1999). No algoritmo proposto, como já foi dito, utilizou-se $k = 20$ (número de variáveis de decisão) para G2 e G3. Os problemas G3, G5 e G11 tiveram suas restrições de igualdade relaxadas por um valor d, e com isso, o valor da porcentagem de factibilidade r para estes problemas podem ser ligeiramente diferentes das apresentadas na tabela 9.1.

Analisando a tabela 9.1 é possível perceber claramente a variedade de características dos 11 problemas testes utilizados. Estas variedades estão relacionadas com os objetivos dos problemas (maximização e minimização), com os diferentes tipos de função objetivo (quadrática, não-linear, linear, polinomial e cúbica) e os diferentes tipos de restrições

(desigualdades não-lineares e lineares, e igualdades não-lineares). Para todos os 11 problemas aplicou-se o algoritmo proposto utilizando a mesma estrutura apresentada no capítulo 7.

Para produzir os resultados utilizando o algoritmo proposto realizou-se 30 simulações para cada caso. Uma das primeiras observações que se pode fazer em relação a este experimento é que em todas as tentativas de busca pela solução ótima foram encontradas soluções factíveis. A obtenção de soluções factíveis em todas as simulações é creditada à fase I do algoritmo proposto que utiliza a *Diminuição Progressiva da Tolerância de Aceitação das Restrições Complexas (DPTARC)* que busca, exclusivamente, a satisfação de restrições.

Nos problemas G2 e G4, a solução factível é sempre encontrada na população inicial gerada aleatoriamente. Isto acontece pelo fato das regiões factíveis destes problemas, ocuparem uma grande parte do espaço de busca. De fato, pois como se pode verificar na tabela 9.1, a porcentagem de factibilidade em relação ao espaço de busca **(** r **)** para estes problemas é dada, respectivamente, por 99,8474% e 52,1230% .

O problema G1 possui nove restrições lineares e, consequentemente, a região factível é convexa. Assim, não houve dificuldades para minimizar a distância representada pelos escalares de violação das restrições e obter soluções factíveis.

Os problemas G3 e G11 possuem somente uma restrição de igualdade não-linear. Para estes problemas, foi tolerada uma infactibilidade de $e = 0{,}0001$. Assim, obter soluções factíveis na primeira fase do algoritmo proposto não foi uma tarefa difícil.

O problema G5 possui cinco restrições compostas de igualdades não-lineares e desigualdades lineares e, por este motivo, encontrar soluções factíveis foi um pouco mais demorado. Realmente, o problema G5 é bastante restrito e porcentagem de factibilidade em relação ao espaço de busca **(** r **)** representado na tabela 9.1 é de 0,0000% . Para este problema, também foi aceita uma infactibilidade de $e = 0{,}0001$. O algoritmo proposto obteve um valor médio de 405,3 gerações para obter a primeira solução factível para este problema.

Os problemas G8 e G9 possuem as restrições compostas de desigualdades não-lineares. No entanto, obter soluções factíveis para estes problemas foi relativamente fácil.

Os problemas G7 e G10 possuem restrições compostas de desigualdades não-lineares e lineares, e estas características, dificultam a obtenção de soluções factíveis. Para o problema G10, o algoritmo proposto obteve um valor médio de 38,7 gerações para obter a primeira solução factível para este problema.

Considerando que, encontrar soluções factíveis independe da função objetivo na primeira fase do algoritmo proposto, pode-se esboçar algumas conclusões sobre como as restrições afetam a habilidade dos Algoritmos Genéticos em encontrar soluções factíveis (VENKATRAMAN; YEN, 2005):

- Em geral, restrições não-lineares introduzem mais dificuldades em encontrar soluções factíveis do que as restrições lineares. Isto pode ser compreendido reconhecendo que os Algoritmos Genéticos são técnicas de busca estocásticas que trabalham baseado nos valores de aptidão. Estes valores de aptidão é um indicador confiável de como cada solução está próxima da região factível quando há um mapeamento linear entre as variáveis de decisão e as restrições. Porém, quando o mapeamento se torna não-linear, então a distância entre soluções ligeiramente infactíveis e soluções completamente factíveis no espaço de decisão, pode ser desproporcional para as diferenças nos valores de violação de restrição. Consequentemente, um passo para a factibilidade no espaço das restrições, pode envolver uma busca exaustiva no espaço das variáveis de decisões.

- A combinação da taxa de factibilidade **(** r **)** e os tipos de restrições definem o grau de dificuldade nos problemas de satisfação das restrições. Como visto na tabela 9.1, os problemas G2 e G4 possuem um valor grande de r e, desta forma, soluções factíveis foram encontradas na população inicial gerada aleatoriamente mesmo estes problemas possuindo restrições não-lineares. Ao mesmo tempo, em problemas com baixo valor de r (como é o caso de G1), soluções factíveis também foram encontradas facilmente, isto acontece porque todas as restrições envolvidas em G1 são lineares. Como verificado, o problema G5 apresenta uma combinação de restrições não-lineares e, além disso, possui um baixo valor de r e, consequentemente, estas características causaram dificuldades na obtenção de soluções factíveis. A combinação de uma baixa taxa de factibilidade e restrições

não-lineares também foram responsáveis por uma demora maior na obtenção de soluções factíveis para os problemas G7 e G10.

9.3 Resultados Finais

Nesta seção, apresentam-se os resultados obtidos da aplicação do algoritmo proposto aos 11 problemas testes. Especificamente, apresentam-se os melhores resultados alcançados na segunda fase, ou seja, apresentam-se a melhor solução encontrada em todas as 30 simulações para cada problema. Estas soluções (valores ótimos e variáveis de decisão) são apresentadas nas tabelas a seguir.

Tabela 9.2: Melhor solução encontrada para o Problema G1.

PROBLEMA G1		
Valor Objetivo		**-14,99990**
Variáveis de Decisão	x_1	1,00000
	x_2	1,00000
	x_3	1,00000
	x_4	1,00000
	x_5	1,00000
	x_6	1,00000
	x_7	1,00000
	x_8	1,00000
	x_9	1,00000
	x_{10}	3,00000
	x_{11}	3,00000
	x_{12}	2,99990
	x_{13}	1,00000

Tabela 9.3: Melhor solução encontrada para o Problema G2.

PROBLEMA G2		
Valor Objetivo		**0,80318350**
Variáveis de Decisão	x_1	3,15616
	x_2	3,11157
	x_3	3,09893
	x_4	3,07220
	x_5	3,05592
	x_6	3,03985
	x_7	2,95829
	x_8	2,91929
	x_9	0,48135
	x_{10}	0,48921
	x_{11}	0,48045
	x_{12}	0,48541
	x_{13}	0,45520
	x_{14}	0,47170
	x_{15}	0,46356
	x_{16}	0,45162
	x_{17}	0,44490
	x_{18}	0,44345
	x_{19}	0,44910
	x_{20}	0,45714

Tabela 9.4: Melhor solução encontrada para o Problema G3.

PROBLEMA G3		
Valor Objetivo		**1,0000**
Variáveis de Decisão	x_1	0,22141
	x_2	0,22270
	x_3	0,22475
	x_4	0,22001
	x_5	0,22139
	x_6	0,22093
	x_7	0,23180
	x_8	0,22195
	x_9	0,22508
	x_{10}	0,22290
	x_{11}	0,22042
	x_{12}	0,22825
	x_{13}	0,22469
	x_{14}	0,22410
	x_{15}	0,22659
	x_{16}	0,22319
	x_{17}	0,22577
	x_{18}	0,22019
	x_{19}	0,22219
	x_{20}	0,22420

Tabela 9.5: Melhor solução encontrada para o Problema G4.

PROBLEMA G4		
Valor Objetivo		**-30665,53924**
Variáveis de Decisão	x_1	77,98979
	x_2	33,00143
	x_3	29,99719
	x_4	44,99918
	x_5	36,77691

Tabela 9.6: Melhor solução encontrada para o Problema G5.

PROBLEMA G5		
Valor Objetivo		**5126,33096**
Variáveis de Decisão	x_1	681,56445594786
	x_2	1024,29347473928
	x_3	0,11770661330
	x_4	-0,39676709250

Tabela 9.7: Melhor solução encontrada para o Problema G6.

PROBLEMA G6		
Valor Objetivo		**-6961,74716**
Variáveis de Decisão	x_1	14,09503
	x_2	0,84302

Tabela 9.8: Melhor solução encontrada para o Problema G7.

PROBLEMA G7		
Valor Objetivo		**24,404733**
Variáveis de Decisão	x_1	2,18023
	x_2	2,35489
	x_3	8,76073
	x_4	5,11438
	x_5	1,00067
	x_6	1,46093
	x_7	1,33106
	x_8	9,83093
	x_9	8,27997
	x_{10}	8,34914

Tabela 9.9: Melhor solução encontrada para o Problema G8.

PROBLEMA G8		
Valor Objetivo		**0,095825**
Variáveis de Decisão	x_1	1,22796
	x_2	4,24533

Tabela 9.10: Melhor solução encontrada para o Problema G9.

PROBLEMA G9		
Valor Objetivo		**680,73512**
Variáveis de Decisão	x_1	2,34006
	x_2	1,95034
	x_3	-0,47972
	x_4	4,36376
	x_5	-0,61939
	x_6	1,03889
	x_7	1,61663

Tabela 9.11: Melhor solução encontrada para o Problema G10.

PROBLEMA G10		
Valor Objetivo		**7060,11542**
Variáveis de Decisão	x_1	543,98517
	x_2	1473,15417
	x_3	5042,97607
	x_4	178,95523
	x_5	298,64359
	x_6	221,04350
	x_7	280,51682
	x_8	398,63681

Tabela 9.12: Melhor solução encontrada para o Problema G11.

PROBLEMA G11		
Valor Objetivo		**0,74991**
Variáveis de Decisão	x_1	0,70531
	x_2	0,49756

Com estes melhores resultados alcançados pelo algoritmo proposto para cada um dos problemas é feito um comparativo com os valores já alcançados por outros autores aplicando outras estratégias nestes mesmos 11 problemas. A tabela a seguir mostra esta comparação.

Tabela 9.13: Comparação dos melhores resultados.

Problema	Valor Ótimo	Koziel e Michalewicz (1999)	Runarsson e Yao (2000)	Farmani e Wright (2003)	Venkatraman e Yen (2005)	Algoritmo Proposto
Min. G1	-15	-14,7864	-15,0000	-15,0000	-14,9999	-14,99990
Max. G2	0,803553	0,799530	0,803515	0,802970	0,803190	0,80318350
Max. G3	1,0	0,9997	1,0000	1,0000	1,000	1,0000
Min. G4	-30665,5	-30664,900	-30665,539	-30665,500	-30665,5312	-30665,53924
Min. G5	5126,4976	NR	5126,4970	5126,9890	5126,63049	**5126,33096**
Min. G6	-6961,8138	-6952,100	-6961,814	-6961,800	-6961,17856	-6961,74716
Min. G7	24,3062091	24,620	24,307	24,480	24,410977	24,404733
Max. G8	0,095825	0,095825	0,095825	0,095825	0,095825	0,095825
Min. G9	680,6300573	680,91	680,63	680,64	680,7622	680,73512
Min.G10	7049,330923	7147,90	7054,32	7061,34	7060,5528	7060,11542
Min.G11	0,75000455	0,75	0,75	0,75	0,7490	0,74991

Analisando a última coluna da tabela 9.13, pode-se perceber que o algoritmo proposto teve um desempenho satisfatório, pois as melhores soluções obtidas estão bem próximas dos valores ótimos apresentados na segunda coluna da tabela.

Para o problema G1, o algoritmo proposto encontrou o valor -14,99990 que está muito próximo do ótimo global -15. Para o problema G2, o resultado foi bastante satisfatório, sendo que, obteve-se o valor 0,80318350 que está próximo do ótimo conhecido 0,803553. Para o problema G3 conseguiu-se alcançar o valor ótimo de 1,0000. Para o problema G4 obteve-se o valor -30665,53924 que está muito próximo do ótimo -30665,5.

Para o problema G6, G7 também se encontrou valores próximos dos ótimos. Para G6 o ótimo vale -6961,8138 e para G7 este valor é 24,3062091. O algoritmo proposto obteve, respectivamente, os valores -6961,74716 e 24,404733 para estes problemas.

O problema G8 foi bastante fácil de ser resolvido. O algoritmo proposto obteve em todas as simulações o valor ótimo 0,095825. Para o problema G9 obteve-se o valor 680,73512 que também está próximo do valor ótimo 680,6300573. Para o problema G10 o valor obtido de 7060,11542 ficou razoavelmente próximo do valor ótimo 7049,330923.

Para o problema G11 encontrou-se o valor 0,74991 que é melhor que o valor ótimo 0,75000455. Isto aconteceu porque foi tolerada uma pequena margem de infactibilidade e, desta forma, a solução obtida é levemente infactível. No entanto, a solução obtida pelo algoritmo proposto é bastante interessante, pois possui uma violação de restrição aceitável com uma melhoria no valor da função objetivo.

O problema G5 merece destaque, pois como aconteceu com o problema G11, foi obtido o valor 5126,33096 que é melhor que o valor ótimo 5126,4976. Além disso, o valor obtido pelo algoritmo proposto é melhor que todos os valores apresentados por outros autores. Novamente para o caso G5, a solução obtida é levemente infactível e, como em G11, esta solução possui uma violação aceitável com uma melhoria no valor objetivo. Para este problema, apesar de não mencionado na tabela 9.13, o algoritmo proposto pode encontrar, nas 30 simulações, mais duas soluções de menor função objetivo do que todos os valores reportados. A seguir, apresenta-se uma tabela com estas alternativas e suas respectivas violações das restrições.

Tabela 9.14: Valores dos objetivos, das variáveis e das violações de restrições dos melhores indivíduos encontrados para G5.

		Indivíduo 1	Indivíduo 2	Indivíduo 3	Melhor indivíduo reportado
Valores Objetivos		**5126,49440**	**5126,46908**	**5126,33096**	5126,4976
Valor das Variáveis	x_1	680,34472275030	678,06766325728	681,56445594786	679,9453
	x_2	1025,63930634336	1028,06265646234	1024,29347473928	1026,067
	x_3	0,11859118409	0,12021181155	0,11770661330	0,1188764
	x_4	-0,39636838464	-0,39559444472	-0,39676709250	-0,3962336
Violação das Restrições	r_1	-0,03504043126577	-0,03419374372697	-0,03552629420057	-0,03489
	r_2	-1,06495956873423	-1,06580625627303	-1,06447370579943	-1,06511
	r_3	0	0,00004589886487	0	0,00003303007691
	r_4	0	0,00052290857275	0	0,00024724085870
	r_5	0,00081724789993	0,00825463676415	0,03317435154986	-0,00009672673013

Analisando esta tabela, pode-se notar que os indivíduos 1 e 3 cumprem exatamente as restrições de igualdade r_3 e r_4, ou seja, $r_3 = 0$ e $r_4 = 0$. Todos os indivíduos cumprem as restrições de desigualdade r_1 e r_2, pois possuem valores negativos para estas restrições. A restrição de igualdade r_5 é desrespeitada por todos os indivíduos da tabela. No entanto, esta restrição possui menor violação para o melhor indivíduo reportado e pior violação para o indivíduo 3.

É importante notar que todos os indivíduos encontrados pelo algoritmo proposto para o caso G5 possuem violações de restrições aceitáveis. Sendo assim, é possível escolher entre esses indivíduos um com menor valor objetivo. No entanto, o indivíduo escolhido pode apresentar uma maior ou menor violação para uma ou outra restrição. A figura a seguir mostra todos os 4 indivíduos da tabela 9.14 organizados em frentes de dominância. Nesta figura foi considerada uma tolerância $e = 0{,}0001$ para calcular o vetor $v(X)$ que quantifica o grau de infactibilidade de cada um dos 4 indivíduos.

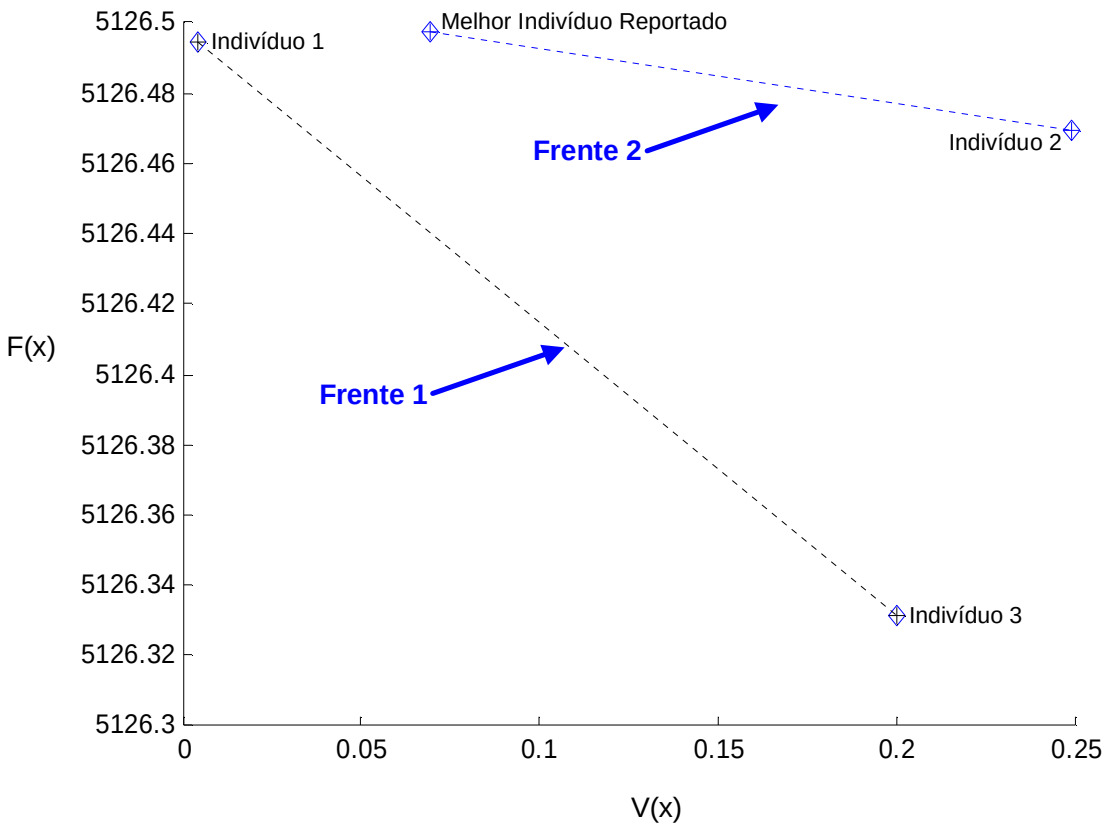

Figura 9.1: Organização das alternativas de solução para G5 em frentes de dominância.

Analisando esta figura é possível verificar que os indivíduos 1 e 3 pertencem a primeira frente de dominância (Frente 1). Já o melhor indivíduo reportado e o indivíduo 2 pertencem a segunda frente de dominância (Frente 2). Sendo assim, os indivíduos 1 e 3 pertencem à frente de melhor qualidade. Em particular, é possível verificar que o indivíduo 1 domina o melhor indivíduo reportado, ou seja, o indivíduo 1 apresenta, simultaneamente, valor objetivo e coeficiente de violação de restrição menor que o melhor indivíduo reportado. Isto mostra que, de fato, a solução 1 encontrada pelo algoritmo proposto supera o melhor indivíduo reportado.

No capítulo a seguir, apresentam-se as conclusões obtidas do desenvolvimento deste trabalho.

CAPÍTULO 10

CONCLUSÕES

10.1 Conclusão Geral

Neste trabalho, foi proposto um Algoritmo Genético composto de duas fases. A qualidade deste algoritmo foi validada através de 11 problemas testes encontrados na literatura especializada. Analisando a tabela 9.13 apresentada no capítulo anterior, que faz um comparativo dos resultados alcançados, verificou-se que os valores obtidos pelo algoritmo proposto são bastante próximos aos alcançados por Venkatraman e Yen (2005). Esta similaridade dos resultados está no fato da filosofia adotada para a elaboração do algoritmo proposto ser semelhante à utilizada por estes autores. A única diferença foi que, no algoritmo proposto, a filosofia utilizada por Venkatraman e Yen (2005) foi aperfeiçoada utilizando as seguintes idéias adicionais: a *Diminuição Progressiva da Tolerância de Aceitação das Restrições Complexas (DPTARC)* na primeira fase e a intensificação de busca na região de interesse utilizando o conceito de *dominância guiada*.

Os dados a seguir foram reproduzidos de (VENKATRAMAN; YEN, 2005) e resume o desempenho obtido na primeira fase do algoritmo por estes autores. Os autores realizaram 50 simulações para cada caso e relaxaram as restrições de igualdades usando uma tolerância

$e = 0{,}001$ para facilitar a obtenção de soluções factíveis. A sigla MGF representa o número médio de gerações para encontrar a primeira solução factível.

Tabela 10.1: Resultados obtidos em (VENKATRAMAN; YEN, 2005) na primeira fase.

Problema	MGF	Soluções Infactíveis Produzidas
Min. G1	11,24	0
Max. G2	0	0
Max. G3	31,68	0
Min. G4	0	0
Min. G5	1807,82	0
Min. G6	289,52	0
Min. G7	53,22	0
Max. G8	9,28	0
Min. G9	5,84	0
Min.G10	99,86	0
Min.G11	13,32	0

A incorporação da *Diminuição Progressiva da Tolerância de Aceitação das Restrições Complexas (DPTARC)* ao algoritmo proposto mostrou um melhor desempenho na primeira fase do que o algoritmo apresentado em (VENKATRAMAN; YEN, 2005). O caso G5 é altamente restrito e, deste modo, encontrar uma solução factível é uma tarefa difícil. Particularmente para este caso, o número médio de gerações para encontrar a primeira solução factível em (VENKATRAMAN; YEN, 2005) sobre 50 simulações do algoritmo e com $e = 0{,}001$ foram de 1807,82 (ver tabela 10.1). Com a incorporação da *Diminuição Progressiva da Tolerância de Aceitação das Restrições Complexas (DPTARC)* conseguiu-se uma média de 405,3 gerações sobre 30 simulações do algoritmo e com $e = 0{,}0001$. O uso de uma menor tolerância permitiu obter um número maior de soluções não-dominadas na região de interesse. Outro caso que é bastante restrito é o problema G10. A média reportada em (VENKATRAMAN; YEN, 2005) foi de 99,86 (ver tabela 10.1). A média obtida aplicando a

Diminuição Progressiva da Tolerância de Aceitação das Restrições Complexas (DPTARC) foi de 38,7. Nesta fase, ambas as propostas obtiveram soluções factíveis em todas as simulações do algoritmo.

Na segunda fase, o algoritmo proposto teve um desempenho similar aos valores reportados em (VENKATRAMAN; YEN, 2005). No entanto, para o problema G5, o método proposto conseguiu encontrar, em 30 simulações, 3 soluções com menor função objetivo que todos os outros resultados apresentados na tabela 9.13 do capítulo anterior. Estas soluções possuem graus de violação aceitáveis (ver tabela 9.14 do capítulo anterior). Sendo assim, é possível escolher uma solução com melhor função objetivo que a do melhor indivíduo reportado a troco de uma violação maior ou menor em uma ou outra restrição. Em especial, o indivíduo 1 encontrado pelo algoritmo proposto domina o melhor indivíduo reportado (ver figura 9.1 do capítulo anterior). Isto mostrou que o indivíduo 1 é de melhor qualidade que o melhor indivíduo reportado.

O algoritmo proposto apresentou a vantagem de não utilizar parâmetros que dependem de informações subjetivas do problema e, deste modo, mostrou-se bastante genérico, pois mesmo os problemas testes possuindo características bem diversificadas, os parâmetros de controle utilizados foram os mesmos para todos estes problemas. A proposta apresentada pode ser abordada em problemas de diferentes campos da engenharia e, preferencialmente, em problemas que possuem variáveis contínuas e são altamente restritos. Enfim, conclui-se que o algoritmo proposto apresentou um bom desempenho, visto que, foram obtidos valores ótimos próximos, iguais e até melhores que os valores ótimos já reportados na literatura especializada.

10.2 Sugestões para Trabalhos Futuros

No algoritmo proposto, os testes realizados utilizaram a recombinação linear e mutação aleatória usando a codificação real. Estas estratégias são as mais simples e, portanto, para trabalhos futuros, podem-se incorporar outras estratégias de recombinação e mutação usando codificação real tais como: recombinação blend, recombinação binária simulada,

recombinação simplex, mutação não-uniforme, mutação polinomial entre outras. A utilização de algoritmos que incorporam o conceito de diversidade lateral, como por exemplo, o NSGA-II controlado, cuja teoria pode ser encontrada em (DEB, 2004), permitem uma busca de melhor qualidade na região de interesse. Outra estratégia interessante que pode ser aplicada ao algoritmo proposto é o Path Relinking, pois armazenando, durante o processo, duas ou mais soluções elites, é possível através destas soluções gerarem novas soluções de qualidade pertencente também à frente não-dominada. Enfim, a utilização das idéias apresentadas permitirá, possivelmente, melhorar alguns resultados.

REFERÊNCIAS

BARCELLOS, J. C. H. **Algoritmos genéticos adaptativos:** um estudo comparativo. 2000. 131f. Dissertação (Mestrado) - Escola Politécnica, Universidade de São Paulo, São Paulo, 2000.

BARDANACHVILI, C. A. **Otimização multiobjetivo com estratégias evolutivas aplicada a projetos estruturais.** 2006. 104f. Tese (Doutorado) - Coordenação dos Programas de Pós Graduação de Engenharia - COPPE, Universidade Federal do Rio de Janeiro-UFRJ, Rio de Janeiro, 2006.

BAZARAA, M. S.; JARVIS, J. J.; SHERALI, H. D. **Linear programming and network flows**, 2.ed. United States of America: John Wiley & Sons, 1990. 673p.

BAZARAA, M. S.; SHERALI, H. D.; SHETTY, C. M. **Nonlinear programming:** theory and algorithms. 2.ed. United States of America: John Wiley & Sons, 1993. 638p.

BRANKE, J.; KAUBLER, T.; SCHMECK, H. **Guiding multi-objective evolutionary algorithms towards interesting regions.** Germany: Institute AIFB-University of Karsruhe, 2000. (Technical Report n. 399).

CASTRO, R. E. **Otimização de estruturas, com multiobjetivos via algoritmo genético de pareto**. 2001. 202f. Tese (Doutorado) - Coordenação dos Programas de Pós Graduação de Engenharia - COPPE, Universidade Federal do Rio de Janeiro - UFRJ, Rio de Janeiro, 2001.

COELLO, C. A. C. **An empirical study of evolutionary techniques for multiobjective optimization in engineering design.** 1996. Thesis (Ph) - Department of Computer Science, Tulane University, New Orleans, 1996.

DEB, K. An Efficient constraint handling method for genetic algorithms. **Computer Methods in Applied Mechanics and Engineering**, Amsterdam, v.186, p. 311-338, 2000.

DEB, K. **Multi-objective optimization using evolutionary algorithms.** Chichester: John Wiley & Sons, 2004. 515p.

DEB, K.; AGRAWAL, S.; PRATAP, A.; MEYARIVAN, T. **A Fast and elitist multi-objective genetic algorithm:** NSGA-II. Indian institute of technology, Kapur: Kapur Genetic Algorithms Laboratory, 2000a, (Technical Report, 200001).

DEB, K.; AGRAWAL, S.; PRATAP, A.; MEYARIVAN, T. A Fast and elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. 2000, **Proceedings of the Parallel problem solving from nature 6** – (PPSN-VI), 2000b, p. 849-858.

DEB, K.; ARGARWAL, S.; PRATAP, A.; MEYARIVAN, T. A Fast and elitist multi-objective genetic algorithm: NSGA II. **IEEE Transactions on Evolutionary Computation**, New York, v.6, p. 182-197, 2002.

DEB, K.; CHAUDHURI S. **I-EMO:** an interactive evolutionary multi-objective optimization tool. Kapur: Kapur Genetic Algorithms Laboratory, 2005. 9p. (Kangal Report Number 2005003).

DEB, K.; ZOPE, P.; JAIN, A. **Distributed computing of pareto-optimal solutions using multi-objective evolutionary algorithms**. Kapur: Kapur Genetic Algorithms Laboratory, 2002a. 18p. (Kangal Report Number 2002008).

FARMANI, R.; WRIGHT, J. Self-adaptive fitness formulation for constrained optimization, **IEEE Transactions on Evolutionary Computation,** New York, v.7, n.5, p. 445-455, 2003.

FONSECA, C. M.; FLEMING, P. J. Genetic algorithms for multi-objective optimization: formulation, discussion and generalization. In: GENETIC ALGORITHMS: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE, 5, 1993, San Francisco. **Proceedings of the...** San Francisco: Morgan Kaufmann: S.n., 1993. p.416-423.

FONSECA, C.; FLEMING, P. J. Multiobjective optimization and multiple constrained handling with evolutionary algorithms i: a unified formulation. **IEEE Transactions on Systems, Man Cybernetics,** New York, Part A**,** v.28, p. 26-37, 1998.

GLOVER, F.; KOCHENBERGER, G. A. **A Handbook of metaheuristic**. Norwell, MA: Kluwer Academic Plublishers, 2003. p. 457-474.

GRANADA, E. M.; ROMERO, R.; ZINI, E. O. C. A Generic algorithm for constrained optimization using multi-objective optimization. In: SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL, 40, 2008, João Pessoa. **Anais...** João Pessoa: SOBRAPO, 2008. 11p.

HORN, J.; NAFPLOITIS, N.; GOLDBERG, D. A Niched pareto genetic algorithm for multi-objective optimization. In: PROCEEDINGS OF THE CONFERENCE ON EVOLUTIONARY COMPUTATION, 1, IEEE WORLD CONGRESS ON COMPUTATIONAL INTELLIGENCE , 1994, Orlando. **Proceedings of the...** Orlando: IEEE, 1994. p. 82-87.

KOZIEL, S.; MICHALEWICZ Z. Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. **Evolutionary Computation,** Cambridge, v.7, p. 19-44, 1999.

LINDEN, R. **Algoritmos genéticos:** uma importante ferramenta da inteligência computacional. Rio de Janeiro: Brasport, 2006.

LUCAS, D. C. **Algoritmos genéticos:** uma introdução. Porto Alegre: Universidade Federal do Rio Grande do Sul - UFRS, 2002. 47p. (Material elaborado para a disciplina de ferramentas de inteligência artificial).

NESMACHNOW, S. **Una versión paralela del algoritmo evolutivo para optimización multiobjetivo nsga-ii y su aplicación al diseño de redes de comunicaciones confiables**. Uruguay: Centro de Cálculo, Instituto de Computación, Facultad de Ingeniería, Universidad de la República, 2004. 50p.

NEVES, F. A. **Programação com multi-objetivos aplicada à otimização do projeto de pontes estaiadas**. 1997. Tese (Doutorado)- Coordenação dos Programas de Pós Graduação de Engenharia - COPPE, Universidade Federal do Rio de Janeiro - UFRJ, Rio de Janeiro, 1997.

OLIVEIRA, A. C. M. **Algoritmos evolutivos para problemas de otimização numérica com variáveis reais.** São José dos Campos: Instituto Nacional de Pesquisas Espaciais - INPE, 2001. 52p. (Curso de Computação Aplicada).

PEÑUELA, C. A.; GRANADA, E. M. Optimización multiobjetivo usando un algoritmo genético y un operador elitista basado en un ordenamiento no-dominado: NSGA-II**. Revista Scientia et Technica,** Pereira, v.13, n.35, p.49-54, 2007.

POWELL, D.; SKOLNICK, M. Using genetic algorithms in engineering design optimization with nonlinear constraints. In: GENETIC ALGORITHMS: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE, 5, 1993, San Francisco. **Proceedings of the...** San Francisco: Morgan Kaufmann, 1993. p. 424-431.

POZO, A.; CAVALHEIRO A. F.; ISHIDA, C.; SPINOSA, E.; RODRIGUES E. M. **Computação evolutiva.** Universidade Federal do Paraná, 61p. (Grupo de Pesquisas em Computação Evolutiva, Departamento de Informática-Universidade Federal do Paraná)

RENDÓN, R. A. G.; ZULUAGA, A. H. E.; ROMERO, R. **Tecnicas de optimizacion combinatorial.** Pereira: La Universidad Tecnológica de Pereira, 2006. 176p.

RICHARDSON J.; PALMAR, M.; LIEPUS, G.; HILLARD, M. Some guidelines for genetic algorithms with penalty functions. In: INTERNATIONAL CONFERENCE ON GENETIC ALGORITHM, 3, San Mateo, 1989. **Proceedings of the...** San Mateo: S.n., 1989. p. 191-197.

ROMERO, R. **Planejamento de sistemas de transmissão usando algoritmo genético.** Ilha Solteira: UNESP, 2005. (Notas de Aula).

ROMERO, R.; MANTOVANI J. R. S. Introdução a metaheurística. In: CONGRESSO DE DINÂMICA E CONTROLE DA SBMAC, DINCON, 3, 2004, Ilha Solteira. **Anais...** Ilha Solteira; Unesp/FEIS, 2004. (CD- ROM)..

RUNARSSON, T.; YAO, X. Stochastic ranking for constrained evolutionary optimization. **IEEE Transactions on Evolutionary Computation**, New York, v.4, p. 344-354, 2000.

SCHAFFER, J. D. **Multiple objective optimization with vector evaluated genetic algorithm**s. Thesis (PhD), Vanderbilt University, Nashville, 1984.

SRINIVAS, N.; DEB, K. Multi-objective function optimization using non-dominated sorting genetic algorithms. **Evolutionary Computation,** Cambridge, v.2, n.3, p. 221-248, 1994.

SURRY, P.; RADCLIFFE, N. J.; BOYD, I. D. A Multi-objective approach to constrained optimization of gas supply networks: the comoga method. In: AISB-95 WORKSHOP ON EVOLUTIONARY COMPUTATING, Sheffield, p. 166-180, 1995.

TAKAHASHI, R. H. C. **Otimização escalar e vetorial.** Belo Horizonte: Departamento de Matemática, Universidade Federal de Minas Gerais - UFMG, 2004. (Notas de Aula).

VENKATRAMAN, S.; YEN, G. G. A Generic framework for constrained optimization using genetic Algorithms. **IEEE Transactions on Evolutionary Computation**, New York, v.9, n.4, p. 424-435, 2005.

WOLPERT, D.; MACREADY, W. No free lunch theorems for optimization. **IEEE Transactions on Evolutionary Computation**, New York, v.l, p. 67-82, 1997.

ZINI, E. O. C. **Introdução à programação linear.** 2005, 65f., Trabalho de Conclusão de Curso (Graduação em Matemática) - Departamento de Matemática, Universidade Federal do Mato Grosso do Sul - UFMS, Três Lagoas, 2005.